Oracle Rdb7
OpenVMS Mixed Cluster

DIGITAL HiTest Notes

Part Number: EK-HORVX-HN. A01

March 1997

This document describes the interoperability testing performed on an Oracle Rdb7 Data Warehousing environment using OpenVMS V7.1 running on an AlphaServer 8400 system and AlphaServer 4100 system in a CI-based mixed cluster with VAX 6620 systems and HSJ40 Controller subsystems.

Revision/Update Information: This is a new manual
Operating System and Version: OpenVMS V7.1
March 1997

Digital Equipment Corporation makes no representations that the use of its products in the manner described in this publication will not infringe on existing or future patent rights, nor do the descriptions contained in this publication imply the granting of licenses to make, use, or sell equipment or software in accordance with the description.

Possession, use, or copying of the software described in this publication is authorized only pursuant to a valid written license from DIGITAL or an authorized sublicensor.

© Digital Equipment Corporation 1997. All rights reserved.

The following are trademarks of Digital Equipment Corporation: AlphaServer, DECevent, DIGITAL, DIGITAL NAS, OpenVMS, POLYCENTER, ServerWORKS, StorageWorks, VAX, VAXcluster, and the DIGITAL logo.

The following are third-party trademarks:
Oracle is a registered trademark and Oracle Rdb7 is a trademark of Oracle Corporation.

All other trademarks are the property of their respective owners.
Table of Contents

1 Introduction .................................................................................... 1–1
   DIGITAL HiTest Suite and Its Advantages ................................................. 1–1
   Overview of this DIGITAL HiTest Suite ..................................................... 1–2

2 Configuration Data ........................................................................ 2–1
   Hardware and Software Components .......................................................... 2–1
   Special Configuration Rules ....................................................................... 2–6

3 Installation and Setup ................................................................... 3–1
   Setting Up the AlphaServer Systems ............................................................ 3–1
   Setting Up Storage ...................................................................................... 3–1
   Installing the Operating System ................................................................. 3–2
   Installing Layered Products ....................................................................... 3–2
   Modifying Account Quotas ....................................................................... 3–3
   Creating and Loading the Database ............................................................ 3–5

4 Interoperability Tests and Results................................................. 4–1
   Test Configuration Overview ...................................................................... 4–2
      Major Components ................................................................................... 4–3
      Storage Subsystem .................................................................................. 4–3
   Test Scenarios ............................................................................................ 4–3
   Test Database ............................................................................................. 4–4
      Database Queries .................................................................................... 4–5
   Test Scripts .................................................................................................. 4–6
      Characterization of the Queries ............................................................... 4–7
         Query 1 .................................................................................................. 4–7
         Query 2 .................................................................................................. 4–8
         Query 3 .................................................................................................. 4–9
         Query 4 .................................................................................................. 4–10
         Query 5 .................................................................................................. 4–11
   Test Results ................................................................................................ 4–12
      Test One: SQL Queries in Parallel ......................................................... 4–12
      Test Two: 5- and 10-User Load Tests ....................................................... 4–13
5 System Limits and Performance Data ..................................................5–1

6 Problems and Resolutions .................................................................6–1

Oracle Rdb7 Database .................................................................................................. 6–1
DIGITAL System/Storage Hardware ............................................................................. 6–1
OpenVMS Operating System....................................................................................... 6–2

A Detailed Hardware Configuration ......................................................... A–1

Graphic Overview (Golden Egg-like) ................................................................. A–2
AlphaServer 8400 Slot Usage .................................................................................. A–3
AlphaServer 4100 Slot Usage .................................................................................. A–4
Database Storage Setup ......................................................................................... A–6
Database Storage Maps......................................................................................... A–7

B Operational Command and Supporting Files........................................ B–1

Figures

Figure 4-1: Diagram of the Test Cluster ..................................................................... 4–2

Tables

Table 2-1: DIGITAL HiTest Template for AlphaServer 8400........................................ 2–2
Table 2-2: DIGITAL HiTest Template for AlphaServer 4100........................................ 2–4
Table 2-3: Component Revision Levels, AlphaServer 8400 System............................ 2–6
Table 2-4: Component Revision Levels, AlphaServer 4100 System............................ 2–6
Table 2-5: Component Revision Levels, Installed Software ........................................ 2–6
Table 4-1: The CPG Database .................................................................................. 4–4
Table 4-2: Data Distribution Across Channels ......................................................... 4–4
Table 4-3: Allocation of Row Cache Buffers .............................................................. 4–6
Table 4-4: Cold Cache Test Results for AlphaServer 4100 System............................. 4–12
Table 4-5: Warm Cache Results for AlphaServer 4100 System .................................. 4–12
Table 4-6: AlphaServer 4100 System 5- and 10-User Load Test Results..................... 4–13
Table 4-7: AlphaServer 8400 System 5- and 10-User Load Test Results..................... 4–13
Table 6-1: AlphaServer 8400 System Bus Usage .................................................... A–3
Table 6-2: AlphaServer 8400 I/O Shelf Usage .......................................................... A–3
Table 6-3: AlphaServer 4100 PCI Motherboard Usage .............................................. A–4
Table 6-4: AlphaServer 4100 System Motherboard Usage ........................................ A–5
Table 6-5: Database Storage Map for HSJ001......................................................... A–7
Table 6-6: Database Storage Map for HSJ002......................................................... A–8
Table 6-7: Database Storage Map for HSJ003......................................................... A–9
Table 6-8: Database Storage Map for HSJ004......................................................... A–10
Table 6-9: Database Storage Map for HSJ005......................................................... A–11
Table 6-10: Database Storage Map for HSJ006....................................................... A–12
Table 6-11: Database Storage Map for HSJ007....................................................... A–13
Table 6-12: Database Storage Map for HSJ008....................................................... A–14
Table 6-13: Database Storage Map for HSJ009....................................................... A–15
Preface

This document provides an overview of DIGITAL HiTest Systems and detailed technical information about interoperability test results for the DIGITAL HiTest Template for Oracle Rdb7 on OpenVMS Mixed Cluster.

Audience

Primary users of this document are DIGITAL and Partners sales representatives and customers. Secondary audiences include technical support personnel, product managers, and the personnel responsible for installing, setting up, and operating a DIGITAL HiTest System.

Road Map

This document contains the following sections:

1. Introduction — Provides a brief summary of the benefits of DIGITAL HiTest Templates and an overview of the Template covered in this document.

2. Configuration Data — Gives tables of configuration data about the hardware and software components that define the Template, and special configuration rules if any.

3. HiTest System Installation and Setup — Presents information useful when installing and tuning a DIGITAL HiTest System configured from this DIGITAL HiTest Template.

4. Interoperability Tests and Results — Describes how the tests were set up (including database organization), what data and programs were placed on what disks, and how the tests were run.

5. System Limits and Performance Data — Summarizes any system limitations or performance data that were identified during testing.

6. Problems and Solutions — Discusses any problems and solutions that were discovered during testing.

Appendix A: Detailed Hardware Configuration — Contains a more detailed treatment of the hardware and software components listed in the Configuration Data section.

Appendix B: Describes the files that were used to setup, load or run tests on the test database.

Feedback and Ordering Information

What our readers think of this or any other DIGITAL documentation is important to us. If you have any comments, no matter how great or small, we'd appreciate hearing from you. Send your comments to: reader-comments@digital.com.
Please reference the document title and part number in your correspondence about this manual.

Copies of this and other DIGITAL documents can be ordered by calling 1-800-DIGITAL.
Introduction

DIGITAL HiTest Suite and Its Advantages

*DIGITAL HiTest Suites* are guidelines for configuring a set of prequalified computer systems. A HiTest Suite often contains all the hardware and software needed for a complete customer solution. DIGITAL HiTest Suites can be used as a basis for configuring systems that satisfy a wide set of customer requirements. Typically, Suites target specific markets such as Data Warehousing or Continuous Computing.

DIGITAL Product Management and Engineering select the components and design the configurations in each HiTest Suite to ensure high system reliability, application performance, and upgradability. A Suite's hardware and software components have been successfully tested for interoperability.

A HiTest Suite specifies allowed ranges of hardware and software components, as well as each component's part number, description, and revision information. These specifications are listed in the *DIGITAL HiTest Template*.

The components in a HiTest Suite are organized into two groups, *the DIGITAL HiTest Foundation* and the *DIGITAL HiTest AppSet*. The HiTest Foundation includes the hardware, operating system, middleware, and database software. The HiTest AppSet includes the software specific to one class of customer solutions.

Configuring a DIGITAL HiTest Suite is easy. Simply select components from the HiTest Template to configure a DIGITAL HiTest System. Any system configured as specified in the DIGITAL HiTest Template can be called a DIGITAL HiTest System.

The HiTest Suite is documented in the *DIGITAL HiTest Notes*. The HiTest Notes list the HiTest Foundation and HiTest AppSet components. HiTest Notes also describe the testing of the Suite and include configuration details, installation instructions, tuning parameters, problems encountered and their solutions, and system diagrams.

Some components listed in the HiTest Foundation or AppSet may be optional. If the minimum quantity is zero (0), then the component is optional. If the minimum quantity is one or more, then you must order at least the minimum quantity.

The maximum quantities represent the largest group of components that were tested for interoperability with all the other components in the Suite. Although it may be possible to place more than the specified maximum quantity of a component on a DIGITAL system, extensive interoperability testing was not done at that level and such a system would not be considered a DIGITAL HiTest System.
You can select any combination of components with quantities ranging from the minimum to the maximum specified. Occasionally, special configuration rules give further guidance or restrict configurations. These rules appear in the Configuration Data section of the HiTest Notes.

A customer can include the Suite-specified hardware and software they need and then layer on additional software. Other types of hardware, called add-on hardware, can also be added to a DIGITAL HiTest System. The add-on hardware is specified in the Configuration Data section of the HiTest Notes, and in the HiTest Systems Web Pages, available through the following URLs:

- http://cosmo.tay.dec.com (Intranet)
- http://www.partner.digital.com:9003 (Internet)

Even though the customer may install application software that is not specified in the Suite, the customer and DIGITAL still experience the advantages of knowing that all of the Suite base hardware and software interoperates correctly. Of course, the full benefit of configuring a system from a HiTest Suite is obtained when the system includes only specified HiTest Foundation and AppSet components.

**Overview of this DIGITAL HiTest Suite**

This AlphaServer 8000 HiTest Template, along with Oracle Rdb7, supports production applications, end-user information management, and transaction processing to provide integration of the enterprise at the information level. This CI-cluster add-on configuration has the power and the flexibility to manage critical business data and production environments demanding 24x365 availability.
Configuration data includes the hardware, software, and firmware components that were tested together. Special configuration rules are explained if required.

**Hardware and Software Components**

The following tables identify the hardware and software components that can be configured when upgrading an existing VAXcluster used for data warehousing. This upgrade can consist of the Oracle Rdb7 OpenVMS AlphaServer 8400 DIGITAL HiTest Template shown in Table 2-1 or the Oracle Rdb7 OpenVMS AlphaServer 4100 DIGITAL HiTest Template shown in Table 2-2 or both.

The remaining tables list the component revision levels for the AlphaServer systems and installed software.
### Table 2-1: Oracle Rdb7 OpenVMS AlphaServer 8400 DIGITAL HiTest Template

**Oracle Rdb7 HiTest Appset**  
OpenVMS AlphaServer 8400 HiTest Foundation

*NOTE: Storage is assumed to be part of the cluster to which this system is attached.  
For documentation and updates: [http://cosmo.ta.y.de.com](http://cosmo.ta.y.de.com) and [http://www.partner.digital.com:9003](http://www.partner.digital.com:9003)  
For hard copy of this Suite’s HiTest Notes, order EK-HORVX-HN .A01*

<table>
<thead>
<tr>
<th>Line Item</th>
<th>Description</th>
<th>Part Number</th>
<th>Tested Range</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Appset Software</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>Oracle Rdb7 Release 7</td>
<td>Call</td>
<td>0 1</td>
</tr>
<tr>
<td>2</td>
<td>Oracle SQL/Services</td>
<td>Included with item 1</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Oracle Parallel Query &amp; Distributed Option</td>
<td>Included with item 1</td>
<td></td>
</tr>
<tr>
<td><strong>Foundation Hardware</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
| 4         | Select just one base system:  
*NOTE: DA = 60 Hz, 208 V; DB = 50 Hz, 380/416 V; DC = 50/60 Hz, 202 V Japan* | DY-292FF-DA  
DY-282FF-D9 | 1 1 |
| **AlphaServer 8400 5/440 CPU 2 GB OVMS** | | | |
| **AlphaServer 8200 5/440 CPU 2 GB OVMS** | | | |
| Hardware includes: | | | |
| • 2 5/440-MHz CPUs each with 4-MB cache  
• KFTHA-AA system I/O module  
• 2 GB Memory  
• KZPSA-BB FWD SCSI Controller with cable  
• KZPDA-AA FNS SCSI Controller with cable  
• DE500-XA Fast Ethernet Adapter  
• BA660-AB StorageWorks Plug-in unit (PIU)  
• DWLPB-AA PCI plug in unit  
• RZ28D-VW 2 GB 3.5 in. SCSI disk  
• RRDCD 600 MB CD-ROM drive  
• DWZZB-VW SCSI signal converter  
• Three-phase power subsystem with cord  
• 1 H7263-AC or H7263-AD non-BBU capable  
• 48 VDC power regulator  
• Shielded console cable for console terminal  
• System cabinet | | |
| Software includes: | | | |
| • OpenVMS base license  
• DIGITAL Enterprise Integration Package (EIP) | | |
| 5 | 440 MHz CPU SMP upgrade (OpenVMS) | 756P1-AX | 0 5 |
| 6 | 2 GB memory module | MS7CC-FA | 0 1 |
| 7 | Order items 7 and 8 to go beyond 12 PCI slots: PCI plug-in unit with one PCI box‡ | DWLPB-AA | 0 1 |
| 8 | Second PCI expansion box for DWLPB-AA‡ | DWLPB-BA | 0 1 |
| 9 | PCI to CI adapter | CIPCA-AA | 1 2 |
| 10 | Select one for each CIPCA: CI bus cable set (select required length: 10, 20, or 45 m) | BNCIA-xx | 1 2 |
| 11 | VT525 text terminal | VT525-AA | 0 1 |
NOTE: Storage is assumed to be part of the cluster to which this system is attached.
For hard copy of this Suite’s HiTest Notes, order EK-HORVX-HN .A01

<table>
<thead>
<tr>
<th>Line Item</th>
<th>Description</th>
<th>Part Number</th>
<th>Tested Range</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><strong>Foundation Software</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>--</td>
<td>OpenVMS base license</td>
<td>Included with item 4</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>OpenVMS V7.1 CD-ROM media and doc.</td>
<td>QA-MT1AA-H8</td>
<td>0 1</td>
</tr>
<tr>
<td>13</td>
<td>OpenVMS V7.1 unlimited license</td>
<td>QL-MT2AG-AA</td>
<td>0 1</td>
</tr>
<tr>
<td>14</td>
<td>DIGITAL Enterprise Integration Package (EIP)</td>
<td>Included with item 4</td>
<td></td>
</tr>
<tr>
<td></td>
<td><em>For items 15 to 18, Refer to OpenVMS Layered Products CD-ROM (QA-03XAA-H8)</em></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>DEC C for OVMS Alpha V5.5</td>
<td></td>
<td>0 1</td>
</tr>
<tr>
<td>16</td>
<td>DEC C++ for OVMS Alpha V 5.5</td>
<td></td>
<td>0 1</td>
</tr>
<tr>
<td>17</td>
<td>DECevent V2.3</td>
<td></td>
<td>0 1</td>
</tr>
<tr>
<td>18</td>
<td>UCX V4.1 (for PCM use only)</td>
<td></td>
<td>0 1</td>
</tr>
</tbody>
</table>
### Table 2-2: Oracle Rdb7 OpenVMS AlphaServer 4100 DIGITAL HiTest Template

**Oracle Rdb7 HiTest Appset**  
OpenVMS AlphaServer 4100 HiTest Foundation

*NOTE: Storage is assumed to be part of the cluster to which this system is attached. For documentation and updates: http://cosmo.ta.y.dec.com and http://www.partner.digital.com:9003. For hard copy of this Suite's HiTest Notes, order No. EK-HORVX-HN.A01*

<table>
<thead>
<tr>
<th>Line Item</th>
<th>Description</th>
<th>Part Number</th>
<th>Tested Range</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Appset Software</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>Oracle Rdb7 Release 7</td>
<td>Call</td>
<td>0 1</td>
</tr>
<tr>
<td>2</td>
<td>Oracle SQL/Services</td>
<td>Included with item 1</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Oracle Parallel Query &amp; Distributed Option</td>
<td>Included with item 1</td>
<td></td>
</tr>
</tbody>
</table>

**Foundation Hardware**

<table>
<thead>
<tr>
<th>Line Item</th>
<th>Description</th>
<th>Part Number</th>
<th>Tested Range</th>
</tr>
</thead>
</table>
| 4 | Select just one base system:  
AS4100 5/400 Drawer OVMS  
AS4000 5/400 Drawer OVMS | DY-51HAB-FB  
DY-52HAB-FB | 1 1 |
|     | Hardware includes:  
• 5/400-MHz CPU with 4-MB cache  
• Integral FNSE SCSI and CD-ROM drive  
• 1.44 MB diskette drive  
• 450 watt power supply  
• 1 GB Memory  
• PCI 10/100 Mbit fast Ethernet controller  
• One-port FWSE SCSI Controller  
• S3 TRIO 1 MB RAM Graphics Adapter  
• 3-button mouse  
• Keyboard (Americas and AP orders only)  
• Integral remote system console  
|     | Software includes:  
• OpenVMS license  
• DIGITAL Enterprise Integration Package (EIP) |  |

Where two part numbers or variants are separated by a "/", the first number applies to the Americas and Asia Pacific and the second number applies to Europe.

<table>
<thead>
<tr>
<th>Line Item</th>
<th>Description</th>
<th>Part Number</th>
<th>Tested Range</th>
</tr>
</thead>
</table>
| 5 | Select one enclosure:  
Pedestal kit AS4100/4000  
or  
19 in. RETMA cabinet  
Both include:  
• StorageWorks shelf and 4.3 GB hard drive for Americas, AP | BA30P-AB/BB  
H9A10-EL/EM | 1 1 |

This HiTest Template supports a memory range from 1 GB to 4 GB. The AlphaServer 4100 System Drawer supports up to three additional memory options.

<table>
<thead>
<tr>
<th>Line Item</th>
<th>Description</th>
<th>Part Number</th>
<th>Tested Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>400 MHz CPU SMP Upgrade (OpenVMS)</td>
<td>KN304-BB</td>
<td>0 3</td>
</tr>
</tbody>
</table>
| 7 | Needed with 3 or more CPUs:  
450 watt power supply | H7291-AA | 0 1 |

1 GB memory option

| 8 | 1 GB memory option | MS330-FA | 0 3 |
| 9 | PCI to CI host bus adapter | CIPCA-AA | 1 2 |
| 10 | Select one for each CIPCA:  
CI bus cable set (select required length: 10, 20, or 45 m) | BNCIA-xx | 1 2 |
Oracle Rdb7 HiTest Appset
OpenVMS AlphaServer 4100 HiTest Foundation

NOTE: Storage is assumed to be part of the cluster to which this system is attached.
For hard copy of this Suite’s HiTest Notes, order No. EK-HORVX-HN.A01

<table>
<thead>
<tr>
<th>Line Item</th>
<th>Description</th>
<th>Part Number</th>
<th>Tested Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>Select one high resolution color monitor:</td>
<td>VRC15-WA</td>
<td>0 1</td>
</tr>
<tr>
<td></td>
<td>15-in Flat-square with 0.28 dot pitch</td>
<td>VRT17-WA</td>
<td></td>
</tr>
<tr>
<td></td>
<td>17-in Trinitron aperture grille, 0.26mm</td>
<td>VRC21-WA</td>
<td></td>
</tr>
<tr>
<td></td>
<td>21-in Diamondtron aperture grille, 0.30mm</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Foundation Software

Order the exact versions and revisions of the software shown below.
Paper documentation can be ordered separately.

<table>
<thead>
<tr>
<th>Line Item</th>
<th>Description</th>
<th>Tested Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td>OpenVMS base license</td>
<td>Included with item 4</td>
</tr>
<tr>
<td>13</td>
<td>OpenVMS V7.1 unlimited license</td>
<td>QL-MT2AG-AA</td>
</tr>
<tr>
<td>14</td>
<td>DIGITAL Enterprise Integration Package (EIP)</td>
<td>Included with item 4</td>
</tr>
</tbody>
</table>

For items 15 to 18, Refer to OpenVMS Layered Products CD-ROM (QA-03XAA-H8)

<table>
<thead>
<tr>
<th>Line Item</th>
<th>Description</th>
<th>Tested Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>DEC C for OVMS Alpha V5.5</td>
<td>$\dfrac{0}{1}$</td>
</tr>
<tr>
<td>16</td>
<td>DEC C++ for OVMS Alpha V 5.5</td>
<td>$\dfrac{0}{1}$</td>
</tr>
<tr>
<td>17</td>
<td>DECevent V2.3</td>
<td>$\dfrac{0}{1}$</td>
</tr>
<tr>
<td>18</td>
<td>UCX V4.1 (for PCM use only)</td>
<td>$\dfrac{0}{1}$</td>
</tr>
</tbody>
</table>

For more details on the hardware configuration, see Appendix A.
The following tables identify component revision levels for the AlphaServer systems and software in the foregoing Templates.

### Table 2-3: Component Revision Levels, AlphaServer 8400 System

<table>
<thead>
<tr>
<th>Hardware Component</th>
<th>Hardware</th>
<th>Firmware</th>
<th>Software</th>
</tr>
</thead>
<tbody>
<tr>
<td>8400 console</td>
<td>4.8-6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SRM console</td>
<td>3.1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5/440 MHz CPU OVMS (756P1-AX) (qty 6)</td>
<td>E02</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2 GB memory module (MS7CC-FA) (qty 2)</td>
<td>B01</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2 GB 3.5 in. disk (RZ28D-AA)</td>
<td>B03</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PCI to CI Host Bus Adapter (CIPCA-AA)</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>System I/O module (KFTHA-AA)</td>
<td>D03</td>
<td></td>
<td></td>
</tr>
<tr>
<td>StorageWorks plug-in unit (BA660-AB)</td>
<td>A01</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PCI plug-in unit (DWLPB-AA) (qty 2)</td>
<td>A02</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PCI plug-in unit (DWLPB-BA)</td>
<td>A02</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Table 2-4: Component Revision Levels, AlphaServer 4100 System

<table>
<thead>
<tr>
<th>Hardware Component</th>
<th>Hardware</th>
<th>Firmware</th>
<th>Software</th>
</tr>
</thead>
<tbody>
<tr>
<td>SRM console</td>
<td>V4.8-5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fast/Wide SCSI Controller (KZPDA-AA)</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.3 GB Fixed Wide Disk (RZ29B-VW)</td>
<td>0016</td>
<td></td>
<td></td>
</tr>
<tr>
<td>400 MHz CPU OVMS (KN304-BB) (qty 4)</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 GB Memory Option (MS330-FA) (qty 4)</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fast Ethernet adapter (DE500-XA)</td>
<td>12</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PCI-CI Host Bus Adapter (CIPCA-AA)</td>
<td>1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Table 2-5: Component Revision Levels, Installed Software

<table>
<thead>
<tr>
<th>Software Component</th>
<th>Version/ Revision</th>
<th>Patch Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>OpenVMS</td>
<td>7.1</td>
<td></td>
</tr>
<tr>
<td>DEC C</td>
<td>V5.5</td>
<td></td>
</tr>
<tr>
<td>DEC C++</td>
<td>V5.5</td>
<td></td>
</tr>
<tr>
<td>DECevent</td>
<td>V2.3</td>
<td></td>
</tr>
<tr>
<td>UCX</td>
<td>V4.1</td>
<td>ECO2</td>
</tr>
<tr>
<td>Oracle Rdb7</td>
<td>Release 7</td>
<td>ECO1</td>
</tr>
<tr>
<td>Oracle SQL/Services</td>
<td>V7.0</td>
<td></td>
</tr>
<tr>
<td>Oracle Parallel Query &amp; Distributed</td>
<td>Release 7</td>
<td></td>
</tr>
</tbody>
</table>

### Special Configuration Rules

None
The following information provides guidelines for system installation and setup.

Setting Up the AlphaServer Systems

Installation of the AlphaServer 8400 and AlphaServer 4100 systems must be performed by authorized Digital Service personnel. Installation procedures are provided with the systems when shipped. Installation manuals are also provided with the options ordered, when appropriate.

Appendix A contains the detailed hardware configuration for the systems that were used for testing. It begins with a detailed graphical view of the test cluster configuration. The slot utilization for the AlphaServer 8400 system can be found in Table 6-1 and Table 6-2. The slot utilization for the AlphaServer 4100 system can be found in Table 6-3 and Table 6-4.

Setting Up Storage

It is understood that a customer’s database and disk layout may already exist. The following information describes how the test disk farm was laid out.

POLYCENTER Console Manager (PCM) was used in the test environment for convenience in performing the following functions:

- Shutting down systems
- Rebooting systems
- Running standalone diagnostics
- Installing layered products
- Configuring HSJ controllers
- Monitoring system and HSJ controller status

PCM was also used to manage the test HSJ configuration and capture failure messages, alerting us to the need for repair or replacement of faulty disk drives. Alarm conditions were reported to a single workstation using PCM. HSJ40 consoles were connected to terminal servers, which were reported as nodes to the PCM, simplifying the management of our large configuration of over 300 disk drives.
PCM was used to set up disks as follows:

1. Configured 39 RAID-0 stripesets (four disks per set) for database storage. Refer to *StorageWorks Array Controllers: HS Family of Array Controllers User Guide*, order number EK-HSFAM-UG.

2. Configured two RAID-5 arrays (six disks per array) as OpenVMS VAX and OpenVMS Alpha operating system disks. (Refer to Appendix A for detailed storage maps.)

3. Used the remaining 35 RAID-0 stripesets for temp and sort space during the database load.

______________________________ Note ___________________________

Write back cache was enabled for all drives attached to the HSJ40 Controllers.

______________________________ Note ___________________________

**Installing the Operating System**

Install standard OpenVMS VAX V7.1 and OpenVMS Alpha V7.1 operating systems using all the default values.

**Installing Layered Products**

Install the following layered products, using all the standard defaults.

- DECEvent V2.3
- DEC C V5.5
- DEC C++ V5.5
- UCX V4.1
- Oracle Rdb7 Release 7
- Oracle SQL/Services
- Oracle Parallel Query & Distributed Option

______________________________ Note ___________________________

POLYCENTER Data Collector and POLYCENTER Performance Advisor would normally have been installed but were not currently supported on OpenVMS V7.1.
Modifying Account Quotas

The following quotas and parameters reflect the values that were used and that proved adequate for testing. Knowledge of system tuning is required before modifying any of these parameters. All testing was conducted from the RDB and SYSTEM accounts.

**RDB Account Quotas**

Maxjobs: 0 Fillm: 300 Bytlm: 65536
Maxacctjobs: 0 Shrfillm: 0 Phytlm: 0
Maxdetach: 0 BIOlm: 200 JTquota: 8192
Prclm: 10 DIOlm: 200 WSdef: 1024
Prio: 4 ASTlm: 300 WSquo: 2048
Queprio: 4 TQElm: 200 Wsextent: 16384
CPU: (none) Enqlm: 2000 Pgflquo: 110000

**SYSTEM Account Quotas**

Maxjobs: 0 Fillm: 300 Bytlm: 65536
Maxacctjobs: 0 Shrfillm: 0 Phytlm: 0
Maxdetach: 0 BIOlm: 200 JTquota: 8192
Prclm: 10 DIOlm: 200 WSdef: 1024
Prio: 4 ASTlm: 300 WSquo: 2048
Queprio: 0 TQElm: 200 Wsextent: 16384
CPU: (none) Enqlm: 2000 Pgflquo: 300000

The following examples of the MODPARAMS.DAT file contain system parameters that were altered to accommodate the database load and query execution.

**MODPARAMS.DAT File for AlphaServer 8400**

```
!++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
! SYS$SYSDEVICE:[SYS1.SYSEXE]MODPARAMS.DAT
! Created during upgrade to OpenVMS AXP V7.1 29-JAN-1997 11:39:23.59
!
! This is a new file created by the OpenVMS upgrade procedure. This file was built by using the data found in the following file(s) previously used by this system:
!
!      SYS$SYSDEVICE:[SYS1.SYSEXE]ALPHAVMSSYS.PAR
!      SYS$SYSDEVICE:[SYS1.SYSEXE]MODPARAMS.DAT
!
! This/These old file(s) have been renamed to:
!
!      SYS$SYSDEVICE:[SYS1.SYSEXE]ALPHAVMSSYS.PAR_OLD
!      SYS$SYSDEVICE:[SYS1.SYSEXE]MODPARAMS.DAT_OLD
!
! A new
!
!      SYS$SYSDEVICE:[SYS1.SYSEXE]ALPHAVMSSYS.PAR
!
! has been built for you in order to ensure compatibility with this release. Previous parameters found to be larger than new defaults were retained. Certain other previous parameters were also retained.
!
! Please check the following sections of this file to see what files were used in what sequence to create the new APLHAVMSSYS.PAR file. Please review and edit this file for possible duplications, additions and deletions you wish to make.
!
--------------------------------------------------------------------------------
```

Oracle Rdb7 OpenVMS Mixed Cluster DIGITAL HiTest Notes 3–3
INSTALLATION AND SETUP

This section contains System Parameters found in
SYS$SYSDEVICE:[SYS1.SYSEXE]ALPHAVMSSYS.PAR
with values that must be preserved when AUTOGEN is run.

SCSSYSTEMID=64520
SCSNODE=DEPOT8
VAXCLUSTER=2
EXPECTED_VOTES=1
VOTES=1
RECNXINTERVAL=20
DISK_QUORUM=
QDSKINTERVAL=10
ALLOCLASS=1
LOCKDIRWT=3
NISCS_CONV_BOOT=0
NISCS_LOAD_PEA0=1
NISCS_PORT_SERV=0
MSCP_LOAD=1
MSCP_SERVE_ALL=1
min_NPAGEDYN=50000000
! increased by ESAC eng. on 17-Feb-1997
! set by Oracle Engineering
! CHANNELCNT=16384
VCC_FLAGS=1
VCC_MAXSIZE=204800
min_GBLPAGES=512000
min_GBLPAGFIL=512000
min_LOCKDIRWT=1100000
min_RESHASHTBL=1000000
min_PQL_DASTLM=16384
min_PQL_MASTLM=16384
min_PQL_DBIOILM=16384
min_PQL_MDBIOILM=16384
min_PQL_DDIOILM=16384
min_PQL_MDDIOILM=16384
min_PQL_DFILLM=16384
min_PQL_MDFILLM=16384
min_PQL_DPRCLM=16384
min_PQL_MPRCLM=16384
min_PQL_DQELM=16384
min_PQL_MQELM=16384
min_PQL_DENQLM=16384
min_PQL_MENQLM=16384
min_PQL_DPGFLQUOTA=5000000
min_PQL_MPGFLQUOTA=5000000

MODPARAMS.DAT File for AlphaServer 4100

+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
! SYS$SYSDEVICE:[SYS0.SYSEXE]MODPARAMS.DAT
! Created during upgrade to OpenVMS AXP V7.1 29-JAN-1997 11:39:12.22
! This is a new file created by the OpenVMS upgrade procedure. This file
! was built by using the data found in the following file(s) previously
! used by this system:
! SYS$SYSDEVICE:[SYS0.SYSEXE]ALPHAVMSSYS.PAR
! SYS$SYSDEVICE:[SYS0.SYSEXE]MODPARAMS.DAT
! This/These old file(s) have been renamed to:
! SYS$SYSDEVICE:[SYS0.SYSEXE]ALPHAVMSSYS.PAR_OLD
! SYS$SYSDEVICE:[SYS0.SYSEXE]MODPARAMS.DAT_OLD
! CHANNELCNT=16384
VCC_FLAGS=1
VCC_MAXSIZE=204800
min_GBLPAGES=512000
Creating and Loading the Database

The Oracle Rdb7 OVMS HiTest System was designed such that a series of custom .COM, .SQL, and other supporting files could be used to set up and configure the test system. These files were not part of a stock system but were created by Oracle Corporation. Descriptions of these files are provided in Appendix B.

These files performed the following functions.

1. Allocating host storage
2. Creating the database storage files
3. Compiling and linking flat file builder program
4. Generating 24 or 72 input streams
5. Loading the database
6. Creating indexes and journals (needed for VLM)
7. Creating the query catalog (see note below)
8. Executing queries

_____________________________ Note____________________________

In this implementation of Oracle Rdb7, two separate databases were created. These were for the 1994 and 1995 data. In cases where setup files are year specific, the year (94 or 95) appears in the file name.

______________________________

Unless otherwise specified, the command files, SQL scripts, or supporting files shown below are located in the top level [RDB] account directory.
Step 1: Allocating database storage

In this particular test, the AlphaServer 4100 (DEPOT7) was used to hold the 1994 database and the .TMP2 files were created for this particular system as were the .SRT2 file shown below. The AlphaServer 8400 (DEPOT8) was used to create and own the 1995 database.

The database was not automatically brought on-line at boot time. When bringing the system up for the first time, the following command files were executed from the RDB account:

$ @INIT DISKS.9x  
$ @INIT DISKS.SRT (and DISKS.SRT2)  
$ @INIT DISKS.ROO  
$ @LOGICALS DISKS.9x  
$ @LOGICALS DISKS.SRT  
$ @LOGICALS DISKS.SRT2  
$ @LOGICALS DISKS.ROO

Note

DISKS.SRT and DISKS.SRT2 started out with a list of 8 disks to initially build the database. Only 3 disks were needed to actually run the queries, so this file was modified after the initial build.

If the system needed to simply be rebooted after it was set up, the MOUNT.COM file was used in place of the INIT.COM shown above.

For use in initial creation of the database only, the following was executed:

$ @INIT DISKS.TMP (and DISKS.TMP2)  
$ @LOGICALS DISKS.TMP (and DISKS.TMP2)

The above commands created and initialized the TEMP space needed to hold the flat file data.

To establish the search list logicals, the following was executed:

$ @CPG.COM

Step 2: Creating the database storage files

For each AlphaServer system, the following command was used to create the files for an empty database:

$ SQL @CPG94.SQL on AlphaServer 4100  
$ SQL @CPG95.SQL on AlphaServer 8400

Note

The preceding commands also created the small (empty) dimension tables in addition to the sales facts table. The schema design for the database was influenced by the queries in that they performed by year query comparisons for an item (i.e., channel) to a group, and unioned the two result sets together. As all queries are by year, for a given channel or product, partitioning by year, channel and product was chosen. This was the initial design chosen. Future schema design would be probably given more testing and analysis time.
Step 3: Compiling and linking flat file builder

These files were located in the [.LOAD] subdirectory.

There were two variants for this: the original and the binary version. The RMU load utility could read a binary or ASCII file. If ASCII was used, RMU would have to convert it to binary, so the more efficient method was to use the binary loader. Using a binary input stream also resulted in a 25% space savings (55 bytes vs. 72 bytes input record length) and reduced use of CPU resources. These savings were needed in order to fit 3 months data per temp device. In all, 8 temp devices were used to represent 2 years worth of data (24 months).

The ASCII and binary flat file builder programs were

FACT_FILE_BLD.C (ASCII)
FACT_FILE_BIN.C (binary)

The following commands compiled and linked either of these:

$ CC file_name
$ LINK file_name

Step 4: Generating the input streams

These files were located in the [.LOAD] subdirectory.

Just as with the flat file builders, there were two versions of the load programs. Either loader program was executed and used the FACT_FILE_Bxx.EXE generated in step 3. The loader programs were:

$ @LOADER (ASCII loader)
$ @BINARY (Binary loader)

Either LOADER.COM or BINARY.COM would be used, but not both. BINARY.COM is recommended since it is an improved (faster) version of LOADER.COM and uses less storage space.

These command files called the appropriate FACT_FILE_Bxx file. If BINARY.COM was executed with no parameters, 24 load files (months) would be created by default.

The input to the above command files was the DAILYnn.DAT files that were provided with the CPG database. The above .COM files read the DAILYxx.DAT files (where xx = the month number) and created a DAILYxx.UNL file. The .DAT files were nearly identical to the .UNL files except that the date was in a different format than the original (Oracle7) format. The date format for the RMU Loader enforced a 4-digit year as in ‘YYYYMMDD’ format vs. ‘DD-MMM-YY’ format used by the Oracle7 SQL loader input files.

Step 5: Loading the database

These files were located in the [.LOAD] subdirectory.

The small dimension tables were loaded by executing:

$ @LOAD-DIM

There was an older, less efficient file called INSERT.SQL that performed the same step. It was basically made obsolete by LOAD-DIM.COM.
There were two variants of command files that were used to load the sales facts table depending on whether binary or ASCII load files were created. These were:

\$ @RMUBLD.COM (ASCII)
\$ @RMUBIN.COM (binary)

Again, execute only one of these, not both. The recommended file is RMUBIN.COM. There are multiple variants of the RMUBIN.COM. These variants end in 4, 8, 17, and 41, and represent the number of parallel loader processes to use to load the flat files. Through experimentation, the RMUBIN17.COM file was found to be the most efficient.

**Step 6: Creating the indexes and journals**

Creating the indexes was done as follows:

\$ SQL @INDEXES-95.SQL (For 1995 database on 8400)
\$ SQL @INDEXES-94.SQL (For 1994 database on 4100)

Creating the journals was done as follows:

\$ SQL @JOURNALS-95.SQL (For 1995 database on 8400)
\$ SQL @JOURNALS-94.SQL (For 1994 database on 4100)

**Step 7: Creating the query catalog**

The query catalog was used to direct the queries to the appropriate database. It used the Oracle Database Integration software to do this. It was configured by executing the following:

\$ SQL @DBI.SQL

**Step 8: Executing the queries**

These files were located in the [.QUERIES] subdirectory.

Queries could be executed on either the 1994 database (AlphaServer 4100) or the 1995 database (AlphaServer 8400). They could be executed as an individual query (1-5) or as a single user load (parallel execution of queries 1-5). They could be executed using the following:

\$ SQL @QUERY-9x-n.SQL (where x = year and n = query number)
\$ @QUERIES.COM (parallel execution of queries 1-5)

or

\$ @QUERY_N_USER n 9x (where n is the number of users to run)
This chapter describes the interoperability testing conducted on the test cluster that was set up and installed as described in Chapter 3. The test cluster consisted of an AlphaServer 8400 and AlphaServer 4100 (described in Chapter 2) connected to an existing CI-based VAXcluster. Since customer databases vary widely in size and purpose, a test database was used, the Consumer Packaged Goods (CPG) Database Demo scripts from Oracle Corporation.

This chapter includes the following topics:

• Test configuration overview
• Test scenarios
• Test database
• Test scripts
• Test results
Test Configuration Overview

The following diagram provides an overview of the cluster as tested.

Figure 4-1: Diagram of the Test Cluster
Major Components

The test cluster included the following major components:

- AlphaServer 8400 base system with 4 GB of memory
- AlphaServer 4100 base system with 4 GB of memory
- Two VAX 6620 systems with 128 MB of memory each
- Nine HSJ40 Controller subsystems
- All interconnected by a CI bus

Storage Subsystem

The storage subsystem consisted of the following components:

<table>
<thead>
<tr>
<th>Component</th>
<th>Part Number</th>
<th>Qty</th>
</tr>
</thead>
<tbody>
<tr>
<td>Star Coupler, 8 node with cabinet</td>
<td>SC008-AC</td>
<td>1</td>
</tr>
<tr>
<td>Star Coupler, 8 node add on</td>
<td>SC008-AD</td>
<td>1</td>
</tr>
<tr>
<td>Basic controller shelf for HSJ40 Controller</td>
<td>BA350-MB</td>
<td>9</td>
</tr>
<tr>
<td>StorageWorks Array Controller, 6 channel CI, 36-42 devices, 32 MB read cache</td>
<td>HSJ40-CF</td>
<td>9</td>
</tr>
<tr>
<td>4.3 GB fixed wide disk</td>
<td>RZ29B-VW</td>
<td>308</td>
</tr>
<tr>
<td>Shelf expansion unit</td>
<td>BA356-JC</td>
<td>54</td>
</tr>
<tr>
<td>SW800 Data Center Cabinet</td>
<td>SW800-FA</td>
<td>3</td>
</tr>
<tr>
<td>8-bit I/O module for BA356 shelf</td>
<td>BA35X-MG</td>
<td>54</td>
</tr>
<tr>
<td>SCSI-2 hi-density “a” cable 1.5 m</td>
<td>BN21H-1E</td>
<td>54</td>
</tr>
<tr>
<td>CI bus cable 20 m</td>
<td>BNCIA-20</td>
<td>9</td>
</tr>
</tbody>
</table>

Test Scenarios

The following items are the test scenarios that were performed on the test system, along with a brief purpose for each test.

1. Single-user/system parallel tests (cold and warm caches):

   Test: Time the parallel execution of queries 1-5 on each system.
   Purpose: Highlight the difference in elapsed time between the VAX systems and AlphaServer systems.
   Comments: The database had to be redirected to the system under test and the VLM cache had to be disabled to run on the VAX system. The test was performed twice (cold cache and warm cache).

2. Multi-user single system parallel tests:

   Test: Simultaneously execute queries 1-5 in parallel X number of times to simulate X number of users, where X = 5 and 10.
   Purpose: To isolate any bottlenecks.
   Comments: The database had to be redirected to the system under test and the VLM cache had to be disabled to run on the VAX system.
Interoperability Tests and Results

Test Database

The database was built using the Consumer Packaged Goods (CPG) Database Demo scripts provided by Oracle Corporation. The CPG Database represents typical marketing and sales data for a consumer products manufacturing firm. The data includes two years of sales data in a data warehouse, star schema, optimized for decision support.

The database, constructed in Oracle Rdb7, consists of six tables:

- The SALES_FACTS table, which consists of the bulk of the database
- Five dimension tables: CHANNEL, DAILY_PERIOD, MONTHLY_PERIOD, MARKET, and PRODUCT.

The CPG database consisted of five table spaces:

<table>
<thead>
<tr>
<th>Table Space</th>
<th>Contents</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>FACTS</td>
<td>SALES_FACTS Table&lt;br&gt;41 data files (varying sizes)</td>
<td>147.2 MB&lt;br&gt;2,799,497,440 rows</td>
</tr>
<tr>
<td>FACTSINDEX</td>
<td>SF_KEY Index on SALES_FACTS Table&lt;br&gt;522 data files</td>
<td>147.2 MB</td>
</tr>
<tr>
<td>DIMENSION</td>
<td>1 Data File&lt;br&gt;Dimension Tables: CHANNEL&lt;br&gt;DAILY_PERIOD&lt;br&gt;MONTHLY_PERIOD&lt;br&gt;MARKET&lt;br&gt;PRODUCT</td>
<td>316 KB&lt;br&gt;41 rows&lt;br&gt;2189 rows&lt;br&gt;72 rows&lt;br&gt;1002 rows&lt;br&gt;522 rows</td>
</tr>
<tr>
<td>DIMINDEX</td>
<td>1 Data File&lt;br&gt;Index for Dimension Tables</td>
<td>138 KB</td>
</tr>
<tr>
<td>TEMPFILE</td>
<td>24 data files (5.5 MB each)&lt;br&gt;required for SF_KEY index build</td>
<td>137.6 MB</td>
</tr>
</tbody>
</table>

Originally, storage areas were created assuming a uniform distribution of data by channel. The initial database design was to partition by channel for data storage, but to partition indexes by product. This benefited the time, channel, and by product queries. By time queries were supported by yearly partitions handled by the Parallel Query Option.

Initial database loading, however, showed that across the 41 channels, data distribution was not uniform. There were 8 instances each of 5 channel groups (Discount, Drug, Warehouse, Supermarket, Convenience) and 1 “Total” channel. The following table summarizes this distribution and the sizes of per channel datasets in terms of row count, storage page count and MB size.
### Table 4-2: Data Distribution Across Channels

<table>
<thead>
<tr>
<th>Group</th>
<th>Records/Mo.</th>
<th>Pages/Mo.</th>
<th>MB/Mo.</th>
<th>Pages/Yr.</th>
<th>MB/Yr.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Warehouse</td>
<td>433,690</td>
<td>1,456</td>
<td>23</td>
<td>17,470</td>
<td>273</td>
</tr>
<tr>
<td>Discount</td>
<td>1,794,528</td>
<td>6,024</td>
<td>94</td>
<td>72,289</td>
<td>1,130</td>
</tr>
<tr>
<td>Drug</td>
<td>1,794,528</td>
<td>6,024</td>
<td>94</td>
<td>72,289</td>
<td>1,130</td>
</tr>
<tr>
<td>Convenience</td>
<td>2,661,908</td>
<td>8,936</td>
<td>140</td>
<td>107,230</td>
<td>1,675</td>
</tr>
<tr>
<td>Supermarket</td>
<td>7,951,376</td>
<td>26,692</td>
<td>417</td>
<td>320,307</td>
<td>5,005</td>
</tr>
<tr>
<td><strong>“Total”</strong></td>
<td>1,794,528</td>
<td>6,024</td>
<td>94</td>
<td>72,289</td>
<td>1,130</td>
</tr>
<tr>
<td>Total by group (41)</td>
<td>118,882,768</td>
<td>399,082</td>
<td>6,236</td>
<td>4,788,979</td>
<td>74,828</td>
</tr>
<tr>
<td>assume 31 days</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data by year</td>
<td>1,426,593,216</td>
<td>4,788,979</td>
<td>74,828</td>
<td></td>
<td></td>
</tr>
<tr>
<td>assume 31 days</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Index by year</td>
<td>1,426,593,216</td>
<td>4,788,979</td>
<td>74,828</td>
<td></td>
<td></td>
</tr>
<tr>
<td>assume 31 days</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2 year small DBMS</td>
<td>2,853,186,432</td>
<td>9,577,957</td>
<td>299,311</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6 year large DBMS</td>
<td>8,559,559,296</td>
<td>28,733,872</td>
<td>897,934</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Database Queries

The query scripts were run in 3 different buffer configurations:

- With global buffers only (20,000 total, each user having 500 buffers, on AlphaServer systems)
- With global buffers and row cache buffers (on AlphaServer systems)
- With global buffers only (1000 total, each user having 100 buffers, on VAX systems)
The row cache buffers were designed to hold all small tables and all referenced data in the larger tables and indexes as follows:

**Table 4-3: Allocation of Row Cache Buffers**

<table>
<thead>
<tr>
<th>Name</th>
<th>Record Length</th>
<th>Cache</th>
<th>Rows</th>
<th>Used</th>
<th>Bytes Allocated</th>
<th>Bytes Used</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSTEM_CACHE</td>
<td>1,000</td>
<td>500</td>
<td>484</td>
<td>500</td>
<td>500,000</td>
<td>484,000</td>
</tr>
<tr>
<td>CHANNEL*</td>
<td>226</td>
<td>41</td>
<td>40</td>
<td>9,266</td>
<td>9,040</td>
<td></td>
</tr>
<tr>
<td>C_CHANNEL</td>
<td>960</td>
<td>10</td>
<td>2</td>
<td>9,600</td>
<td>1,920</td>
<td></td>
</tr>
<tr>
<td>C_CHANNEL_ID</td>
<td>960</td>
<td>10</td>
<td>0</td>
<td>9,600</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>C_CHNLGRP</td>
<td>215</td>
<td>10</td>
<td>3</td>
<td>2,160</td>
<td>648</td>
<td></td>
</tr>
<tr>
<td>DAILY_PERIOD*</td>
<td>70</td>
<td>2,189</td>
<td>2,189</td>
<td>153,230</td>
<td>153,230</td>
<td></td>
</tr>
<tr>
<td>D_DAY</td>
<td>960</td>
<td>100</td>
<td>0</td>
<td>96,000</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>D_MONTH</td>
<td>215</td>
<td>100</td>
<td>17</td>
<td>21,600</td>
<td>3,672</td>
<td></td>
</tr>
<tr>
<td>D_QUARTER</td>
<td>215</td>
<td>100</td>
<td>0</td>
<td>21,600</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>MARKET*</td>
<td>161</td>
<td>1,002</td>
<td>140</td>
<td>164,328</td>
<td>22,960</td>
<td></td>
</tr>
<tr>
<td>MONTHLY_PERIOD*</td>
<td>52</td>
<td>72</td>
<td>0</td>
<td>3,744</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>M_DISTRICT</td>
<td>215</td>
<td>100</td>
<td>18</td>
<td>21,600</td>
<td>3,888</td>
<td></td>
</tr>
<tr>
<td>M_MARKET</td>
<td>960</td>
<td>100</td>
<td>2</td>
<td>96,000</td>
<td>1,920</td>
<td></td>
</tr>
<tr>
<td>M_MARKET_ID</td>
<td>960</td>
<td>100</td>
<td>0</td>
<td>96,000</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>M_REGION</td>
<td>215</td>
<td>100</td>
<td>2</td>
<td>21,600</td>
<td>432</td>
<td></td>
</tr>
<tr>
<td>PRODUCT</td>
<td>332</td>
<td>522</td>
<td>21</td>
<td>173,304</td>
<td>6,972</td>
<td></td>
</tr>
<tr>
<td>P_BRAND</td>
<td>215</td>
<td>100</td>
<td>0</td>
<td>21,600</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>P_MFR</td>
<td>215</td>
<td>100</td>
<td>0</td>
<td>21,600</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>P_PRODUCT</td>
<td>960</td>
<td>100</td>
<td>3</td>
<td>96,000</td>
<td>2,880</td>
<td></td>
</tr>
<tr>
<td>P_PRODUCT_ID</td>
<td>960</td>
<td>100</td>
<td>0</td>
<td>96,000</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>SALES_FACTS*</td>
<td>55</td>
<td>10,000</td>
<td>2,394,477</td>
<td>560,000,000</td>
<td>134,090,712</td>
<td></td>
</tr>
<tr>
<td>SF_KEY</td>
<td>960</td>
<td>1,000,000</td>
<td>38,512</td>
<td>960,000,000</td>
<td>36,971,520</td>
<td></td>
</tr>
<tr>
<td><strong>Totals</strong></td>
<td><strong>1,521,634,832</strong></td>
<td><strong>171,753,794</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

* Tables

**Test Scripts**

Functional verification of the ability to perform query operations was demonstrated using five SQL join scripts. These queries exercised functionality of the RDB7 server, SQL Utility, and Parallel Query components.

The scripts were designed to emulate typical decision support questions about the historical activity of a product sales environment. In most cases the results of these types of queries were used to generate sales trends.

Typical of the kind of question asked was:

“What percentage of product share in dollars and units did a particular product have in a particular area as compared to competitive products in the same area?”

The queries were designed to search the database in varying ways to exercise the database.
Characterization of the Queries

The test process consisted of five queries. Each query was supported by two scripts, one for the 1994 database (for example, QUERY-94-1.SQL) and one for the 1995 database (for example, QUERY-95-1.SQL).

Query 1

Query 1 asked “What was the product share of a specific brand of cereal as compared to other cereals in the same product category, in a particular state in a particular type of store.”

The information was grouped by month to show market trends.

The business question asked was:

“How did 20 oz. Wheat Flakes do as compared to all types of wheat flakes in supermarkets in the state of Connecticut?”

Script for Query 1 (QUERY-9x-1.SQL)

```sql
set dialect 'oracle level1';
/* 1. Star -- Product Share of Brand */

SELECT 'All Wheat Flakes' Product, AL2.MONTH,
      sum(AL5.UNIT_SALES) UNITS, sum(AL5.DOLLAR_SALES) DOLLARS, count(*),
      DISTRICT, CHANNEL_GROUP CHNL
FROM PRODUCT AL4,
     SALES_FACT AL5,
     CHANNEL AL1,
     DAILY_PERIOD AL2,
     MARKET AL3
WHERE (AL5.PRODUCT_ID=AL4.PRODUCT_ID
      AND AL5.MARKET_ID=AL3.MARKET_ID
      AND AL5.CHANNEL_ID=AL1.CHANNEL_ID
      AND AL5.DAY=AL2.DAY)
      AND (district='Connecticut'
           AND CHANNEL_GROUP in('Supermarket')
           AND BRAND in ('Quellogs Wheat Flakes')
           AND YEAR=x)           (where x = year, 1994 or 1995)
GROUP BY DISTRICT, CHANNEL_GROUP, Product, AL2.MONTH

UNION

SELECT '20 Oz Wheat Flakes' Product, AL2.MONTH,
      sum(AL5.UNIT_SALES) UNITS, sum(AL5.DOLLAR_SALES) DOLLARS, count(*),
      DISTRICT, CHANNEL_GROUP CHNL
FROM PRODUCT AL4,
     SALES_FACT AL5,
     CHANNEL AL1,
     DAILY_PERIOD AL2,
     MARKET AL3
WHERE (AL5.PRODUCT_ID=AL4.PRODUCT_ID
      AND AL5.MARKET_ID=AL3.MARKET_ID
      AND AL5.CHANNEL_ID=AL1.CHANNEL_ID
      AND AL5.DAY=AL2.DAY)
      AND (district='Connecticut'
           AND CHANNEL_GROUP in('Supermarket')
           AND PRODUCT='QLGS WHT FLK 20 OZ'
           AND YEAR=x)           (where x = year, 1994 or 1995)
GROUP BY DISTRICT, CHANNEL_GROUP, product, AL2.MONTH;
```
Query 2

Query 2 compared the sales of a specific product, in a particular outlet in a region, against the sales of the same product through all channel outlets. The information was grouped by month to show market trends.

The business question asked was:

“What percentage of sales of 15 oz. Wheat Flakes were made in the Safeway stores in NY and PA as compared to all outlets in the NY and PA areas?”

Script for Query 2 (QUERY-9x-2.SQL)

```sql
set dialect 'oracle level1';
/*  2. Star -- Channel share of all channels */
SELECT 'All Channels' CHNL, AL2.MONTH,
    sum(AL5.UNIT_SALES) Units, sum(AL5.DOLLAR_SALES) Dollars, count(*),
    'NY + PA' DISTRICT, PRODUCT
FROM PRODUCT AL4,
     SALES_FACT AL5,
     CHANNEL AL1,
     DAILY_PERIOD AL2,
     MARKET AL3
WHERE (AL5.PRODUCT_ID=AL4.PRODUCT_ID
    AND AL5.MARKET_ID=AL3.MARKET_ID
    AND AL5.CHANNEL_ID=AL1.CHANNEL_ID
    AND AL5.DAY=AL2.DAY)
    AND DISTRICT in ('New York', 'Pennsylvania')
    AND CHANNEL_GROUP in ('Supermarket', 'Convenience',
    'Warehouse', 'Drug', 'Discount')
    AND PRODUCT= 'QLGS WHT FLK 15 OZ'
    AND YEAR=x
    (where x = year, 1994 or 1995)
GROUP
    BY CHANNEL, PRODUCT, AL2.MONTH
UNION

SELECT CHANNEL CHNL, AL2.MONTH,
    sum(AL5.UNIT_SALES) Units, sum(AL5.DOLLAR_SALES) Dollars, count(*),
    'NY + PA' DISTRICT, PRODUCT
FROM PRODUCT AL4,
     SALES_FACT AL5,
     CHANNEL AL1,
     DAILY_PERIOD AL2,
     MARKET AL3
WHERE (AL5.PRODUCT_ID=AL4.PRODUCT_ID
    AND AL5.MARKET_ID=AL3.MARKET_ID
    AND AL5.CHANNEL_ID=AL1.CHANNEL_ID
    AND AL5.DAY=AL2.DAY)
    AND DISTRICT in ('New York', 'Pennsylvania')
    AND CHANNEL='Safeway'
    AND PRODUCT= 'QLGS WHT FLK 15 OZ'
    AND YEAR=x
    (where x = year, 1994 or 1995)
GROUP
    BY CHANNEL, PRODUCT, AL2.MONTH;
```
Query 3

Query 3 compared the market share of a product in a particular type of store, in a particular market location, to sales of all types of outlets in the region. The information was grouped by month to show market trends.

The business question asked was:

“How are 10 oz. Wheat Flakes doing in convenience stores in Bridgeport Connecticut as compared to the entire northeast region?”

Script for Query 3 (QUERY-9x-3.SQL)

```sql
set dialect 'oracle level1';

/* 3. Star Market share of Region */

SELECT 'Northeast Total' MARKET, AL2.MONTH,
       sum(AL5.UNIT_SALES) Units, sum(AL5.DOLLAR_SALES) Dollars, count(*),
       CHANNEL_GROUP, PRODUCT
FROM PRODUCT AL4,
     SALES_FACT AL5,
     CHANNEL AL1,
     DAILY_PERIOD AL2,
     MARKET AL3
WHERE (AL5.PRODUCT_ID=AL4.PRODUCT_ID
       AND AL5.MARKET_ID=AL3.MARKET_ID
       AND AL5.CHANNEL_ID=AL1.CHANNEL_ID
       AND AL5.DAY=AL2.DAY)
       AND (REGION='Northeast'
           AND CHANNEL_GROUP in ('Convenience')
           AND PRODUCT='QLGS WHT FLK 10 OZ'
           AND YEAR=x)       (where x = year, 1994 or 1995)
GROUP
   BY MARKET, CHANNEL_GROUP, PRODUCT, AL2.MONTH

UNION

SELECT MARKET, AL2.MONTH,
       sum(AL5.UNIT_SALES) Units, sum(AL5.DOLLAR_SALES) Dollars, count(*),
       CHANNEL_GROUP, PRODUCT
FROM PRODUCT AL4, SALES_FACT AL5,
     CHANNEL AL1, DAILY_PERIOD AL2, MARKET AL3
WHERE (AL5.PRODUCT_ID=AL4.PRODUCT_ID
       AND AL5.MARKET_ID=AL3.MARKET_ID
       AND AL5.CHANNEL_ID=AL1.CHANNEL_ID
       AND AL5.DAY=AL2.DAY)
       AND (MARKET='Bridgeport'
           AND CHANNEL_GROUP in ('Convenience')
           AND PRODUCT='QLGS WHT FLK 10 OZ'
           AND YEAR=x)       (where x = year, 1994 or 1995)
GROUP
   BY MARKET, CHANNEL_GROUP, PRODUCT, AL2.MONTH;
```
Query 4

Query 4 compared the market share of a particular product, in a particular type of store, in a particular market location to all sales of competitive products in the same market location. The information was grouped by month to show market trends.

The business question asked was:

“What was the market share of 20 oz. Wheat Flakes in Connecticut supermarkets?”

Script for Query 4 (QUERY-9x-4.SQL)

```sql
set dialect 'oracle level1';
/*  4. Star -- Product share of SubCategory -all competitive prods */
SELECT 'All Wheat Products' Product, AL2.MONTH,
        sum(AL5.UNIT_SALES) Units, sum(AL5.DOLLAR_SALES) Dollars, count(*),
        DISTRICT, CHANNEL_GROUP CHNL
FROM PRODUCT AL4,
     SALES_FACT AL5,
     CHANNEL AL1,
     DAILY_PERIOD AL2,
     MARKET AL3
WHERE (AL5.PRODUCT_ID=AL4.PRODUCT_ID
        AND AL5.MARKET_ID=AL3.MARKET_ID
        AND AL5.CHANNEL_ID=AL1.CHANNEL_ID
        AND AL5.DAY=AL2.DAY)
        AND (district='Connecticut' AND CHANNEL_GROUP in('Supermarket')
        AND BRAND IN ('Quellogs Wheat Flakes', 'Boast Weeties', 'Boast Oatey Rounds',
                        'Quellogs Wheaten Rye')
        GROUP BY DISTRICT, CHANNEL_GROUP, Product, AL2.MONTH
        AND YEAR=x) (where x = year, 1994 or 1995)
UNION

SELECT '20 Oz Wheat Flakes' Product, AL2.MONTH,
        sum(AL5.UNIT_SALES) Units, sum(AL5.DOLLAR_SALES) Dollars, count(*),
        DISTRICT, CHANNEL_GROUP CHNL
FROM PRODUCT AL4,
     SALES_FACT AL5,
     CHANNEL AL1,
     DAILY_PERIOD AL2,
     MARKET AL3
WHERE (AL5.PRODUCT_ID=AL4.PRODUCT_ID
        AND AL5.MARKET_ID=AL3.MARKET_ID
        AND AL5.CHANNEL_ID=AL1.CHANNEL_ID
        AND AL5.DAY=AL2.DAY)
        AND (district='Connecticut' AND CHANNEL_GROUP in('Supermarket')
        AND PRODUCT='QLGS WHT FLK 20 OZ'
        AND YEAR=x) (where x = year, 1994 or 1995)
GROUP BY DISTRICT, CHANNEL_GROUP, Product, AL2.MONTH;
```
Query 5

Query 5 compared the product share of a given product, combining several areas, to total sales across the same areas.

The business question asked was:

“What was the market share of 20 oz. Wheat Flakes across 10 test market areas?”

**Script for Query 5 (QUERY-9x-5.SQL)**

```sql
set dialect 'oracle level1';
/* 5. Star -- Product share of brand in 10 test markets aggregated */
/* include the star hint - 1/29/97 JMM */
SELECT 'All Wheat Flakes' Product, AL2.MONTH,
    sum(AL5.UNIT_SALES) Units, sum(AL5.DOLLAR_SALES) Dollars, count(*),
    CHANNEL_GROUP CHNL, '10-States'
FROM PRODUCT AL4,
    SALES_FACT AL5,
    CHANNEL AL1,
    DAILY_PERIOD AL2,
    MARKET AL3
WHERE (AL5.PRODUCT_ID=AL4.PRODUCT_ID
    AND AL5.MARKET_ID=AL3.MARKET_ID
    AND AL5.CHANNEL_ID=AL1.CHANNEL_ID
    AND AL5.DAY=AL2.DAY)
    AND (district in ('Connecticut',
        'Delaware', 'Maine', 'Pennsylvania', 'New York',
        'Oregon', 'Alaska', 'CA North', 'CA South', 'Washington')
    AND CHANNEL_GROUP in ('Supermarket')
    AND BRAND IN ('Quellogs Wheat Flakes')
    AND YEAR=x)  /* (where x = year, 1994 or 1995) */
GROUP BY Product, AL2.MONTH, CHANNEL_GROUP
UNION

SELECT '20 Oz Wheat Flakes' Product, AL2.MONTH,
    sum(AL5.UNIT_SALES) Units, sum(AL5.DOLLAR_SALES) Dollars, count(*),
    CHANNEL_GROUP CHNL, '10-States'
FROM PRODUCT AL4, SALES_FACT AL5,
    CHANNEL AL1, DAILY_PERIOD AL2, MARKET AL3
WHERE (AL5.PRODUCT_ID=AL4.PRODUCT_ID
    AND AL5.MARKET_ID=AL3.MARKET_ID
    AND AL5.CHANNEL_ID=AL1.CHANNEL_ID
    AND AL5.DAY=AL2.DAY)
    AND (district in ('Connecticut',
        'Delaware', 'Maine', 'Pennsylvania', 'New York',
        'Oregon', 'Alaska', 'CA North', 'CA South', 'Washington')
    AND CHANNEL_GROUP in ('Supermarket')
    AND PRODUCT='QLGS WHT FLK 20 OZ'
    AND YEAR=x)  /* (where x = year, 1994 or 1995) */
GROUP BY Product, AL2.MONTH, CHANNEL_GROUP;
```
Interoperability Tests and Results

Test Results

Initial tests on the high-end VLM platform verified the ability to process queries, in which all query data resides on the local system, in a single database. The set of five SQL join scripts, as described above, was executed.

Test One: SQL Queries in Parallel

This test performed the five SQL queries in parallel. This test was run twice, once with cold cache and once with a warm cache.

The following table summarizes the runtime results on the test systems with cold cache conditions.

<table>
<thead>
<tr>
<th></th>
<th>AlphaServer 4100 (hr:min:sec)</th>
<th>AlphaServer 8400 (hr:min:sec)</th>
<th>VAX 6620 (dy hr:min:sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Query 1</td>
<td>22:34.78</td>
<td>25:21.98</td>
<td>0 17:06:26.47</td>
</tr>
<tr>
<td>Query 2</td>
<td>54:18.35</td>
<td>1:04:33.08</td>
<td>0 11:45:53.00</td>
</tr>
<tr>
<td>Query 3</td>
<td>4:12.27</td>
<td>4:30.56</td>
<td>1 04:12:25.36</td>
</tr>
<tr>
<td>Query 4</td>
<td>2:12:05.37</td>
<td>2:12:37.48</td>
<td>3 14:10:57.90</td>
</tr>
<tr>
<td>Query 5</td>
<td>56:23.80</td>
<td>1:04:01.48</td>
<td>3 14:54:11.68</td>
</tr>
</tbody>
</table>

It was readily apparent by the VAX runtimes that it could not compete with the AlphaServer systems due to the AlphaServer VLM capabilities. It was also thought that there would be no significant difference between the cold and warm cache tests on the VAX (due to the small amount of memory available on the VAX), nor was there time during this project to run them, so no warm or multi-user VAX times are shown.

______________________________ Note ___________________________
Query 5 failed on the VAX system due to Virtual Address Space Full error.

______________________________________________________________

The following table summarizes the runtime results on the test systems with warm cache conditions.

<table>
<thead>
<tr>
<th></th>
<th>AlphaServer 4100 (min:sec)</th>
<th>AlphaServer 8400 (min:sec)</th>
<th>VAX 6620 (min:sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Query 1</td>
<td>2:14.60</td>
<td>1:51.39</td>
<td></td>
</tr>
<tr>
<td>Query 2</td>
<td>2:53.75</td>
<td>2:33.83</td>
<td></td>
</tr>
<tr>
<td>Query 3</td>
<td>2:55.49</td>
<td>2:34.44</td>
<td></td>
</tr>
<tr>
<td>Query 4</td>
<td>6:50.20</td>
<td>6:24.68</td>
<td></td>
</tr>
<tr>
<td>Query 5</td>
<td>44:58.33</td>
<td>51:28.96</td>
<td></td>
</tr>
</tbody>
</table>
Test Two: 5- and 10-User Load Tests

This test consisted of 5- and 10-user load tests.

The following table shows the runtime results of the 5- and 10-user load tests run on the AlphaServer 4100 system. The 5-user test was comprised of a total of 25 queries issued simultaneously with 5 users each issuing 5 queries. The 10-user test is comprised of a total of 50 queries issued simultaneously with 10 users each issuing 5 queries.

Table 4-6: AlphaServer 4100 System 5- and 10-User Load Test Results

<table>
<thead>
<tr>
<th>(hr:min:sec)</th>
<th>5 Users</th>
<th>10 Users</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Min</td>
<td>Max</td>
</tr>
<tr>
<td>Query 1</td>
<td>8:27.07</td>
<td>9:42.60</td>
</tr>
<tr>
<td>Query 3</td>
<td>12:14.74</td>
<td>12:41.63</td>
</tr>
<tr>
<td>Query 4</td>
<td>22:16.84</td>
<td>23:09.78</td>
</tr>
<tr>
<td>Query 5</td>
<td>1:24:15.83</td>
<td>1:26:44.10</td>
</tr>
</tbody>
</table>

The following table gives the runtime results of the 5- and 10-user load tests run on the AlphaServer 8400 system. The 5-user test was comprised of a total of 25 queries issued simultaneously with 5 users each issuing 5 queries. The 10-user test was comprised of a total of 50 queries issued simultaneously with 10 users each issuing 5 queries.

Table 4-7: AlphaServer 8400 System 5- and 10-User Load Test Results

<table>
<thead>
<tr>
<th>(hr:min:sec)</th>
<th>5 Users</th>
<th>10 Users</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Min</td>
<td>Max</td>
</tr>
<tr>
<td>Query 1</td>
<td>4:27.57</td>
<td>4:51.04</td>
</tr>
<tr>
<td>Query 2</td>
<td>7:07.49</td>
<td>7:09.63</td>
</tr>
<tr>
<td>Query 3</td>
<td>7:15.78</td>
<td>7:19.69</td>
</tr>
<tr>
<td>Query 4</td>
<td>12:12.50</td>
<td>12:29.01</td>
</tr>
<tr>
<td>Query 5</td>
<td>5:33.42</td>
<td>6:04.91</td>
</tr>
</tbody>
</table>
System Limits and Performance Data

The tests demonstrated the interoperability of the AlphaServer systems in a VAXcluster.

In the tests of SQL queries in parallel, the cold cache test times for the AlphaServer systems ranged from about 2 min. to 2 hr. 12 min. while the VAX systems ranged from over 11 hr. to over 3 days.

In the 5- and 10-user load tests, the test times for the AlphaServer systems ranged from 4 min. 27 sec. to 3 hr. 50 min. The load tests were not run on the VAX systems because comparably long runtimes were expected.

When comparing VAX and AlphaServer architectures, we performed a simple query on a VAX node, only to realize that the comparison was so overwhelmingly in favor of the AlphaServer nodes that any further testing was considered unnecessary for comparison purposes. (See Table 4-4 for actual data.)

The test configuration was not optimized for performance. A tuning effort would have obtained better runtime results. The 64-bit and VLM capabilities of the AlphaServer systems were clearly evident with the runtimes that were observed.

In conducting the tests on the VAX and AlphaServer systems, certain architectural differences were uncovered, the primary one being the difference in memory addressing between a 32-bit system (VAX system) and a 64-bit system (AlphaServer system).

In designing the database buffering schemes for both platforms, these differences had to be taken into account. The Oracle Rdb7 database provides several different buffer management styles to support 32- and 64-bit systems.

These buffer styles are:

- Conventional process sharing local buffers (available on VAX and AlphaServer systems). This scheme is limited to a process’ 32-bit address space.
- Conventional process sharing global buffers (available on VAX and AlphaServer systems). This scheme is limited to a process’ 32-bit address space (1 GB).
- System sharing global buffers (available on AlphaServer systems only). This scheme is limited to a system’s 32-bit address space (1.8 GB).
- System sharing row cache (available on AlphaServer systems only). This scheme is limited to a system’s 32-bit address space (1.8 GB).
- Large memory system sharing row cache (for example, VLM, available on AlphaServer systems only). This scheme is limited to a system’s 64-bit address space or amount of physical memory available on the system.
The first three buffering styles are additive and must all reside within the limits of conventional 32-bit addressing. The latter two options are only available on 64-bit AlphaServer systems and account for a vast performance difference between the platforms.
Problems and Resolutions

The following problems were identified during testing. They were categorized according to the aspect of the project in which the problems were encountered: Oracle Rdb7 database, system/storage hardware, and OpenVMS operating system.

Problems of various kinds were encountered during this systems integration effort. Some of them could cause a significant disruption if encountered during an installation in a customer environment. They are documented here for the purpose of disseminating valuable information uncovered during testing.

Oracle Rdb7 Database

The following problems were related to setup and use of the Oracle database.

1. **Problem**
   Adding one or more AlphaServer systems into a VAXcluster may degrade database performance due to flooding the lock manager with lock requests.

   **Resolution**
   Ensure that the AlphaServer systems own the locks by setting SYSGEN parameter LOCKDIRWT higher for the AlphaServer systems than the VAX systems.

2. **Problem**
   Excessive database creation time (24 hours). Observed a high split transfer rate during database creation.

   **Resolution**
   Initialize the drives with /NOHIGHWATER to prevent Oracle Rdb7 from doing a time-intensive security erase. As a result, the database was created in about 2 hours.

3. **Problem**
   Partitioning the database by channel results in uneven data distribution.

   **Resolution**
   Identify and distribute “hot channels” across stripesets on different HSJ40 controllers based on a month’s worth of data.

4. **Problem**
   The one-user cold cache test aborted on the AlphaServer 8400 system due to insufficient System Page Table Entries (SPTEs).

   **Resolution**
   Reduce the size of the buffer cache from 50 KB buffers down to 20 KB.
DIGITAL System/Storage Hardware

The following problems were related to the AlphaServer hardware.

1. Problem  
Satuated a single CI while trying to load the database (11.4 MB/sec @ 220 I/O/sec). Maximum CI speed is 9.8 MB/sec @ 400 I/Os/sec.

Resolution  
Add a second CI to each system (CIPCA for AlphaServer systems, CIXCD for VAX systems).

2. Problem  
Experienced various CLUEXIT bugchecks on the AlphaServer 8400 system during the database load. One of the CIPCs on the AlphaServer 8400 system exceeded its retry count and brought the port offline.

Resolution  
The system was configured such that process quotas were at maximum on the SYSTEM and RDB accounts. Under heavy load, this resulted in processes running with these quotas to have little to no limits on system resource consumption. An LED was found to be on for one of the memory modules. The module was reseated. The 4x4 8400 was replaced with another 12x4 8400 during all of this, so it was undetermined as to which specific action resolved the problem, but the problem did not reoccur after the 8400 swap.

3. Problem  
Errors (file inconsistency) were reported while accessing the database from the AlphaServer 4100 system.

Resolution  
This appears to be a known problem with some AlphaServer 4100 systems. Server Engineering is already investigating the problem. The original AlphaServer 4100 system was swapped with another machine.

4. Problem  
Media Robot Utility would not see TL812 robot after installation and set up.

Resolution  
A typographical error in the Media Robot Utility Version 1.1 Guide to Installation, Connectivity, and Operation (AA-QTTGB-TE) calls for the robot to be configured as LUN 1 off bus 0 when it is really SCSI ID 0 on bus 0. The TL812 was removed from the configuration due to hardware and software support issues with OpenVMS V7.1.

5. Problem  
Database went offline and became corrupted.

Resolution  
Replaced a defective RZ29B disk in stripeset $1$DUA74 and rebuilt database.
OpenVMS Operating System

The following problem was noted during the course of testing with OpenVMS V7.1.

<table>
<thead>
<tr>
<th>Problem</th>
<th>One-user load test failed on the VAX system (DEPOTX).</th>
</tr>
</thead>
<tbody>
<tr>
<td>Resolution</td>
<td>By default, Oracle Rdb7 uses the SYSTEM account quotas to start the Rdb7 monitor process. In this test, the process consumed nearly 200 KB of page file quota. SYSTEM quota PGFLQUOTA was increased from 110 KB to 300 KB and the page file was increased from 300 KB to 2 GB.</td>
</tr>
</tbody>
</table>
This appendix describes the minimum and maximum hardware configuration for the following:

- Graphic overview (Golden Egg-like)
- AlphaServer 8400 slot usage
- AlphaServer 4100 slot usage
- Database storage setup
- Database storage maps
AlphaServer 8400 Slot Usage

The following tables describe system bus and I/O shelf usage.

Table 6-1: AlphaServer 8400 System Bus Usage

<table>
<thead>
<tr>
<th>Location</th>
<th>Part Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Slot 0</td>
<td>756P1-AX</td>
<td>CPU module (dual CPU)</td>
</tr>
<tr>
<td>Slot 1</td>
<td>756P1-AX</td>
<td></td>
</tr>
<tr>
<td>Slot 2</td>
<td>756P1-AX</td>
<td></td>
</tr>
<tr>
<td>Slot 3</td>
<td>756P1-AX</td>
<td></td>
</tr>
<tr>
<td>Slot 4</td>
<td>756P1-AX</td>
<td></td>
</tr>
<tr>
<td>Slot 5</td>
<td>756P1-AX</td>
<td></td>
</tr>
<tr>
<td>Slot 6</td>
<td>MS7CC-FA</td>
<td>2 GB memory module</td>
</tr>
<tr>
<td>Slot 7</td>
<td>MS7CC-FA</td>
<td></td>
</tr>
<tr>
<td>Slot 8</td>
<td>KFTHA</td>
<td>I/O module (4 ports)</td>
</tr>
</tbody>
</table>

The system used for testing contained three PCI backplanes connected to three I/O ports or hoses.

Table 6-2: AlphaServer 8400 I/O Shelf Usage

<table>
<thead>
<tr>
<th>PCI Number</th>
<th>Connections</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>CIPCA, KZPSA</td>
</tr>
<tr>
<td>2</td>
<td>CIPCA</td>
</tr>
<tr>
<td>3</td>
<td>DE500</td>
</tr>
</tbody>
</table>
AlphaServer 4100 Slot Usage

The following illustration and table identify the components plugged into the PCI motherboard.

![Diagram of AlphaServer 4100 PCI motherboard usage]

<table>
<thead>
<tr>
<th>Slot</th>
<th>Part Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCI1-5</td>
<td>DE500-XA</td>
<td>PCI 10/100-Mbit fast Ethernet controller</td>
</tr>
<tr>
<td>PCI1-4</td>
<td>open</td>
<td></td>
</tr>
<tr>
<td>PCI1-3</td>
<td>KZPDA-AA</td>
<td>Fast Wide Single-ended SCSI controller</td>
</tr>
<tr>
<td>PCI1-2</td>
<td>CIPCA-AA</td>
<td>Port - PCI to CI host bus adapter</td>
</tr>
<tr>
<td>PCI0-5</td>
<td>CIPCA-AA</td>
<td>Port</td>
</tr>
<tr>
<td>EISA-3/PCI0-4</td>
<td>CIPCA-AA</td>
<td>Link</td>
</tr>
<tr>
<td>EISA-2/PCI0-3</td>
<td>CIPCA-AA</td>
<td>Link</td>
</tr>
<tr>
<td>EISA-1/PCI0-2</td>
<td>open</td>
<td></td>
</tr>
</tbody>
</table>
The following illustration and table identify the components plugged into the system motherboard. The configuration is similar for the AlphaServer 4000 system.

### Table 6-4: AlphaServer 4100 System Motherboard Usage

<table>
<thead>
<tr>
<th>Slot</th>
<th>Part Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU3</td>
<td>KN304-BB</td>
<td>400 MHz CPU</td>
</tr>
<tr>
<td>Mem1H</td>
<td>MS330-FA</td>
<td>1 GB memory option</td>
</tr>
<tr>
<td>CPU2</td>
<td>KN304-BB</td>
<td>400 MHz CPU</td>
</tr>
<tr>
<td>Mem1L</td>
<td>MS330-FA</td>
<td>1 GB memory option</td>
</tr>
<tr>
<td>Mem3L</td>
<td>MS330-FA</td>
<td>1 GB memory option</td>
</tr>
<tr>
<td>Mem2L</td>
<td>MS330-FA</td>
<td>1 GB memory option</td>
</tr>
<tr>
<td>CPU1</td>
<td>KN304-BB</td>
<td>400 MHz CPU</td>
</tr>
<tr>
<td>Mem0H</td>
<td>MS330-FA</td>
<td>1 GB memory option</td>
</tr>
<tr>
<td>Mem3H</td>
<td>MS330-FA</td>
<td>1 GB memory option</td>
</tr>
<tr>
<td>Mem2H</td>
<td>MS330-FA</td>
<td>1 GB memory option</td>
</tr>
<tr>
<td>CPU0</td>
<td>KN304-BB</td>
<td>400 MHz CPU</td>
</tr>
<tr>
<td>Mem0L</td>
<td>MS330-FA</td>
<td>1 GB memory option</td>
</tr>
<tr>
<td>IOD01</td>
<td>open</td>
<td></td>
</tr>
</tbody>
</table>
Database Storage Setup

Data was distributed across 9 HSJ controllers located in 3 SW800 cabinets. Each cabinet housed 3 HSJ40 controllers and 18 BA356 shelves such that each controller supported 6 shelves. 308 RZ29B-VW disks were evenly distributed across the six ports of each HSJ40 Controller.

Each disk was given a disk name that consisted of a label, port number, target number, and logical unit number (LUN).

Disk Name = DISK130

<table>
<thead>
<tr>
<th>Logical unit number</th>
</tr>
</thead>
<tbody>
<tr>
<td>Target number</td>
</tr>
<tr>
<td>Port number</td>
</tr>
<tr>
<td>Label</td>
</tr>
</tbody>
</table>

Disks were grouped into stripesets and RAIDsets that spanned several ports of each controller. Each set was assigned a container name such as S1 (stripeset) or R2 (RAIDset) that corresponded to an OpenVMS device name such as $1$DUA12 and an HSJ LUN such as D12.

<table>
<thead>
<tr>
<th>OpenVMS Device Name</th>
<th>LUN</th>
<th>Container Name</th>
<th>Disks</th>
</tr>
</thead>
<tbody>
<tr>
<td>$1$DUA12</td>
<td>D12</td>
<td>S1</td>
<td>DISK110</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>DISK210</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>DISK310</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>DISK410</td>
</tr>
</tbody>
</table>
Database Storage Maps

The following tables map the OpenVMS device names to the physical devices in each stripeset (RAID-0) or RAIDset (RAID-5).

**Table 6-5: Database Storage Map for HSJ001**

<table>
<thead>
<tr>
<th>OpenVMS Device Name</th>
<th>Volume Label</th>
<th>Volume Description</th>
<th>HSJ LUN</th>
<th>HSJ Container Name</th>
<th>HSJ Physical Devices</th>
<th>Chunk Size (blocks)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$1$SDUA12</td>
<td>DISK1D</td>
<td>95 database data disk</td>
<td>D12</td>
<td>S1</td>
<td>DISK110 210 310 410</td>
<td>128</td>
</tr>
<tr>
<td>$1$SDUA11</td>
<td>VAXVMS071</td>
<td>VAX system disk</td>
<td>D11</td>
<td>R1</td>
<td>DISK100 200 300 400 500 600</td>
<td>256</td>
</tr>
<tr>
<td>$1$SDUA13</td>
<td>DISK11</td>
<td>95 database index</td>
<td>D13</td>
<td>S2</td>
<td>DISK120 220 510 610</td>
<td>128</td>
</tr>
<tr>
<td>$1$SDUA14</td>
<td>DISK9D</td>
<td>94 database data disk</td>
<td>D14</td>
<td>S4</td>
<td>DISK130 230 330 430</td>
<td>128</td>
</tr>
<tr>
<td>$1$SDUA15</td>
<td>See note 2</td>
<td></td>
<td>D15</td>
<td>S5</td>
<td>DISK140 240 530 630</td>
<td>128</td>
</tr>
<tr>
<td>$1$SDUA16</td>
<td>DISK9I</td>
<td>95 database index disk</td>
<td>D16</td>
<td>S7</td>
<td>DISK150 250 350 450</td>
<td>128</td>
</tr>
<tr>
<td>$1$SDUA17</td>
<td>See note 2</td>
<td></td>
<td>D17</td>
<td>S3</td>
<td>DISK320 420 520 620</td>
<td>128</td>
</tr>
<tr>
<td>$1$SDUA18</td>
<td>See note 2</td>
<td></td>
<td>D18</td>
<td>S6</td>
<td>DISK340 440 540 640</td>
<td>128</td>
</tr>
</tbody>
</table>

1$ S1, S2, … = stripeset, R1, R2, … = RAIDset.
2$ Used for temp and sort disks during the database build.
Table 6-6: Database Storage Map for HSJ002

<table>
<thead>
<tr>
<th>OpenVMS Device Name</th>
<th>Volume Label</th>
<th>Volume Description</th>
<th>HSJ LUN</th>
<th>HSJ Container Name</th>
<th>HSJ Physical Devices</th>
<th>Chunk Size (blocks)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$1$SDUA21</td>
<td>ALPHASYS071</td>
<td>AlphaServer system disk</td>
<td>D21</td>
<td>R1</td>
<td>DISK100: 200, 300, 400, 500, 600</td>
<td>256</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>DISK110: 210, 310, 410</td>
<td>128</td>
</tr>
<tr>
<td>$1$SDUA22</td>
<td>DISK2D</td>
<td>95 database data disk</td>
<td>D22</td>
<td>S1</td>
<td>DISK120: 220, 310, 410</td>
<td>128</td>
</tr>
<tr>
<td>$1$SDUA23</td>
<td>DISK2I</td>
<td>95 database index disk</td>
<td>D23</td>
<td>S2</td>
<td>DISK130: 230, 330, 430</td>
<td>128</td>
</tr>
<tr>
<td>$1$SDUA24</td>
<td>DISK10D</td>
<td>94 database data disk</td>
<td>D24</td>
<td>S4</td>
<td>DISK140: 240, 530, 630</td>
<td>128</td>
</tr>
<tr>
<td>$1$SDUA25</td>
<td>See note 2</td>
<td></td>
<td>D25</td>
<td>S5</td>
<td>DISK150: 250, 350, 450</td>
<td>128</td>
</tr>
<tr>
<td>$1$SDUA26</td>
<td>DISK10I</td>
<td>94 database index disk</td>
<td>D26</td>
<td>S7</td>
<td>DISK320: 420, 520, 620</td>
<td>128</td>
</tr>
<tr>
<td>$1$SDUA27</td>
<td>See note 2</td>
<td></td>
<td>D27</td>
<td>S3</td>
<td>DISK340: 440, 540, 640</td>
<td>128</td>
</tr>
<tr>
<td>$1$SDUA28</td>
<td>See note 2</td>
<td></td>
<td>D28</td>
<td>S6</td>
<td></td>
<td>128</td>
</tr>
</tbody>
</table>

1S1, S2, … = stripeset, R1, R2, … = RAIDset.
2Used for temp and sort disks during the database build.
### Table 6-7: Database Storage Map for HSJ003

<table>
<thead>
<tr>
<th>OpenVMS Device Name</th>
<th>Volume Label</th>
<th>Volume Description</th>
<th>HSJ LUN</th>
<th>HSJ Container Name</th>
<th>HSJ Physical Devices</th>
<th>Chunk Size (blocks)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$1$DUA31</td>
<td>DISK3D</td>
<td>95 database data</td>
<td>D31</td>
<td>S1</td>
<td>DISK100 200 300 400</td>
<td>128</td>
</tr>
<tr>
<td>$1$DUA32</td>
<td>DISK3I</td>
<td>95 database index</td>
<td>D32</td>
<td>S2</td>
<td>DISK110 210 500 600</td>
<td>128</td>
</tr>
<tr>
<td>$1$DUA33</td>
<td>See note 2</td>
<td></td>
<td>D33</td>
<td>S4</td>
<td>DISK120 220 320 420</td>
<td>128</td>
</tr>
<tr>
<td>$1$DUA34</td>
<td>DISK11D</td>
<td>94 database data disk</td>
<td>D34</td>
<td>S5</td>
<td>DISK130 230 520 620</td>
<td>128</td>
</tr>
<tr>
<td>$1$DUA35</td>
<td>DISK11I</td>
<td>94 database index disk</td>
<td>D35</td>
<td>S7</td>
<td>DISK140 240 340 440</td>
<td>128</td>
</tr>
<tr>
<td>$1$DUA36</td>
<td>See note 2</td>
<td></td>
<td>D36</td>
<td>S6</td>
<td>DISK330 430 530 630</td>
<td>128</td>
</tr>
<tr>
<td>$1$DUA37</td>
<td>See note 2</td>
<td></td>
<td>D37</td>
<td>S3</td>
<td>DISK310 410 510 610</td>
<td>128</td>
</tr>
<tr>
<td>$1$DUA38</td>
<td>See note 2</td>
<td></td>
<td>D38</td>
<td>S8</td>
<td>DISK150 250 540 640</td>
<td>128</td>
</tr>
<tr>
<td>$1$DUA39</td>
<td>See note 2</td>
<td></td>
<td>D39</td>
<td>S9</td>
<td>DISK350 450 550 650</td>
<td>128</td>
</tr>
</tbody>
</table>

1. S1, S2, … = stripeset, R1, R2, … = RAIDset.
2. Used for temp and sort disks during the database build.
Table 6-8: Database Storage Map for HSJ004

<table>
<thead>
<tr>
<th>OpenVMS Device Name</th>
<th>Volume Label</th>
<th>Volume Description</th>
<th>HSJ LUN</th>
<th>HSJ Container Name</th>
<th>HSJ Physical Devices</th>
<th>Chunk Size (blocks)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$1SDUA41</td>
<td>DISK4D</td>
<td>95 database data disk</td>
<td>D41</td>
<td>S1</td>
<td>DISK100 200 300 400</td>
<td>128</td>
</tr>
<tr>
<td>$1SDUA42</td>
<td>DISK4I</td>
<td>95 database index disk</td>
<td>D42</td>
<td>S2</td>
<td>DISK110 210 500 600</td>
<td>128</td>
</tr>
<tr>
<td>$1SDUA43</td>
<td>See note 2</td>
<td></td>
<td>D43</td>
<td>S4</td>
<td>DISK120 220 320 420</td>
<td>128</td>
</tr>
<tr>
<td>$1SDUA44</td>
<td>DISK12D</td>
<td>94 database data disk</td>
<td>D44</td>
<td>S5</td>
<td>DISK130 230 520 620</td>
<td>128</td>
</tr>
<tr>
<td>$1SDUA45</td>
<td>DISK12I</td>
<td>94 database index disk</td>
<td>D45</td>
<td>S7</td>
<td>DISK140 240 340 440</td>
<td>128</td>
</tr>
<tr>
<td>$1SDUA46</td>
<td>See note 2</td>
<td></td>
<td>D46</td>
<td>S8</td>
<td>DISK150 250 540 640</td>
<td>128</td>
</tr>
<tr>
<td>$1SDUA47</td>
<td>See note 2</td>
<td></td>
<td>D47</td>
<td>S3</td>
<td>DISK310 410 510 610</td>
<td>128</td>
</tr>
<tr>
<td>$1SDUA48</td>
<td>See note 2</td>
<td></td>
<td>D48</td>
<td>S6</td>
<td>DISK330 430 530 630</td>
<td>128</td>
</tr>
<tr>
<td>$1SDUA49</td>
<td>See note 2</td>
<td></td>
<td>D49</td>
<td>S9</td>
<td>DISK350 450 550 650</td>
<td>128</td>
</tr>
</tbody>
</table>

1 S1, S2, … = stripset, TEMPDR1, R2, … = RAIDset.
2 Used for temp and sort disks during the database build.
### Table 6-9: Database Storage Map for HSJ005

<table>
<thead>
<tr>
<th>OpenVMS Device Name</th>
<th>Volume Label</th>
<th>Volume Description</th>
<th>HSJ LUN</th>
<th>HSJ Container Name</th>
<th>HSJ Physical Devices</th>
<th>Chunk Size (blocks)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$1$SDUA51</td>
<td>DISK5D</td>
<td>95 database data disk</td>
<td>D51</td>
<td>S1</td>
<td>DISK100 200 300 400</td>
<td>128</td>
</tr>
<tr>
<td>$1$SDUA52</td>
<td>DISK5I</td>
<td>95 database index disk</td>
<td>D52</td>
<td>S2</td>
<td>DISK110 210 500 600</td>
<td>128</td>
</tr>
<tr>
<td>$1$SDUA53</td>
<td>See note 2</td>
<td></td>
<td>D53</td>
<td>S4</td>
<td>DISK120 220 320 420</td>
<td>128</td>
</tr>
<tr>
<td>$1$SDUA54</td>
<td>DISK13D</td>
<td>94 database data disk</td>
<td>D54</td>
<td>S5</td>
<td>DISK130 230 520 620</td>
<td>128</td>
</tr>
<tr>
<td>$1$SDUA55</td>
<td>DISK13I</td>
<td>94 database index disk</td>
<td>D55</td>
<td>S7</td>
<td>DISK140 240 340 440</td>
<td>128</td>
</tr>
<tr>
<td>$1$SDUA56</td>
<td>See note 2</td>
<td></td>
<td>D56</td>
<td>S8</td>
<td>DISK150 250 540 640</td>
<td>128</td>
</tr>
<tr>
<td>$1$SDUA57</td>
<td>See note 2</td>
<td></td>
<td>D57</td>
<td>S3</td>
<td>DISK310 410 510 610</td>
<td>128</td>
</tr>
<tr>
<td>$1$SDUA58</td>
<td>See note 2</td>
<td></td>
<td>D58</td>
<td>S6</td>
<td>DISK330 430 530 630</td>
<td>128</td>
</tr>
<tr>
<td>$1$SDUA59</td>
<td>See note 2</td>
<td></td>
<td>D59</td>
<td>S9</td>
<td>DISK350 450 550 650</td>
<td>128</td>
</tr>
</tbody>
</table>

1. S1, S2, … = stripeset, R1, R2, … = RAIDset.
2. Used for temp and sort disks during the database build.
### Table 6-10: Database Storage Map for HSJ006

<table>
<thead>
<tr>
<th>OpenVMS Device Name</th>
<th>Volume Label</th>
<th>Volume Description</th>
<th>HSJ LUN</th>
<th>HSJ Container Name</th>
<th>HSJ Physical Devices</th>
<th>Chunk Size (blocks)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$1$SDUA61</td>
<td>DISK6D</td>
<td>95 database data disk</td>
<td>D61</td>
<td>S1</td>
<td>DISK100 200 300 400</td>
<td>128</td>
</tr>
<tr>
<td>$1$SDUA62</td>
<td>DISK6I</td>
<td>95 database index disk</td>
<td>D62</td>
<td>S2</td>
<td>DISK110 210 300 600</td>
<td>128</td>
</tr>
<tr>
<td>$1$SDUA63</td>
<td>See note 2</td>
<td></td>
<td>D63</td>
<td>S4</td>
<td>DISK120 220 320 420</td>
<td>128</td>
</tr>
<tr>
<td>$1$SDUA64</td>
<td>DISK14D</td>
<td>94 database data disk</td>
<td>D64</td>
<td>S5</td>
<td>DISK130 230 520 620</td>
<td>128</td>
</tr>
<tr>
<td>$1$SDUA65</td>
<td>DISK14I</td>
<td>94 database index disk</td>
<td>D65</td>
<td>S7</td>
<td>DISK140 240 340 440</td>
<td>128</td>
</tr>
<tr>
<td>$1$SDUA66</td>
<td>See note 2</td>
<td></td>
<td>D66</td>
<td>S8</td>
<td>DISK150 250 540 640</td>
<td>128</td>
</tr>
<tr>
<td>$1$SDUA67</td>
<td>See note 2</td>
<td></td>
<td>D67</td>
<td>S3</td>
<td>DISK310 410 510 610</td>
<td>128</td>
</tr>
<tr>
<td>$1$SDUA68</td>
<td>See note 2</td>
<td></td>
<td>D68</td>
<td>S6</td>
<td>DISK330 430 530 630</td>
<td>128</td>
</tr>
<tr>
<td>$1$SDUA69</td>
<td>See note 2</td>
<td></td>
<td>D69</td>
<td>S9</td>
<td>DISK350 450 550 650</td>
<td>128</td>
</tr>
</tbody>
</table>

1 S1, S2, ... = stripeset, R1, R2, ... = RAIDset.
2 Used for temp and sort disks during the database build.
<table>
<thead>
<tr>
<th>OpenVMS Device Name</th>
<th>Volume Label</th>
<th>Volume Description</th>
<th>HSJ LUN</th>
<th>HSJ Container Name</th>
<th>HSJ Physical Devices</th>
<th>Chunk Size (blocks)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$1$SDUA71</td>
<td>DISK7D</td>
<td>95 database data disk</td>
<td>D71</td>
<td>S1</td>
<td>DISK100 200 300 400</td>
<td>128</td>
</tr>
<tr>
<td>$1$SDUA72</td>
<td>DISK7I</td>
<td>95 database index disk</td>
<td>D72</td>
<td>S2</td>
<td>DISK110 210 500 600</td>
<td>128</td>
</tr>
<tr>
<td>$1$SDUA73</td>
<td>See note 2</td>
<td></td>
<td>D73</td>
<td>S4</td>
<td>DISK120 220 320 420</td>
<td>128</td>
</tr>
<tr>
<td>$1$SDUA74</td>
<td>DISK15D</td>
<td>94 database data disk</td>
<td>D74</td>
<td>S5</td>
<td>DISK130 230 520 620</td>
<td>128</td>
</tr>
<tr>
<td>$1$SDUA75</td>
<td>DISK15I</td>
<td>94 database index disk</td>
<td>D75</td>
<td>S7</td>
<td>DISK140 240 340 440</td>
<td>128</td>
</tr>
<tr>
<td>$1$SDUA76</td>
<td>See note 2</td>
<td></td>
<td>D76</td>
<td>S8</td>
<td>DISK150 250 540 640</td>
<td>128</td>
</tr>
<tr>
<td>$1$SDUA77</td>
<td>See note 2</td>
<td></td>
<td>D77</td>
<td>S3</td>
<td>DISK310 410 510 610</td>
<td>128</td>
</tr>
<tr>
<td>$1$SDUA78</td>
<td>SORT0</td>
<td>95 database sort disk</td>
<td>D78</td>
<td>S6</td>
<td>DISK330 430 530 630</td>
<td>128</td>
</tr>
<tr>
<td>$1$SDUA79</td>
<td>SORTC</td>
<td>94 database sort disk</td>
<td>D79</td>
<td>S9</td>
<td>DISK350 450 550 650</td>
<td>128</td>
</tr>
</tbody>
</table>

1 $S1$, $S2$, … = stripeset, $R1$, $R2$, … = RAIDset.
2 Used for temp and sort disks during the database build.
Table 6-12: Database Storage Map for HSJ008

<table>
<thead>
<tr>
<th>OpenVMS Device Name</th>
<th>Volume Label</th>
<th>Volume Description</th>
<th>HSJ LUN</th>
<th>HSJ Container Name</th>
<th>HSJ Physical Devices</th>
<th>Chunk Size (blocks)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$1$SDUA81</td>
<td>DISK8D</td>
<td>95 database data disk</td>
<td>D81</td>
<td>S1</td>
<td>DISK100 200 300 400</td>
<td>128</td>
</tr>
<tr>
<td>$1$SDUA82</td>
<td>DISK8I</td>
<td>95 database index disk</td>
<td>D82</td>
<td>S2</td>
<td>DISK110 210 300 400</td>
<td>128</td>
</tr>
<tr>
<td>$1$SDUA83</td>
<td>See note 2</td>
<td></td>
<td>D83</td>
<td>S4</td>
<td>DISK120 220 300 400</td>
<td>128</td>
</tr>
<tr>
<td>$1$SDUA84</td>
<td>DISK16D</td>
<td>94 database data disk</td>
<td>D84</td>
<td>S5</td>
<td>DISK130 230 320 420</td>
<td>128</td>
</tr>
<tr>
<td>$1$SDUA85</td>
<td>DISK16I</td>
<td>94 database index disk</td>
<td>D85</td>
<td>S7</td>
<td>DISK140 240 340 440</td>
<td>128</td>
</tr>
<tr>
<td>$1$SDUA86</td>
<td>See note 2</td>
<td></td>
<td>D86</td>
<td>S8</td>
<td>DISK150 250 350 450</td>
<td>128</td>
</tr>
<tr>
<td>$1$SDUA87</td>
<td>See note 2</td>
<td></td>
<td>D87</td>
<td>S3</td>
<td>DISK310 410 510 610</td>
<td>128</td>
</tr>
<tr>
<td>$1$SDUA88</td>
<td>SORT1</td>
<td>95 database sort disk</td>
<td>D88</td>
<td>S6</td>
<td>DISK330 430 530 630</td>
<td>128</td>
</tr>
<tr>
<td>$1$SDUA89</td>
<td>SORTB</td>
<td>94 database sort disk</td>
<td>D89</td>
<td>S9</td>
<td>DISK350 450 550 650</td>
<td>128</td>
</tr>
</tbody>
</table>

1 S1, S2, … = stripeset, R1, R2, … = RAIDset.
2 Used for temp and sort disks during the database build.
Table 6-13: Database Storage Map for HSJ009

<table>
<thead>
<tr>
<th>OpenVMS Device Name</th>
<th>Volume Label</th>
<th>Volume Description</th>
<th>HSJ LUN</th>
<th>HSJ Container Name</th>
<th>HSJ Physical Devices</th>
<th>Chunk Size (blocks)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$1$DUA91</td>
<td>SORT2</td>
<td>95 database sort disk</td>
<td>D91</td>
<td>S1</td>
<td>DISK100 200 300 400</td>
<td>128</td>
</tr>
<tr>
<td>$1$DUA92</td>
<td>SORTA</td>
<td>94 database sort disk</td>
<td>D92</td>
<td>S2</td>
<td>DISK110 210 500 600</td>
<td>128</td>
</tr>
<tr>
<td>$1$DUA93</td>
<td>See note 2</td>
<td></td>
<td>D93</td>
<td>S4</td>
<td>DISK120 220 320 420</td>
<td>128</td>
</tr>
<tr>
<td>$1$DUA94</td>
<td>See note 2</td>
<td></td>
<td>D94</td>
<td>S5</td>
<td>DISK130 230 520 620</td>
<td>128</td>
</tr>
<tr>
<td>$1$DUA95</td>
<td>ROOT0</td>
<td>root disk 94/95</td>
<td>D95</td>
<td>S3</td>
<td>DISK310 410 510 610</td>
<td>128</td>
</tr>
<tr>
<td>$1$DUA96</td>
<td>STORAGE</td>
<td>??</td>
<td>D96</td>
<td>S6</td>
<td>DISK330 430 530 630</td>
<td>128</td>
</tr>
</tbody>
</table>

1. $S_1$, $S_2$, … = stripeset, $R_1$, $R_2$, … = RAIDset.
2. Used for temp and sort disks during the database build.
Operational Command and Supporting Files

The following files were used to set up, load, or run tests on the CPG database. Details about file contents are available upon request.

**BINARY.COM**

A DCL command procedure to create binary input load files. BINARY.COM created twelve files, each 5.5 GB per month, that constituted a year’s worth of sales data.

**CACHES.SQL**

An SQL script to create VLM row caches. This script is included in the CPGxx.SQL script (where “xx” is the year number).

**CPG.COM**

A DCL command procedure to create RMS search list logicals for all keys directories (i.e., CPG - all database storage directories, TMP - all temporary files, SRT - all sort devices, AIJ - all after image journal files).

**CPG94.SQL**

An SQL script to create the 1994 database using SQL$CPG94 as the root name.

**CPG95.SQL**

An SQL script to create the 1995 database using SQL$CPG95 as the root name.

**DBLSQL**

An SQL file that creates the query catalog used to direct the queries to the appropriate database.

**DISKS.93**

A device list file detailing the devices comprising the 1993 database.
DISKS.94
A device list file detailing the devices comprising the 1994 database.

DISKS.95
A device list file detailing the devices comprising the 1995 database.

DISMOUNT.COM
A DCL command procedure to dismount files.

INDEXES-94.SQL
Creates the indexes for the 1994 database on the AlphaServer 4100 system.

INDEXES-95.SQL
Creates the indexes for the 1995 database on the AlphaServer 8400 system.

INIT.COM
A command file that initializes the database after a system boot.

JOURNALS-94.COM
Creates the journals for the 1994 database on the AlphaServer 4100 system.

JOURNALS-95.COM
Creates the journals for the 1995 database on the AlphaServer 8400 system.

LOAD-DIM.COM
A command file that loads the small dimension tables.

LOADER.COM
A DCL command procedure to create ASCII input load files. LOADER.COM created twelve files, each 6.8 GB per month, that constituted a years worth of sales data.

LOGICALS.COM
A DCL command procedure to define device logical names.

LOGIN.COM
A login command procedure to establish the VMSS$MENU logical names.
MOUNT.COM
A DCL command procedure to mount disk devices.

RMU-COPY95.COM
A DCL command procedure to perform an RMU copy of the 1995 database.

RMUBIN.COM
A DCL command file that serially loads the sales facts table from binary load files.

RMUBIN17.COM
A DCL command file that loads the sales facts table in parallel using 17 threads.

RMUBLD.COM
A DCL command file that loads the sales facts table from ASCII load files.

SF_KEY-94.SQL
An SQL script used for contingencies to build the index on the sales facts table for 1994.

SF_KEY-95.SQL
An SQL script used for contingencies to build the index on the sales facts table for 1995.