

# Oracle8™

## Administrator's Reference

Release 8.0.6 for IBM DYNIX/ptx

July 2000

Part No. A85379-01

### Topics Including:

Optimal Flexible Architecture on Oracle8

Administering Oracle8 on DYNIX/ptx

Tuning Oracle8 on DYNIX/ptx

Administering SQL\*Plus on DYNIX/ptx

Using Oracle Precompilers and the Oracle Call Interface on DYNIX/ptx

Configuring Oracle Net8

Running Oracle Data Cartridge Demos on DYNIX/ptx

**ORACLE®**

---

Oracle8 Administrator's Reference for IBM DYNIX/ptx

Release 8.0.6

Part No. A85379-01

Copyright © 2000, Oracle Corporation. All rights reserved.

Authors: Zeynep Taspinar and Tom Kavanaugh, with Jamshed Patel, Rajesh Shah, Mike Kavanaugh, Kevin Adams, Nicholas Hind, Nik Ormseth, Lynn Robinson, Sally Norton, Joycelyn Wee, and Pallavi Bhowmik.

**The programs are not intended for use in any nuclear, aviation, mass transit, medical, or other inherently dangerous applications. It shall be licensee's responsibility to take all appropriate fail-safe, back up, redundancy and other measures to ensure the safe use of such applications if the Programs are used for such purposes, and Oracle disclaims liability for any damages caused by such use of the Programs.**

This Program contains proprietary information of Oracle Corporation; it is provided under a license agreement containing restrictions on use and disclosure and is also protected by copyright patent and other intellectual property law. Reverse engineering of the software is prohibited.

The information contained in this document is subject to change without notice. If you find any problems in the documentation, please report them to us in writing. Oracle Corporation does not warrant that this document is error free.

If this Program is delivered to a U.S. Government Agency of the Department of Defense, then it is delivered with Restricted Rights and the following legend is applicable:

**Restricted Rights Legend** Programs delivered subject to the DOD FAR Supplement are 'commercial computer software' and use, duplication and disclosure of the Programs shall be subject to the licensing restrictions set forth in the applicable Oracle license agreement. Otherwise, Programs delivered subject to the Federal Acquisition Regulations are 'restricted computer software' and use, duplication and disclosure of the Programs shall be subject to the restrictions in FAR 52..227-14, Rights in Data -- General, including Alternate III (June 1987). Oracle Corporation, 500 Oracle Parkway, Redwood City, CA 94065.

**Registered Trademarks of Oracle Corporation** Oracle, the Oracle logo, Oracle Book, ConText, Oracle ConText, Oracle Names, Pro\*Ada, Pro\*COBOL, Pro\*FORTRAN, Pro\*Pascal, Pro\*PL/I, SQL\*Loader, SQL\*Module, SQL\*Net, and SQL\*Plus are registered trademarks of Oracle Corporation.

**Non-Registered Trademarks of Oracle Corporation** Advanced Networking Option, Advanced Replication Option, Developer/2000, Enabling the Information Age, InterOffice, JDBC OCI Driver, JDBC Thin Driver, Network Computing Architecture, Oracle Applications, Oracle Call Interface, Oracle Data Gatherer, Oracle Enterprise Manager, Oracle Installer, Oracle InterOffice, Oracle Multiprotocol Interchange, Oracle Network Manager, Oracle Objects Option, Oracle Parallel Server, Oracle Parallel Server Management Components, Oracle Partitioning Option, Oracle Server Manager, Oracle Time Series Cartridge, Oracle Toolkit, Oracle TRACE, Oracle Visual Information Retrieval Cartridge, Oracle WebServer, Oracle7, Oracle7 Enterprise Backup Utility, Oracle8, Net8, PL/SQL, Pro\*C/C++, and Trusted Oracle are trademarks of Oracle Corporation.

UNIX is a registered trademark in the United States and other countries, licensed exclusively through X/Open Company Limited. All other products or company names are used for identification purposes only, and may be trademarks of their respective owners.

For more information about Oracle's trademarks and intellectual property policies, contact the Oracle Legal Department at (650) 506-5100.

---

---

# Contents

<b>Send Us Your Comments .....</b>	<b>ix</b>
<b>Preface.....</b>	<b>xi</b>
<b>Purpose .....</b>	<b>xi</b>
<b>Audience .....</b>	<b>xi</b>
<b>Typographic Conventions .....</b>	<b>xii</b>
<b>Command Syntax.....</b>	<b>xii</b>
<b>Contacting Customer Support.....</b>	<b>xiii</b>
<b>Related Documentation .....</b>	<b>xiv</b>
<b>Ordering Documentation .....</b>	<b>xiv</b>
<b>Shipping Inquiries .....</b>	<b>xiv</b>
 <b>1 Optimal Flexible Architecture on Oracle8</b>	
<b>Optimal Flexible Architecture (OFA) .....</b>	<b>1-2</b>
Characteristics of OFA-Compliant Database .....	1-2
<b>OFA Implemented on Oracle8 for UNIX .....</b>	<b>1-4</b>
Naming Mount Points.....	1-4
Naming Directories .....	1-5
Naming Files .....	1-6
Naming Tablespaces.....	1-8
Exploiting OFA Structure for Oracle Files .....	1-9
OFA File Mapping .....	1-10
Raw Device Sizes .....	1-11
File Mapping for Multiple-Instance OFA Database.....	1-11

Directory Structure .....	1-13
Default OFA Database.....	1-17

## 2 Administering Oracle8 on DYNIX/ptx

<b>Customizing the initsid.ora File</b> .....	2-2
Sample initsid.ora File .....	2-2
<b>Setting the Environment</b> .....	2-4
Displaying and Setting Environment Variables .....	2-4
Setting a Common Environment .....	2-5
Database Examples.....	2-6
<b>Environment Variables for Oracle8</b> .....	2-7
Oracle Environment Variables on UNIX .....	2-7
UNIX Environment Variables Used with Oracle8 .....	2-11
Setting the System Time.....	2-13
<b>Estimating Oracle8 Server Memory Usage</b> .....	2-13
<b>Calculating Cluster Size and Index Size</b> .....	2-14
Calculating Cluster Size .....	2-14
Calculating Index Size.....	2-14
Server Resource Limits.....	2-15
<b>Initialization Parameters</b> .....	2-15
Default Initialization Parameter Values.....	2-15
Obsolete Oracle7 initsid.ora Parameters.....	2-18
<b>Controlling the System Global Area</b> .....	2-19
Size limits of the SGA .....	2-19
Calculating the Size of the SGA.....	2-19
Relocating the SGA .....	2-20
<b>Managing Special Accounts and Groups</b> .....	2-21
<b>Managing Security</b> .....	2-22
Groups and Security .....	2-22
Security for Oracle Server Utilities .....	2-23
Security for Server Manager Commands .....	2-24
Security for Database Files.....	2-24
Network Security .....	2-25
Enabling Automatic Logins for Oracle Net8.....	2-25
Checking Order.....	2-26

Security and Remote Passwords.....	2-28
<b>Administering Login Home Directories</b> .....	2-30
<b>Building and Running Demonstrations</b> .....	2-32
Loading PL/SQL Demonstrations .....	2-32
Running PL/SQL Demonstrations.....	2-33
SQL*Loader Demonstrations .....	2-34
Administering SQL*Loader .....	2-35
Oracle Security Server.....	2-37
<b>Oracle8 Server SQL Reference</b> .....	2-37
CREATE CONTROLFILE Parameters .....	2-37

### 3 Tuning Oracle8 on DYNIX/ptx

<b>The Importance of Tuning</b> .....	3-2
Before Tuning the System.....	3-2
<b>DYNIX/ptx Tools</b> .....	3-2
sar .....	3-2
swap .....	3-3
<b>SQL Scripts</b> .....	3-4
utlbstat and utlestat SQL Scripts .....	3-4
<b>Tuning Memory Management</b> .....	3-4
Allocate Sufficient Swap Space .....	3-4
Control Paging .....	3-5
Hold the SGA in a Single Shared Memory Segment .....	3-5
Optimize Number of Database Buffers .....	3-6
Use Indirect Database Buffers.....	3-6
Optimize Number of Redo Buffers .....	3-8
Optimize the Shared Pool Size.....	3-8
Verify Data Dictionary Cache Effectiveness .....	3-9
Allocate Adequate Library Cache Space.....	3-9
Lock Large PL/SQL Blocks into the Shared Pool.....	3-10
Optimize the Session Cache Cursors.....	3-10
<b>Tuning Disk I/O</b> .....	3-11
Use Logical Volumes .....	3-11
Choose the Appropriate File System or Raw Devices .....	3-12
Tune the Database Writer to Increase Write Bandwidth .....	3-12

Separate Indexes from Tables.....	3-14
Place Redo Logs on their Own Disk Device .....	3-15
Look for Large Disk Request Queues .....	3-15
Move Hot Files to Other Disks.....	3-15
Reduce I/O to Hot Files .....	3-16
Table Striping .....	3-16
Check for Excessive Database Fragmentation .....	3-16
When Disk I/O Optimizations Fail .....	3-17
Disk Performance Issues .....	3-18
<b>Monitoring Disk Performance .....</b>	<b>3-18</b>
<b>Tuning CPU Usage .....</b>	<b>3-19</b>
Keep All Oracle Users/Processes at the Same Priority .....	3-19
Use Processor Affinity/Binding on Multi-Processor Systems.....	3-19
Reorganize Usage Patterns .....	3-19
Use a Client/Server Configuration.....	3-19
Use the Post-Wait Driver.....	3-20
Use Single-Task Linking for Large Exports/Imports and SQL*Loader Jobs .....	3-20
<b>Tuning Oracle Resource Contention .....</b>	<b>3-20</b>
Tune UNIX Kernel Parameters .....	3-21
Use V\$ Tables to Isolate Contention .....	3-21
Isolate the Segment Causing Contention .....	3-22
Reduce Latch Free Contention .....	3-24
Reduce Rollback Segment Contention .....	3-24
Reduce Redo Log Buffer Latch Contention .....	3-25
Reduce Parallel Query/Parallel DML Contention .....	3-25
Tune Spin Count.....	3-26
<b>Tuning Block Size and File Size .....</b>	<b>3-27</b>
Tune Block Size.....	3-27
<b>Tuning the DYNIX/ptx Buffer Cache Size .....</b>	<b>3-27</b>
<b>Tuning Resource Contention for Oracle Parallel Server .....</b>	<b>3-29</b>
Avoid Index Contention .....	3-29
Avoid Free List Contention .....	3-29
Avoid Lock Contention .....	3-29
Localize Disk I/O.....	3-30
Monitor Contention .....	3-30

<b>Using Trace and Alert Files .....</b>	<b>3-31</b>
Trace File Names .....	3-31
Alert Files .....	3-31
<b>Raw Devices .....</b>	<b>3-32</b>
Disadvantages of Raw Devices .....	3-32
Guidelines for Using Raw Devices .....	3-33
Setting Up Raw Devices.....	3-34

## 4 Administering SQL\*Plus on DYNIX/ptx

<b>Administering SQL*Plus .....</b>	<b>4-2</b>
Setup Files .....	4-2
The Site Profile .....	4-2
The User Profile .....	4-2
The PRODUCT_USER_PROFILE Table .....	4-3
Demonstration Tables .....	4-3
Help Facility .....	4-4
<b>Using SQL*Plus .....</b>	<b>4-6</b>
Using a System Editor from SQL*Plus .....	4-6
Setting the Order of the Editor .....	4-6
Setting the _editor option .....	4-6
Setting Environment Variables .....	4-6
Default Settings .....	4-7
Running Operating System Commands from SQL*Plus .....	4-7
Interrupting SQL*Plus .....	4-8
Using the SPOOL Command .....	4-8
<b>Restrictions .....</b>	<b>4-8</b>
COPY Command .....	4-8
Resizing Windows.....	4-9
Return Codes .....	4-9

## 5 Using Oracle Precompilers and the Oracle Call Interface on DYNIX/ptx

<b>Overview of Oracle Precompilers .....</b>	<b>5-2</b>
Relinking Precompiler Executables .....	5-2
Precompiler Configuration Files .....	5-3
Issues Common to All Precompilers .....	5-3

Supplemental Documentation.....	5-4
<b>Pro*C/C++ .....</b>	<b>5-4</b>
Administering Pro*C/C++ .....	5-4
Using Pro*C/C++ .....	5-4
<b>Pro*COBOL.....</b>	<b>5-6</b>
Administering Pro*COBOL .....	5-7
Environment Variables .....	5-7
Using Pro*COBOL.....	5-8
<b>Pro*FORTRAN .....</b>	<b>5-10</b>
Administering Pro*FORTRAN .....	5-10
Using Pro*FORTRAN .....	5-10
<b>Oracle Call Interface .....</b>	<b>5-12</b>
Using the Oracle Call Interface .....	5-12
<b>Oracle Precompiler and Oracle Call Interface Linking and Makefiles .....</b>	<b>5-14</b>
Custom Makefiles .....	5-14
Undefined Symbols .....	5-15
<b>Thread Support .....</b>	<b>5-15</b>
<b>Static and Dynamic Linking with Oracle Libraries .....</b>	<b>5-15</b>
<b>Using Signal Handlers .....</b>	<b>5-17</b>
Signals .....	5-17
<b>XA Functionality .....</b>	<b>5-20</b>

## 6 Configuring Oracle Net8

<b>Supplemental Documentation.....</b>	<b>6-2</b>
Supplementary Information in README Files .....	6-2
<b>Core Net8 Products and Features .....</b>	<b>6-3</b>
Net8 Files and Utilities .....	6-3
Oracle Connection Manager .....	6-4
Multi-Threaded Server .....	6-4
Oracle Names .....	6-4
Net8 Assistant .....	6-5
<b>Oracle Net8 Protocol Adapters .....</b>	<b>6-5</b>
<b>The BEQ Protocol Adapter .....</b>	<b>6-7</b>
Overview of the BEQ Protocol Adapter .....	6-7
Specifying a BEQ ADDRESS .....	6-7



<b>The IPC Protocol Adapter .....</b>	<b>6-9</b>
Overview of the IPC Protocol Adapter .....	6-9
Specifying an IPC ADDRESS .....	6-9
<b>The RAW Protocol Adapter .....</b>	<b>6-10</b>
<b>The TCP/IP Protocol Adapter .....</b>	<b>6-11</b>
Overview of the TCP/IP Protocol Adapter .....	6-11
Specifying a TCP/IP ADDRESS .....	6-11
<b>The SPX/IPX Protocol Adapter .....</b>	<b>6-12</b>
The <b>ntisbsd</b> m Broadcast Daemon .....	6-12
The <b>ntspxctl</b> Utility .....	6-12
SPX/IPX Protocol Adapter Command Summary .....	6-14
The <b>getname</b> Command .....	6-14
Specifying the SPX/IPX ADDRESS .....	6-16
<b>Oracle Enterprise Manager (OEM) Intelligent Agent .....</b>	<b>6-17</b>
<b>Oracle Advanced Networking Option .....</b>	<b>6-18</b>

## 7 Running Oracle Data Cartridge Demos on DYNIX/ptx

<b>Issues Common to Data Cartridges .....</b>	<b>7-2</b>
<b>Oracle8 Time Series Cartridge.....</b>	<b>7-2</b>
Installing Time Series Cartridge Demos .....	7-2
<b>Oracle8 Visual Information Retrieval Cartridge .....</b>	<b>7-2</b>
Creating the Demo.....	7-2
<b>Oracle8 Image Cartridge .....</b>	<b>7-4</b>
Creating the Demo.....	7-4

## Index



---

# Send Us Your Comments

## **Oracle8 Administrator's Reference, Release 8.0.6 for IBM DYNIX/ptx Part No. A85379-01**

Oracle Corporation welcomes your comments and suggestions on the quality and usefulness of this publication. Your input is an important part of the information used for revision.

- ☐ Did you find any errors?
- ☐ Is the information clearly presented?
- ☐ Do you need more information? If so, where?
- ☐ Are the examples correct? Do you need more examples?
- ☐ What features did you like most about this manual?

If you find any errors or have other suggestions for improvement, please indicate the book title, part number, chapter, and section. You can send comments to:

Tom Leah-Martin, Technical Documentation Manager  
Platform Technologies Division  
500 Oracle Parkway, Mailstop 1op4  
Redwood Shores, CA 94065  
USA

If you would like a reply, please give your name, postal or email address, and telephone number.

If you have problems with the software, please contact your local Oracle Support Center.



---

---

# Preface

## Purpose

This reference provides UNIX-specific and IBM DYNIX/ptx-specific information required to successfully administer and tune the Oracle8 Server. This reference supplements information found in the Oracle8 and Oracle8 Enterprise Edition documentation set.

## Audience

This document is intended for anyone responsible for administering the Oracle8 Server on an IBM DYNIX/ptx system.

# Typographic Conventions

<code>monospace</code>	Monospace type indicates UNIX commands, directory names, path names, and filenames.
brackets [ ]	Words enclosed in brackets indicate key names (for example, Press [Return]). Note that brackets have a different meaning when used in command syntax.
<i>italics</i>	Italic type indicates a variable, including variable portions of filenames, or emphasis.
UPPERCASE	Uppercase letters indicate Structured Query Language (SQL) commands, initialization parameters, or environment variables.

Because UNIX is case-sensitive, conventions in this document may differ from those used in Oracle product documentation.

## Command Syntax

Command syntax appears in `monospace` font. The following conventions apply to command syntax:

backslash \	A backslash indicates a command that is too long to fit on a single line. Enter the line as printed (with a backslash) or enter it as a single line without a backslash: <pre>dd if=/dev/rdsd/c0t1d0s6 of=/dev/rst0 bs=10b \ count=10000</pre>
braces { }	Braces indicate required items: <code>.DEFINE {macro1}</code>
brackets [ ]	Brackets indicate optional items: <code>cvtrct termname [outfile]</code>  Note that brackets have a different meaning when used in regular text.
ellipses ...	Ellipses indicate an arbitrary number of similar items: <code>CHKVAL fieldname value1 value2 ... valueN</code>
<i>italics</i>	Italic type indicates a variable. Substitute a value for the variable: <i>library_name</i>
vertical line	A vertical line indicates a choice within braces or brackets: <code>SIZE filesize [K M]</code>

# Contacting Customer Support

*Please copy this page and distribute it within your organization as necessary.*

Oracle Support Services (OSS) can be reached at the following numbers (the hours are specified in your support contract):

- In the United States, call: **1.650.506.1500**.
- In Europe, call: **+44.1344.860160**.
- In Asia, call: **+81.3.5717.1860**.

Please prepare the following information before you call:

- Your CSI number (if applicable) or complete contact details, including any special project information.
- The release levels of the Oracle Server and associated products (for example, Oracle8 Server release 8.0.6.0, and Oracle Forms release 4.5.6.3.2).
- Operating system name and release level, including patches and packages.
- Details of error codes, numbers, and descriptions associated with the problem.
- A full description of the issue, including:
  - What happened? For example, the command used and result obtained.
  - When did it happen? For example, time of day, or after a particular command, or after an operating system or Oracle upgrade.
  - Where did it happen? For example, on a particular system, or within a particular procedure or table.
  - What is the extent of the problem? For example, is your production system unavailable, or is the impact less severe? Is the problem getting worse?

Keep in mind what did *not* happen, as well as what did happen.

- Copies of any trace files, core dumps, or log files recorded near the time of the incident.

For installation-related problems, please have the following information available:

- Listings of the contents of the ORACLE\_HOME directory, and any staging area, if applicable.
- Contents of the installation log files in the \$ORACLE\_HOME/orainst directory: install.log, sql.log, make.log, and os.log.

*For more information, see <http://www.oracle.com/support>.*

## Related Documentation

Advanced configuration and tuning recommended for a production database system is provided in the following manuals:

- *Oracle8 Administrator's Guide*. Use this as a starting point for tasks associated with the Oracle8 Server, such as database creation, managing database objects, and creating users.
- *Net8 Administrator's Guide*
- *Oracle8 Tuning*

Unfamiliar with the concepts or terminology associated with relational database management systems? Read Chapter 1 in the *Oracle8 Concepts* before beginning your installation.

## Ordering Documentation

- In the United States, call Documentation Sales at: **1.800.252.0303**.
- In the United Kingdom, call Oracle Direct Response at: **+44.990.332200**.
- In other European countries, contact your local Oracle Support office.
- In the Asia-Pacific region, contact your Oracle sales representative.

## Shipping Inquiries

- In the United States, call Client Relations at: **1.650.506.1500**.
- In the United Kingdom, call Customer Relations at: **+44.990.622300**.
- In other European countries, contact your local Oracle Support office.
- In the Asia-Pacific region, contact your Oracle sales representative.



---

# Optimal Flexible Architecture on Oracle8

- Optimal Flexible Architecture (OFA)
- OFA Implemented on Oracle8 for UNIX

## Optimal Flexible Architecture (OFA)

Oracle Corporation recommends the Optimal Flexible Architecture (OFA) standard for Oracle8. The OFA standard is a set of configuration guidelines for fast, reliable Oracle databases that require little maintenance.

OFA is designed to:

- Organize large amounts of complicated software and data on disk to avoid device bottlenecks and poor performance
- Facilitate routine administrative tasks like software and data backup functions, which are often vulnerable to data corruption
- Alleviate switching among multiple Oracle databases
- Adequately manage and administer database growth
- Help eliminate fragmentation of free space in the data dictionary, isolate other fragmentation, and minimize resource contention

## Characteristics of OFA-Compliant Database

An OFA-compliant database provides the following benefits:

### File System Organization

The file system is organized to allow easy administration and accommodate scalability for:

- Adding data into existing databases
- Adding users
- Creating databases
- Adding hardware

### Distributed I/O Loads

I/O loads are distributed across enough disk drives to prevent performance bottlenecks.

### Hardware Support

Hardware costs are minimized only when they do not conflict with operational considerations.

### **Safeguards Against Drive Failures**

By spreading applications across more than one drive, drive failures impact as few applications as possible.

### **Distribution of Home Directories**

The following items can be distributed across more than one disk drive:

- Collection of home directories
- Contents of an individual home directory

### **Integrity of Login Home Directories**

It is possible to add, move, or delete login home directories without having to revise programs that refer to them.

### **Independence of UNIX Directory Subtrees**

Categories of files are separated into independent UNIX directory subtrees so that files in one category are minimally affected by operations on files in other categories.

### **Supports Concurrent Execution of Application Software**

It is possible to execute multiple versions of applications software simultaneously, allowing the user to test and use a new release of an application before abandoning the previous version. Transferring to a new version after an upgrade is simple for the administrator and transparent for the user.

### **Distinguishes Administrative Information for each Database**

The ability to separate administrative information about one database from that of another, ensures a reasonable structure for the organization and storage of administrative data.

### **Uses Consistent Database File Naming**

Database files are named so that:

- Database files are easily distinguishable from all other files
- Files of one database are easily distinguishable from files of another database
- Control files, redo log files, and data files are identifiable
- Association of data file to tablespace is clearly indicated

### **Separation of Tablespace Contents**

Tablespace contents are separated to:

- Minimize tablespace free space fragmentation
- Minimize I/O request contention
- Maximize administrative flexibility

### **Tuning of I/O Loads across all Drives**

I/O loads are tuned across all drives, including drives storing Oracle data in raw devices.

### **Additional Benefits of OFA for Parallel Server**

For Oracle Parallel Server Installations:

- Administrative data is stored in a central place, accessible to all database administrators
- Administrative data for an instance is associated with the instance by the file name

## **OFA Implemented on Oracle8 for UNIX**

A careful naming strategy for database files eliminates data administration problems. The OFA rules provided here correspond to the original OFA recommendations published in *The OFA Standard: Oracle8 for Open Systems*.

## **Naming Mount Points**

### **Mount Point Syntax**

Name all mount points using the syntax */pm*, where *p* is a string constant and *m* is a unique fixed-length key (typically a two-digit number) used to distinguish each mount point. For example: */u01* and */u02*, or */disk01* and */disk02*.

### Naming Mount Points for Very Large Databases (VLDBs)

If each disk drive contains database files from one application and there are enough drives for each database to ensure no I/O bottleneck, then use the syntax */q/dm* for naming mount points, as explained in Table 1–1.

**Table 1–1   Syntax for Naming Mount Points**

<i>q</i>	a string denoting that Oracle data is stored here
<i>dm</i>	the value of the initialization parameter DB_NAME (synonymous with the instance <i>sid</i> for single-instance databases)

For example, mount points named */u01/oradata/test01* and */u01/oradata/test02* allocate two drives for the Oracle test database.

## Naming Directories

### Home Directory Syntax

Name home directories using the syntax */pm/h/u*, as explained in Table 1–2.

**Table 1–2   Syntax for Naming Home Directories**

<i>pm</i>	a mount point name
<i>h</i>	a standard directory name
<i>u</i>	the name of the owner of the directory

For example, */u01/app/oracle* is the Oracle server software owner home directory (also referred to as ORACLE\_BASE and defaulted by the Installer) and */u01/app/applmgr* is an Oracle applications software owner home directory.

Placing home directories at the same level in the UNIX file system is advantageous for the following reason: it allows the collection of applications owner login home directories on different mount points, to be referred to with the single pattern matching string, */\*/app/\**.

### Referring to Pathnames

Refer to explicit pathnames only in files designed specifically to store them, such as */etc/passwd* and the Oracle *oratab* file. Refer to group memberships only in the */etc/group* file.

### Software Directories

In order to help fulfill the OFA requirement that it be possible to simultaneously execute multiple versions of application software, store each version of the Oracle8 Server software in a directory matching the pattern `/pm/h/product/v`, as explained in Table 1–3.

**Table 1–3   Syntax for Naming Oracle8 Server Software Directories**

<i>h</i>	a standard directory name
<i>v</i>	the version of the software

For example: `/u01/app/oracle/product/8.0.6` indicates the start of the directory structure where the Oracle8 Server files are located. Set the `ORACLE_HOME` environment variable to this directory.

## Naming Files

### Administration Files

To facilitate the organization of administrative data, it is recommended that you store database-specific administration files in subdirectories according to `h/admin/d/a/`, where *h* is the Oracle software owner’s home directory, *d* is the database name (`DB_NAME`), and *a* is a subdirectory for each of the following database administration files described in Table 1–4:

**Table 1–4   Subdirectories for Database Administration Files**

<code>adhoc</code>	ad hoc SQL scripts for a given database
<code>arch</code>	archived redo log files
<code>adump</code>	audit files (Set <code>AUDIT_FILE_DEST</code> in <code>configdb_name.ora</code> to point here. This subdirectory should be cleaned out periodically).
<code>bdump</code>	background process trace files
<code>cdump</code>	core dump files
<code>create</code>	programs used to create the database
<code>exp</code>	database export files
<code>logbook</code>	files recording the status and history of the database
<code>pfile</code>	instance parameter files
<code>udump</code>	user SQL trace files

As an example, the subdirectory `adhoc` would have the following pathname:  
`/u01/app/oracle/admin/sab/adhoc/`

Database Files

The following naming convention for database files ensures that they are easily identifiable:

- n For control files, use `/pm/q/d/control.ctl`
- n For redo log files, use `/pm/q/d/redon.log`
- n For data files use, `/pm/q/d/tn.dbf`

This syntax is explained in Table 1–5.

Table 1–5 Syntax for Naming Database Files

<i>pm</i>	a mount point name described earlier in this chapter
<i>q</i>	a string distinguishing Oracle data from all other files (usually named ORACLE or oradata)
<i>d</i>	the DB_NAME of the database
<i>t</i>	an Oracle tablespace name
<i>n</i>	a two-digit string

**Note:** Do not store files other than a control, redo log or data file associated with database *d* in the path `/pm/q/d`.

Following this convention could produce for example, a data file with the name, `/u03/oradata/sab/system01.dbf`, making it easy to see to which database the file belongs.

Separate Segments with Different Requirements

Separate groups of segments with different lifespans, I/O request demands, and backup frequencies across different tablespaces.

For each Oracle database, create the special tablespaces described in Table 1–6. These tablespaces are in addition to those needed for application segments.

**Table 1–6 Special Tablespaces**

SYSTEM	data dictionary segments
TEMP	temporary segments
RBS	rollback segments
TOOLS	general-purpose tools
USERS	miscellaneous user segments

This method is effective because dictionary segments are never dropped, and no other segments that can be dropped are allowed in the SYSTEM tablespace. This ensures that the SYSTEM tablespace does not require a rebuild due to tablespace free space fragmentation.

Because rollback segments are not stored in tablespaces holding applications data, the administrator is not blocked from taking an application’s tablespace offline for maintenance. The segments are partitioned physically by type, and the administrator can record and predict data growth rates without complicated tools.

## Naming Tablespaces

Name tablespaces descriptively using a maximum of eight characters.

Although Oracle8 tablespace names can be thirty characters long, portable UNIX file names are restricted to fourteen characters. The recommended standard for a data file basename is *tn.dbf*, where *t* is a descriptive tablespace name and *n* is a two-digit string. Because the extension plus the two-digit string occupy a total of six characters, only eight characters remain for the tablespace name.

Descriptive names allow the name of a data file to be associated with the tablespace that uses it. For example, the names GLD and GLX might be used for the tablespaces storing General Ledger data and indices, respectively.

---

**Note:** Do not embed reminders of the word “tablespace” in your tablespace names. Tablespaces are distinguishable by context, and names do not need to convey information about type.

---



## Exploiting OFA Structure for Oracle Files

Table 1–7 shows the syntax used for identifying classes of files.

**Table 1–7    Directory Structure Syntax for Identifying Classes of Files**

/u[0-9][0-9]	user data directories
/*/home/*	user home directories
/*/app/*	user application software directories
/*/app/applmgr	Oracle apps software subtrees
/*/app/oracle/product	Oracle Server software subtrees
/*/app/oracle/product/8.0.6	Oracle Server 8.0.6 distribution files
/*/app/oracle/admin/sab	sab database administrative subtrees
/*/app/oracle/admin/sab/arch/*	sab database archived log files
/*/oradata	Oracle data directories
/*/oradata/sab/*	sab database files
/*/oradata/sab/*.log	sab database redo log files

## OFA File Mapping

Table 1–8 shows an hierarchical file mapping of a sample OFA-compliant database, including each file’s mount point, application, database, and tablespace. The file names indicate the file type (control, log, or data).

**Table 1–8 Hierarchical File Mapping for OFA Installation**

/	root mount point
u01/	'user data' mount point #1
app/	subtree for app software
oracle/	home for <i>oracle</i> software owner
admin/	subtree for database admin files
TAR/	subtree for Support logs
db_name1/	admin subtree for <i>db_name1</i> database
db_name2/	admin subtree for <i>db_name2</i> database
doc/	online documentation
local/	subtree for local Oracle software
aps6/	an Oracle6 admin package
aps7/	an Oracle7 admin package
product/	distribution files
7.3.2/	ORACLE_HOME for 7.3.2 instances
7.3.3/	ORACLE_HOME for 7.3.3 instances
8.0.6/	ORACLE_HOME for 8.0.6 instances
home/	subtree for login home directories
ltb/	home for a user
sbm/	home for a user
oradata/	subtree for Oracle data
db_name1/	subtree for <i>db_name1</i> database files
db_name2/	subtree for <i>db_name2</i> database files
u02/	'user data' mount point #2
home/	subtree for login home directories
cvm/	home for a user
vrm/	home for a user
oradata/	subtree for Oracle data
db_name1/	subtree for <i>db_name1</i> database files
db_name2/	subtree for <i>db_name2</i> database files
u03/	'user data' mount point #3
home/	subtree for login home directories
oradata/	subtree for Oracle data
db_name1/	subtree for <i>db_name1</i> database files
db_name2/	subtree for <i>db_name2</i> database files

## Raw Device Sizes

Choose a small set of standard sizes for all raw devices that may be used to store Oracle database files.

In general, standardizing on a single size is recommended. If a single size is used, raw files can be moved from one partition to another safely. The size should be small enough so that a fairly large number can be created, but large enough to be convenient.

For example, a 2 GB drive could be divided into 10 partitions of 200 MB each—a good balance between size and number. Any tablespace using raw devices should stripe them across several drives. If possible, the striping should be done with a logical volume manager.

## File Mapping for Multiple-Instance OFA Database

Multiple-instance databases (Oracle Parallel Server installations) have an additional guideline for file mapping.

### Administrative Home for Oracle Parallel Server

When using the Oracle Parallel Server, select one node to act as the Oracle *administrative home* for the cluster. The administrative home contains the administrative subtree. Create subdirectories for each instance accessing the database within the `bdump`, `cdump`, `logbook`, `pfile`, and `udump` directories of `~/admin/d/`. The `admin` directory for the administrative home should be mounted as the `admin` directory for every instance. An example is shown in Table 1–9.

**Table 1–9 Administrative Directory Structure for Dual-Instance Oracle Parallel Server**

u01/app/oracle/admin/sab/		administrative directory for sab database
adhoc/		directory for miscellaneous scripts
arch/		log archive dest for all instances
	redo001.arc	archived redo log file
bdump/		directory for background dump files
	inst1/	background dump dest for <i>inst1</i> instance
	inst2/	background dump dest for <i>inst2</i> instance
cdump/		directory for core dump files
	inst1/	core dump dest for <i>inst1</i> instance
	inst2/	core dump dest for <i>inst2</i> instance
create/		directory for creation scripts
	1-rdbms.sql	SQL script to create <i>inst</i> database
exp/		directory for exports
	970920full.dmp	Sept 20 full export dump file
	export/	directory for export parfiles
	import/	directory for import parfiles
logbook/		directory for <i>inst</i> logbook entries
	inst1/	directory for <i>inst1</i> instance reports
		params.lst V\$PARAMETER report for <i>inst1</i> instance
	inst2/	directory for <i>inst2</i> instance reports
		params.lst V\$PARAMETER report for <i>inst2</i> instance
	user.lst	DBA_USERS report
pfile/		directory for instance parameter files
	inst1/	directory for <i>inst1</i> instance parameters
		init instance parameters for <i>inst1</i> instance
	inst2/	directory for <i>inst2</i> instance parameters
		init instance parameters for <i>inst2</i> instance
udump/		directory for user dump files
	inst1/	user dump dest for <i>inst1</i> instance
	inst2/	user dump dest for <i>inst2</i> instance

## Directory Structure

### ORACLE\_BASE Directory

ORACLE\_BASE is the root of the Oracle directory structure. ORACLE\_BASE directory structure and content is described in Table 1–10. When installing an OFA-compliant database using the Oracle Installer, ORACLE\_BASE is by default, set to `/pm/app/oracle`.

**Table 1–10 ORACLE\_BASE Directory Structure and Content**

admin	administrative files
doc	online documentation
local	subtree for local Oracle software
product	Oracle software

### ORACLE\_HOME Directory

If you install an OFA-compliant Oracle Server, the ORACLE\_HOME directory is `/mount_point/app/oracle/product/release_number`. ORACLE\_HOME directory structure and content is described in Table 1–11. Under UNIX, the ORACLE\_HOME directory contains the following subdirectories, as well as a subdirectory for each installed Oracle product:

**Table 1–11 ORACLE\_HOME Directory Structure and Content**

bin	binaries for all products
ctx	ConText cartridge
db	<code>init<sub>sid</sub>.ora</code> , <code>lk<sub>sid</sub></code>
jdbc	JDBC drivers
lib	Oracle product libraries
md	Spatial cartridge
mlx	Xerox Stemmer (for ConText cartridge)
network	Net8
nlsrtl	NLS runtime loadable data
ocommon	common files for all products
odg	data gatherer
opsm	Parallel Server Manager Components

**Table 1–11    *ORACLE\_HOME* Directory Structure and Content**

oracore	core libraries
orainst	master installation files and programs
ord	data cartridges
otrace	Oracle TRACE
plsql	PL/SQL
precomp	precompilers
rdbms	server files and libraries required for the database
slax	SLAX parser
sqlplus	SQL*Plus
svrmgr	Server Manager

**Oracle Product Subdirectories**

Product subdirectories may include those described in Table 1–12, depending on the Oracle products available on your system and the products you purchase.

**Table 1–12    *Oracle Product Subdirectories***

network	Oracle Net8
ocommon	libraries and SQL messages. All products depend on this directory, which is installed automatically
plsql	PL/SQL version 2, procedural option
sqlplus	SQL*Plus
svrmgr	Server Manager

## Contents of Product Subdirectories

Each product subdirectory contains the subdirectories described in Table 1–13.

**Table 1–13** *Contents of Product Subdirectories*

admin	administrative SQL and shell scripts (for example, <code>catalog.sql</code> , <code>catexp.sql</code> , and <code>demo.sql</code> )
admin/*	special directories for other products
admin/resource	resource files
admin/terminal	runtime terminal files
demo	demonstration scripts and datafiles
doc	README files (for example, <code>readmeunix.doc</code> )
install	product installation scripts
lib	product libraries and distributed makefiles
log	trace files and log files (for example, <code>orasrv.log</code> and <code>*.trc</code> files)
msg	U.S. message files, and Multilingual Option (formerly National Language Support) message text and binary files (for example, <code>oraus.msg</code> and <code>oraus.msb</code> )

## Examples of Product Subdirectories

Examples of product subdirectories and their contents are shown in Table 1–14.

**Table 1–14** *Examples of Product Subdirectories*

rdbms	install, lib, admin, doc, msg, log
sqlplus	install, demo, lib, admin, doc, msg

## File Naming Conventions in the admin Directory

The `rdbms/admin` directory contains the SQL scripts shown in Table 1–15.

**Table 1–15** *admin Directory, File Naming Conventions*

---

<code>cat*.sql</code>	creates catalog and data dictionary tables and views. The following files are run automatically during installation: <code>catalog.sql</code> (for all installations) <code>catproc.sql</code> (for all installations) <code>catparr.sql</code> (for Parallel Server option installations) <code>catrep.sql</code> (for all installations)
<code>dbms*.sql</code>	additional database packages
<code>utl*.sql</code>	creates tables and views for database utilities

---

## Filename Extensions

A description of filename extensions is shown in Table 1–16.

**Table 1–16** *Filename Extensions*

---

<code>.a</code>	object file libraries; Ada runtime libraries
<code>.ada</code>	Ada source files
<code>.aud</code>	Oracle audit file
<code>.bdf</code>	X11 font description file
<code>.bmp</code>	X11 bitmap file
<code>.c</code>	C source file
<code>.ctl</code>	SQL*Loader control file; Oracle Server control file
<code>.dat</code>	SQL*Loader datafile
<code>.dbf</code>	Oracle Server tablespace file
<code>.dei</code>	ORCA de-installation script
<code>.dmp</code>	Export file
<code>.doc</code>	ASCII text file
<code>.env</code>	shell script file for setting environment
<code>.f</code>	FORTTRAN source file
<code>.h</code>	C header file; also, <code>sr.h</code> is a SQL*Report Writer help file
<code>.ins</code>	ORCA installation script
<code>.l</code>	UNIX manual page



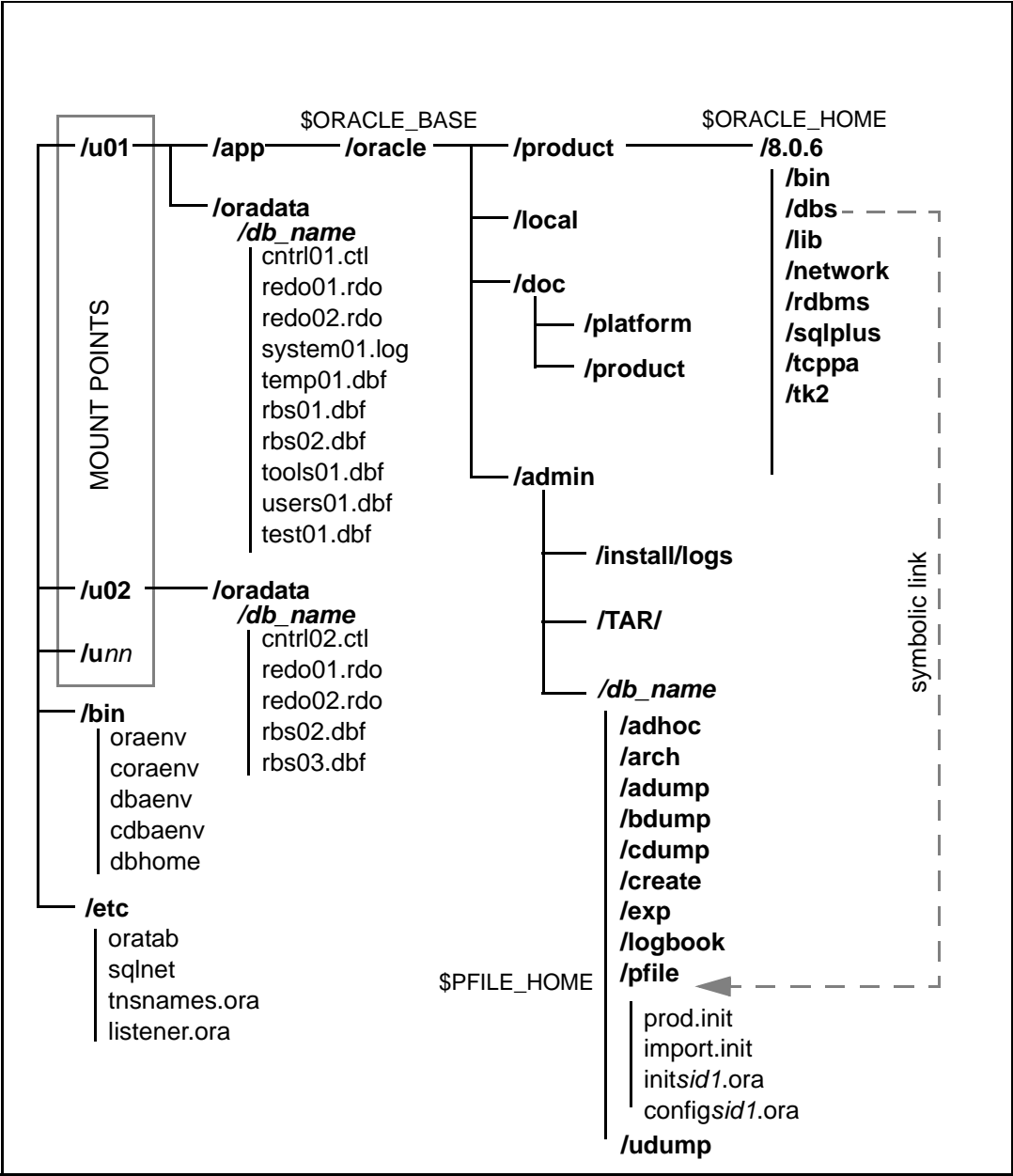
**Table 1–16 Filename Extensions**

.lis	output of SQL*Plus scripts
.log	installation log files; Oracle Server redo log files
.map	Installer product component files
.mk	make files
.msb	NLS message file (binary)
.msg	NLS message file (text)
.o	object module
.ora	Oracle configuration files
.orc	installation prototype files
.pad	Pro*Ada source file
.pc	Pro*C source file
.pco	Pro*COBOL source file
.ppd	printer driver file
.pfo	Pro*FORTRAN source file
.prd	product registration template file (for orainst)
.res	Toolkit II resource file
.sh	Bourne shell script file
.sql	SQL* script files
.sys	Bourne shell script file
.tab	SQL* script file
.trc	trace files
.tut	Bourne shell script file
.us	orainst message file
.utd	Uniform Terminal Definitions
.vrf	Installer Dependencies Verification Script

## Default OFA Database

An OFA default database created using the Oracle Installer is shown in Figure 1–1.

Figure 1–1 Default Oracle Installation



---

# Administering Oracle8 on DYNIX/ptx

- » Customizing the initsid.ora File
- » Setting the Environment
- » Environment Variables for Oracle8
- » Estimating Oracle8 Server Memory Usage
- » Calculating Cluster Size and Index Size
- » Initialization Parameters
- » Controlling the System Global Area
- » Managing Special Accounts and Groups
- » Managing Security
- » Administering Login Home Directories
- » Building and Running Demonstrations
- » Oracle8 Server SQL Reference

## Customizing the initsid.ora File

This section documents the default `initsid.ora` file provided with the Oracle8 distribution. The Oracle Installer creates it in the `$ORACLE_BASE/admin/db_name/pfile` directory. You can modify it to customize your Oracle8 installation.

Some `initsid.ora` parameter settings are generic to any size installation. For those parameter settings requiring different values for different size installations, three scenarios are provided: small, medium, and large. In the sample `initsid.ora` file, parameters dependent on installation size are shown for each setting. You can comment out settings that do not apply to your installation by inserting a number sign (#) at the beginning of a line.

Table 2–1 suggests the approximate SGA sizes corresponding to the three scenarios provided for in the `initsid.ora` file.

**Table 2–1 Block and SGA Sizes for Sample initsid.ora File**

Installation/Database Size			
Block Size	Small	Medium	Large
2 KB	4500 KB	6800 KB	17000 KB
4 KB	5500 KB	8800 KB	21000 KB

### Sample initsid.ora File

This file is provided by Oracle Corporation to assist in customizing the RDBMS installation. Some parameter settings are generic to any size installation. For parameters that require different values in different size installations, three scenarios are provided: SMALL, MEDIUM and LARGE. Any parameter that needs to be tuned according to installation size will have three settings, each one commented according to installation size.

```
# replace DEFAULT with your database name
db_name=DEFAULT

db_files = 80                                # SMALL
# db_files = 400                             # MEDIUM
# db_files = 1000                            # LARGE

db_file_multiblock_read_count = 8            # SMALL
# db_file_multiblock_read_count = 16         # MEDIUM
# db_file_multiblock_read_count = 32         # LARGE

db_block_buffers = 100                       # SMALL
```

```
# db_block_buffers = 550                                # MEDIUM
# db_block_buffers = 3200                                # LARGE

shared_pool_size = 3500000                               # SMALL
# shared_pool_size = 5000000                             # MEDIUM
# shared_pool_size = 9000000                             # LARGE

log_checkpoint_interval = 10000

processes = 50                                           # SMALL
# processes = 100                                         # MEDIUM
# processes = 200                                         # LARGE

parallel_max_servers = 5                                # SMALL
# parallel_max_servers = 4 x (number of CPUs)             # MEDIUM
# parallel_max_servers = 4 x (number of CPUs)             # LARGE

log_buffer = 8192                                        # SMALL
# log_buffer = 32768                                     # MEDIUM
# log_buffer = 163840                                    # LARGE

sequence_cache_entries = 10                             # SMALL
# sequence_cache_entries = 30                             # MEDIUM
# sequence_cache_entries = 100                           # LARGE

sequence_cache_hash_buckets = 10                        # SMALL
# sequence_cache_hash_buckets = 23                       # MEDIUM
# sequence_cache_hash_buckets = 89                       # LARGE

# audit_trail = true                                     # if you want auditing
# timed_statistics = true                               # if you want timed statistics
max_dump_file_size = 10240                               # limit trace file size to 5 Meg each

# Uncommenting the line below will cause automatic archiving if archiving has
# been enabled using ALTER DATABASE ARCHIVELOG.
# log_archive_start = true
# log_archive_dest = disk${rdbsms:[oracle.archive]}
# log_archive_format = "T%TS%S.ARC"

# If using private rollback segments, place lines of the following
# form in each of your instance-specific init.ora files:
# rollback_segments = (name1, name2)

# If using public rollback segments, define how many
# rollback segments each instance will pick up, using the formula
```

```
# # of rollback segments = transactions / transactions_per_rollback_segment
# In this example each instance will grab 40/10 = 4:
# transactions = 40
# transactions_per_rollback_segment = 10

# Global Naming -- enforce that a dblink has same name as the db it connects to
global_names = TRUE

# Edit and uncomment the following line to provide the suffix that will be
# appended to the db_name parameter (separated with a dot) and stored as the
# global database name when a database is created. If your site uses
# Internet Domain names for e-mail, then the part of your e-mail address after
# the '@' is a good candidate for this parameter value.

# db_domain = us.acme.com # global database name is db_name.db_domain

#_db_block_cache_protect = true # memory protect buffers
#event = "10210 trace name context forever, level 2" # data block checking
#event = "10211 trace name context forever, level 2" # index block checking
#event = "10235 trace name context forever, level 1" # memory heap checking
#event = "10049 trace name context forever, level 2" # memory protect cursors

# define parallel server (multi-instance) parameters
#ifile = ora_system:inits.ora

# define two control files by default
control_files = (ora_control1, ora_control2)

# Uncomment the following line if you wish to enable the Oracle Trace product
# to trace server activity. This enables scheduling of server collections
# from the Oracle Enterprise Manager Console.
# Also, if the oracle_trace_collection_name parameter is non-null,
# every session will write to the named collection, as well as enabling you
# to schedule future collections from the console.

# oracle_trace_enable = TRUE
```

## Setting the Environment

### Displaying and Setting Environment Variables

To display the value of an environment variable, use the echo command. For example, to display the value of ORACLE\_SID, enter:

```
$ echo $ORACLE_SID
```

## Setting and Exporting the Value of a Variable in a Current Session

For the Bourne or Korn shell, enter:

```
$ ORACLE_SID=test  
$ export ORACLE_SID
```

For the C shell, enter:

```
% setenv ORACLE_SID test
```

where `test` is the value of the variable `ORACLE_SID`.

## Setting a Common Environment

Oracle8 allows a DBA to set a common environment for all users. A common environment makes it easier for system administrators and database administrators to make changes to the physical Oracle Server system.

### The oraenv Command File

The `oraenv` (`coraenv` for the C shell) command file is created during installation. It contains values for Oracle environment variables and provides:

- a central means of updating all user accounts with database changes
- a mechanism for switching back and forth between Oracle Server databases

For example, a database needs to move from `/usr/oracle` to `/usr1/oracle`. Without a common environment-setting routine, user startup files would need to be updated individually. With `oraenv`, each user profile calls the `oraenv` command file and the changes must be made only to that file.

### Local bin Directory

Placing `oraenv` (or `coraenv`) and `dbhome` in the local `bin` directory, separate from the Oracle software home directory, ensures that these files are accessible to all users. It also ensures that `oraenv` (`coraenv`) continues to work even if you change the path to point to a different `ORACLE_HOME`.

### Moving Between Databases

To switch from one database or instance to another, call the `oraenv` routine, and reply to the prompt with the *sid* of the desired database. Always provide the full path of the `oraenv` command file. For example:

```
$ . /usr/local/bin/oraenv
ORACLE_SID= [default]? sid
```

## Database Examples

In the following examples, it is assumed your local `bin` directory is called `/usr/local/bin` and your production database is called `PROD`. If you prefer not to be prompted for the `ORACLE_SID` at startup, set the `ORAENV_ASK` environment variable to `No`.

In the examples below, `ORAENV_ASK` is reset to the default, `Yes`, after `oraenv` is executed. This ensures that the system prompts for a different `ORACLE_SID` the next time `oraenv` is executed.

### Single Instance

For the Bourne or Korn shell, add or replace the following line in the `.profile` file:

```
. local_bin_directory/oraenv
```

with the following lines:

```
PATH=${PATH}:/usr/local/bin
ORACLE_SID=PROD
export PATH ORACLE_SID
ORAENV_ASK=NO
. oraenv
ORAENV_ASK=
```

For the C shell, add or replace the following line in the `.cshrc` file:

```
source local_bin_directory/coraenv
```

with the following lines:



```
setenv PATH ${PATH}:/usr/local/bin
setenv ORACLE_SID PROD
set ORAENV_ASK = NO
source /usr/local/bin/coraenv
unset ORAENV_ASK
```

## Multiple Instances

For multiple instances, define the *sid* at startup.

For the Bourne or Korn shell:

```
PATH=${PATH}:/usr/local/bin
ORACLE_SID=PROD
export PATH ORACLE_SID
SIDLIST= `awk -F: '/^[^#]/ {printf "%s ", $1}' /etc/oratab`
echo "SIDS on this machine are $SIDLIST"
ORAENV_ASK=
oraenv
```

For the C shell:

```
setenv PATH ${PATH}:/usr/local/bin
setenv ORACLE_SID PROD
set sidlist = `awk -F: '/^[^#]/ {printf "%s ", $1}' /etc/oratab`
echo "SIDS on this machine are $sidlist"
unset ORAENV_ASK
source /usr/local/bin/coraenv
```

## Environment Variables for Oracle8

Certain variables in the UNIX environment must be set prior to installation of the Oracle system.

---

---

**See Also:** *Oracle8 Installation Guide for IBM DYNIX/ptx.*

---

---

## Oracle Environment Variables on UNIX

Table 2–2 provides the syntax and examples for Oracle8 variables.

Table 2–2 Oracle8 Environment Variables on UNIX

Variable	Detail	Definition
EPC_DISABLED	Function	Disables Oracle TRACE
	Syntax	true or false
	Example	true
NLS_LANG	Function	Specifies the language and character set used for output. See the <i>Oracle8 Installation Guide for IBM DYNIX/ptx</i> for a range of values.
	Syntax	language_territory.characterset
	Example	french_france.we8dec
ORA_NLS33	Function	Points to the directory where languages and character sets are stored.
	Set to	\$ORACLE_HOME/ocommon/nls/admin/data
ORACLE_BASE	Function	Specifies the base of the Oracle directory structure for OFA-compliant databases.
	Syntax	directory_path
	Example	/mount_point/app/oracle
ORACLE_HELP	Function	Specifies the directory containing help files.
	Syntax	directory_path
	Example	\$ORACLE_HOME/help/admin/resource
ORACLE_HOME	Function	Specifies the directory containing the Oracle software distribution.
	Syntax	directory_path
	Example	/mount_point/app/oracle/product/release_number
ORACLE_PATH	Function	Specifies the search pathname for files used by Oracle applications, such as SQL*Plus. If not specified, the application reads from and writes to the current directory.
	Syntax	colon-separated list of directories <i>directory:directory:directory</i>
	Example	/u01/oracle/adhoc/sqlplus: <b>Note:</b> The period adds the current working directory to the search path.

**Table 2–2 Oracle8 Environment Variables on UNIX**

Variable	Detail	Definition
ORACLE_SID	Function	Specifies the Oracle System Identifier.
	Syntax	The string of numbers and characters must begin with a letter. For more information, see the <i>Oracle8 Installation Guide for IBM DYNIX/ptx</i> .
	Example	SAL1
ORACLE_TERM	Function	Specifies the terminal type identifier. Used by the Installer and Oracle products to determine the correct Toolkit II (.res) resource file. If not set, the value of the operating-system variable TERM is used.
	Syntax	string of characters
	Range of Values	The value of this variable must be set such that the pattern <code>tk2c\${ORACLE_TERM}.res</code> corresponds to valid resource files in the Toolkit II resource directory or directories. See the <i>Oracle8 Installation Guide for IBM DYNIX/ptx</i> for a list of valid values.
	Example	vt100
ORACLE_TERMINAL	Function	Specifies an additional directory to search for Toolkit II (.res) resource files.
	Syntax	directory_name
	Example	<code>\$ORACLE_HOME/guicommon/tk21/admin/terminal</code>
ORACLE_TRACE	Function	Turns on tracing of Bourne shell scripts during install. If set to T, many Oracle shell scripts run with <code>set -x</code> flag on.
	Range of Values	T or anything else.
ORAENV_ASK	Function	Controls whether (c)oraenv prompts for ORACLE_SID or ORACLE_HOME. If set to NO (c)oraenv does not prompt and, if set to anything else, it does.
	Syntax	string
	Range of Values	NO or anything else.

Table 2–2 Oracle8 Environment Variables on UNIX

Variable	Detail	Definition
TNS_ADMIN	Function	Sets the directory containing the Oracle Net8 configuration files.
	Syntax	directory_path
	Range of Values	Any directory; for more information, see the <i>Oracle8 Installation Guide for IBM DYNIX/ptx</i> .
	Example	\$ORACLE_HOME/network/admin
TWO_TASK	Function	Sets the default Oracle Net8 connect string descriptor alias defined in the tnsnames.ora file.
	Syntax	available network alias
	Range of Values	Any valid Oracle Net8 alias defined in the tnsnames.ora file.
	Example	PRODDB_TCP

**Note:** Environment variables should not be defined with names that are identical to names of Oracle Server processes, for example: arch, pmon, and dbwr.

Abbreviations for ORACLE\_HOME and ORACLE\_SID

In Oracle8 Server files and programs, a question mark (?) represents the value of ORACLE\_HOME. For example, Oracle8 expands the question mark in the following SQL statement to the full pathname of ORACLE\_HOME:

```
alter tablespace TEMP add datafile '?/dbs/dbs2.ora' size 2M
```

The @ sign represents \$ORACLE\_SID. For example, to indicate that a file belongs to an instance, enter:

```
alter tablespace tablespace_name add datafile 'dbsfile@.ora'
```

## UNIX Environment Variables Used with Oracle8

Table 2–3 provides the syntax and examples for UNIX environment variables used with Oracle8.

**Table 2–3 UNIX Environment Variables Used with Oracle8**

Variable	Detail	Definition
DISPLAY	Function	Used by X-based tools. Specifies the display device used for input and output. See vendor's X Windows documentation for details.
	Syntax	hostname:display Hostname is the network identifier for the display device; display is a number which is almost always 0.
	Example	135.287.222.12:0 bambi:0
HOME	Function	The user's home directory.
LANG or LANGUAGE	Function	Specifies the language and character set used by the operating system for messages and other output. See the operating system documentation, and your <i>Oracle8 Installation Guide for IBM DYNIX/ptx</i> .
LDOPTS	Function	Specifies the default linker options on some platforms. See man pages on <code>ld</code> for details.
LPDEST	Function	Specifies the user's default printer for System V-based systems.
	Syntax	printer_name
	Example	docqms
LDPATH	Function	Default directories used by the linker to find shared object libraries. See man pages on <code>ld</code> for details.
LD_LIBRARY_PATH	Function	Used on some platforms by the shared library loader at runtime to find shared object libraries. See man pages on <code>ld</code> and <code>ldlopen</code> for details.
	Syntax	colon-separated list of directories <i>directory:directory:directory</i>
	Example	<code>/lib:\$ORACLE_HOME/lib</code>
PATH	Function	Used by the shell to locate executable programs; needs to include <code>\$ORACLE_HOME/bin</code> .
	Syntax	colon-separated list of directories <i>directory:directory:directory</i>

**Table 2–3** *UNIX Environment Variables Used with Oracle8*

Variable	Detail	Definition
PRINTER	Example	/bin:/usr/bin:/usr/local/bin: /usr/bin/X11:\$ORACLE_HOME/bin:\$HOME/bin. <b>Note:</b> The period adds the current working directory to the search path.
	Function	Selects the user's default printer for IBM DYNIX/ptx systems.
	Syntax	printer_name
SHELL	Example	docqms
	Function	Specifies the command interpreter used during a host command.
	Syntax	shell pathname
TERM	Range of Values	/bin/sh or /bin/csh or /bin/ksh or any other command interpreter supplied with IBM DYNIX/ptx
	Example	/bin/sh
	Function	Used by Oracle Toolkit II character mode tools to determine terminal types; also used by other UNIX tools for the same purpose.
TMPDIR	Example	vt100
	Function	Specifies the default directory for temporary disk files; if set, tools that create temporary files do so in this directory.
	Syntax	directory_path
XENVIRONMENT	Example	/u02/oracle/tmp
	Function	Specifies a file containing X Windows system resource definitions. See your X Windows documentation for more information.

## Setting the System Time

The TZ variable sets your time zone. Check your DYNIX/ptx documentation to see if your operating system uses this environment variable.

It allows a user to adjust the clock for daylight saving time changes, or different time zones. The adjusted time is used to time-stamp files, produce the output of the `date` command, and obtain the current SYSDATE.

---

---

**WARNING:** Users are discouraged from changing their personal TZ value. Using different values of TZ such as GMT+24 may change the day a transaction is recorded. This affects Oracle applications that use SYSDATE, such as Oracle Financials. Use sequence numbers to order a table instead of date columns to avoid this problem.

---

---

## Estimating Oracle8 Server Memory Usage

Before starting the Oracle8 Server, virtual memory requirements can be estimated using this formula:

$$\begin{aligned}
 & \text{<size of the oracle executable text>} \\
 + & \text{<size of the SGA>} \\
 + & n * ( \text{<size of tool executables private data section>} \\
 & \quad + \text{<size of oracle executables uninitialized data section>} \\
 & \quad + \text{<8192 bytes for the stack>} \\
 & \quad + \text{<2048 bytes for the processes user area>} )
 \end{aligned}$$

where  $n$  = number of background processes.

For each Oracle back-end connection, use the following formula to estimate virtual memory requirements:

$$\begin{aligned}
 & \text{<size of oracle executable data section>} \\
 + & \text{<size of oracle executables uninitialized data section>} \\
 + & \text{<8192 bytes for the stack>} \\
 + & \text{<2048 bytes for processes user area>} \\
 + & \text{<cursor area needed for the application>}
 \end{aligned}$$

Use the `size` command to estimate an executable's text size, private data section size, and uninitialized data section size (or *bss*). Program text is only counted once, no matter how many times the program is invoked, because all Oracle executable text is always shared.

To compute actual Oracle physical memory usage while the database is up and users are connecting to it, use the `ps` command. Look for all the front end, server, and background Oracle process entries. For each entry, add the “real size of process” columns for the resident memory use subtotal. Now add the text size for the Oracle executable and every other Oracle tool executable running on the system to that subtotal. Remember to count executable sizes only once, regardless of how many times the executable was invoked.

---

**See Also:** Refer to your DYNIX/ptx man pages or documentation for a list of available switches for the `ps` command.

---

## Calculating Cluster Size and Index Size

### Calculating Cluster Size

Use size guidelines in Table 2–4 to calculate cluster size using the formula in Appendix A of the *Oracle8 Administrator's Guide*.

**Table 2–4 Cluster Size Values**

Type	Size
Fixed header size	68 bytes
Variable transaction header	24* <i>INITRANS</i> value for the table
Row directory	4 bytes per row of a clustered table

### Calculating Index Size

Use Table 2–5 to calculate the size required by an index using the formula in Appendix A of the *Oracle8 Administrator's Guide*.

**Table 2–5 Index Size Values**

Type	Size
Fixed header size	113 bytes
Variable transaction header	24* <i>INITRANS</i> value for the index
Entry header	5 bytes



## Server Resource Limits

DYNIX/ptx inherits resource limits from the parent process (see `getrlimit(2)` in your operating system documentation). These limits apply to the Oracle8 Server shadow process that executes for user processes. The DYNIX/ptx default resource limits are high enough for any Oracle8 Server shadow or background process. However, if these limits are lowered, the Oracle8 Server system could be affected. Discuss this with your DYNIX/ptx system manager.

Disk quotas established for the Oracle `dba` user ID may hinder the operation of the Oracle8 system. Confer with your Oracle8 database administrator and the DYNIX/ptx system manager before establishing disk quotas.

## Initialization Parameters

Initialization parameters can be modified in the `init.ora` file for the Oracle8 Server instance.

---

---

**See Also:** *Oracle8 Administrator's Guide.*

---

---

## Default Initialization Parameter Values

Table 2–6 lists default initialization parameter values on DYNIX/ptx. All Oracle8 Server instances assume these values if you do not specify different values for them in the `init.ora` file. Oracle Corporation recommends that you include in the `init.ora` file only those parameters that differ from the default initialization parameter values.

To display the current values of these parameters on the system, use Server Manager to execute the SQL statement `SHOW PARAMETERS`.

**See Also:** *Oracle8 Server Reference.*

**Table 2–6   Default Initialization Parameters**

Parameter	Default Value
BACKGROUND_DUMP_DEST	\$ORACLE_HOME/rdbms/log
BITMAP_MERGE_AREA_SIZE	1048576
COMMIT_POINT_STRENGTH	1
CONTROL_FILES	\$ORACLE_HOME/dbs/ctrl@.dbf (where @ represents ORACLE_SID)
CREATE_BITMAP_AREA_SIZE	8388608
DB_BLOCK_BUFFERS	50
DB_BLOCK_SIZE	4096
DB_FILES	62 (maximum of 2000000)
DB_FILE_DIRECT_IO_COUNT	64 (maximum of 1048576)
DB_FILE_MULTIBLOCK_READ_COUNT	8 (range of 1-128, but should not exceed one quarter of DB_BLOCK_BUFFERS)
DISTRIBUTED_TRANSACTIONS	8
HASH_AREA_SIZE	0
HASH_MULTIBLOCK_IO_COUNT	1
LOCK_SGA	TRUE
LOCK_SGA_AREAS	0
LOG_ARCHIVE_BUFFER_SIZE	127
LOG_ARCHIVE_BUFFERS	4 (maximum of 128)
LOG_ARCHIVE_DEST	\$ORACLE_HOME/dbs/arch/
LOG_ARCHIVE_FORMAT	“%t%s.dbf”
LOG_BUFFER	393216
LOG_CHECKPOINT_INTERVAL	4294967294
LOG_SMALL_ENTRY_MAX_SIZE	80
MTS_MAX_DISPATCHERS	5
MTS_MAX_SERVERS	20
MTS_SERVERS	0

**Table 2–6 Default Initialization Parameters**

Parameter	Default Value
MTS_LISTENER_ADDRESS	ADDRESS=address (See Chapter 6)
NLS_LANGUAGE	AMERICAN
NLS_TERRITORY	AMERICA
OBJECT_CACHE_MAX_SIZE_PERCENT	10
OBJECT_CACHE_OPTIMAL_SIZE	102400
OPEN_CURSORS	50
OS_AUTHENT_PREFIX	ops\$
PROCESSES	25
SHARED_POOL_SIZE	3500000
SORT_AREA_SIZE	65536
SORT_READ_FAC	5
SORT_SPACEMAP_SIZE	512
USER_DUMP_DEST	\$ORACLE_HOME/rdbms/log

## Obsolete Oracle7 *init<sub>sid</sub>.ora* Parameters

The following Oracle7 DYNIX/ptx-specific *init<sub>sid</sub>.ora* parameters are no longer valid in Oracle8.

**Table 2–7    Obsolete *init<sub>sid</sub>.ora* Parameters**

Obsolete Parameter	Explanation
ASYNCH_WRITE	This parameter has been replaced by a generic parameter, <code>DISK_ASYNC_IO</code> . The default setting for this parameter is <code>TRUE</code> , resulting in Oracle8 automatically using Asynchronous I/O on all datafiles located on raw devices.
USE_READV	I/O operations now use vectored I/O automatically for scattered reads and gathered writes.
IO_TIMEOUT	I/O operations are now truly asynchronous within Oracle8 and timeouts are no longer used or needed.
_DIRECT_READ	Oracle8 now dynamically determines whether it is more efficient to use UNIX buffered reads for long contiguous scans or DYNIX/ptx's Direct I/O for shorter, disparate reads.
_LOCK_SGA	This parameter has been replaced by the generic parameter <code>LOCK_SGA</code> , which defaults to <code>TRUE</code> on DYNIX/ptx Systems.
_SPIN_CPU_YIELD_FREQ	Oracle8 now automatically calculates a reasonable frequency to yield the processor based on the value specified by the <code>SPIN_COUNT</code> parameter.
_NO_PREEMPT	Oracle8 now automatically asks the operating system not to preempt a process while holding a latch, if the operating system is configured to support it. To use this feature, you need to change a DYNIX/ptx kernel source file and recompile the DYNIX/ptx kernel. Change line 235 in <code>/usr/conf/uts/kernel/i386_space/param_space.c</code> from:  <code>bool_t root_nopreempt=1; /*must be root to do NOPREEMPT*/</code>  to:  <code>bool_t root_nopreempt=0; /*must be root to do NOPREEMPT*/</code>

## Controlling the System Global Area

The System Global Area (SGA) is the Oracle structure that resides in shared memory. It contains static data structures, locks, and data buffers. Sufficient shared memory must be available to each `oracle` process to address the entire SGA.

### Size limits of the SGA

The maximum size of a single shared memory region is specified by the IBM DYNIX/ptx parameter `SHMMAX`. An SGA that is 2048 KB can use four shared memory regions of 512 KB each.

If the size of the SGA exceeds the maximum size of a shared memory segment (`SHMMAX`), Oracle8 attempts to attach more contiguous segments to fulfill the requested SGA size. `SHMSEG` is the maximum number of segments that can be attached by a process. To attach the segments at contiguous addresses, `SHMMAX` must be set to its maximum value on systems where its size is limited.

The following `init.ora` parameters control the size of the SGA:

- n `DB_BLOCK_BUFFERS`
- n `DB_BLOCK_SIZE`
- n `SORT_AREA_SIZE`
- n `SHARED_POOL_SIZE`

Use caution when setting values for these parameters. When values are set too high, too much of the machine's physical memory is devoted to shared memory resulting in poor performance. As a guideline, the total of all instance's SGA sizes should be no more than one-third of the total physical RAM.

### Calculating the Size of the SGA

The approximate size of an instance's SGA can be calculated with this formula:

$$\begin{aligned} & (\text{DB\_BLOCK\_BUFFERS} \times \text{DB\_BLOCK\_SIZE}) \\ & + \text{SORT\_AREA\_SIZE} \\ & + \text{SHARED\_POOL\_SIZE} \\ & + 1\text{MB} \end{aligned}$$

To display the size of the SGA for a running database in bytes, use the Server manager `show sga` command. This command displays the size of the SGA in bytes.

## Relocating the SGA

The address at which the SGA is attached affects the amount of virtual address space available for such things as database buffers in the SGA and cursors in the user's application data area.

1. Determine the valid virtual address range for attaching shared memory segments (in the resulting `tstshm` display, the lines "Lowest shared memory address" and "Highest shared memory address" indicate the valid range):

```
$ tstshm
```

2. Check the "Segment boundaries" output of `tstshm` to determine the valid virtual address boundaries at which a shared memory segment can be attached.
3. Determine the size of your SGA. SGA size is displayed next to the heading Total System Global Area when your database system starts.
4. Change to the `$ORACLE_HOME/rdbms/lib` directory, and run `genksms` to generate the file `ksms.s`:

```
$ cd $ORACLE_HOME/rdbms/lib
$ $ORACLE_HOME/bin/genksms -b sgabeg > ksms.s
```

where *sgabeg* is the starting address of the SGA (which defaults to 0x20000000), and should fall within the range determined in step 2.

5. Change the value of *sgabeg* in the `ksms.s` file.
6. Shut down the existing Oracle database.
7. Rebuild the `oracle` executable in the `$ORACLE_HOME/rdbms/lib` directory:

```
$ make -f ins_rdbms.mk ioracle
```

Using `ioracle`:

- Backs up the old executable (`oracle0`)
- Assigns the correct privileges to the new `oracle` executable
- Moves the new executable into the `$ORACLE_HOME/bin` directory

The result is a new Oracle kernel that loads the SGA at the address specified by `sgabeg`.

## Managing Special Accounts and Groups

The DBA should be familiar with special accounts required by the Oracle Server, and should make sure these accounts belong to the appropriate groups. The following section describes special user accounts. UNIX accounts are described in Table 2–8, Oracle server accounts are described in Table 2–9. Special group accounts are described in Table 2–10.

**Table 2–8 UNIX Accounts**

<code>oracle</code>	The <i>oracle</i> software owner represents the account that owns the Oracle8 software. This maintenance account requires DBA privileges in order to CREATE, STARTUP, SHUTDOWN, and CONNECT as INTERNAL to the database. The <i>oracle</i> software owner must never be the superuser.
<code>root</code>	The <code>root</code> user is a special UNIX account with maximum privileges (called superuser privileges). This account is used to configure the UNIX kernel, configure and install networking software, and create user accounts and groups.

**Table 2–9 Oracle Server Accounts**

<code>SYS</code>	This is a standard Oracle8 account with DBA privileges automatically created during installation. The <code>SYS</code> account owns all the base tables for the data dictionary. This account is used by the DBA.
<code>SYSTEM</code>	This account is also a standard Oracle8 account, with DBA privileges automatically created during installation. Additional tables or views can be created by the <code>SYSTEM</code> user. DBAs may log in as <code>SYSTEM</code> to monitor or maintain databases.

**Table 2–10    Special Group Accounts**

dba group	The <i>oracle</i> software owner is the only required member of the <code>dba</code> group. You can add the <code>root</code> user, or any other UNIX user, to the <code>dba</code> group. Members of this group have access to Server Manager specially privileged functions. If your account is not a member of the <code>dba</code> group, you must enter a password in order to connect as <code>INTERNAL</code> or gain access to the other administrative functions of Server Manager. The default group ID is <code>dba</code> .
oper group	This is an optional UNIX group you can create. Members have database <code>OPERATOR</code> privileges. <code>OPERATOR</code> privileges are a restricted set of <code>dba</code> privileges.
root group	Only the <code>root</code> user should be a member of the <code>root</code> group.

## Managing Security

Oracle8 uses several features of the UNIX operating system to provide a secure environment for users. These features include file ownership, group accounts, and the ability of a program to change its user ID upon execution.

The two-task architecture of Oracle8 improves security by dividing work (and address space) between the user program and the `oracle` program. All database access is achieved through the shadow process and special authorizations on the `oracle` program.

## Groups and Security

To ensure greater security on an Oracle8 database, create user groups at the operating system level. Groups are controlled by the UNIX file `/etc/group`. Oracle programs are divided into two sets for security purposes: those executable by all (*other*, in UNIX terms), and those executable by DBAs only. A recommended approach to security is:

- Before installing the Oracle Server, create a database administrators' group (`dba`) and assign the `root` and *oracle* software owner IDs to this group. Programs executable by `dba` only have permission `710`. Server Manager system-privileged commands are assigned automatically to the `dba` group upon installation.



- Add an `oracle` group of authorized users to allow a subset of UNIX users limited access to Oracle8. Give Oracle utilities the `oracle` group ID. Publicly executable programs, such as `SQL*Plus`, should be executable by this group. Set the permissions on the utilities to 710 to grant execute permissions to this group, but not *other*.
- Grant permission 711 to programs executable by *other*. Restrict this permission to programs that do not affect database security.

Although you can assign any name to the database administrators' group, `dba` is the default group name, and the convention used in this document. If you change this group name, the Oracle Installer relinks the kernel automatically during Installation. If you have multiple databases with the same `ORACLE_HOME` (a configuration which Oracle Corporation *strongly* discourages), they should have the same database administrators' group. These restrictions do not apply to the group name for ordinary users (known as the `oracle` group).

---

---

**WARNING:** Even though both the *oracle* software owner and `root` user should belong to the `dba` group, the *oracle* software owner should *not* be a member of the `root` group. The `root` user should be the *only* member of the `root` group.

---

---

## Security for Oracle Server Utilities

Protect the Oracle8 executables from unauthorized use. The method you use depends on your environment and whether you use single-task utilities. These are suggestions for protecting Oracle8 executables:

- Keep all programs in the `$ORACLE_HOME/bin` directory and give ownership to the *oracle* software owner.
- Give all user utilities (`sqlplus`, `exp`, `imp`) a protection of 711 so all users on the machine can access the Oracle Server.
- Give all DBA utilities (such as Server Manager) a protection of 700 to restrict the use of these utilities to the DBA username, usually the *oracle* software owner.

## Security for Server Manager Commands

If you do not have SQL\*Plus, you can use Server Manager to make SQL queries. However, be careful how you assign access to Server Manager. The following system-privileged statements should not be accessible to anyone but the *oracle* software owner and the *dba* group users, as they grant special operating system privileges:

- STARTUP
- SHUTDOWN
- CONNECT INTERNAL

---

---

**WARNING:** System-privileged statements can damage your database if used incorrectly. Note that non-dba group users can connect as internal if they have the necessary password.

---

---

## Security for Database Files

The user ID used to install Oracle8 should own the database files. The default user ID is the *oracle* software owner. Set the authorizations on these files to permission 0600: read/write (rw) by owner only, with no write authorizations for group or other users.

The *oracle* software owner should own the directories containing the database files. For added security, revoke read permission from *group* and other users.

To access the protected database files, the *oracle* program must have its set user ID (setuid) bit on. To set this bit, enter:

```
$ chmod 6751 $ORACLE_HOME/bin/oracle
```

This sets the authorization for the *oracle* program to:

```
-rwsr-s--x 1 oracle dba 443578 Mar 10 23:03 oracle
```

## Setting the User ID

The Oracle Installer automatically sets the user ID. The *s* in the user execute field means when you execute the *oracle* program, it has an effective user ID of *oracle*, regardless of the actual user ID of the person invoking it.

## Network Security

### Using Passwords on the Network

Remote users on the network can enter their passwords in clear or encrypted text. When you use clear text, passwords can be picked up by unauthorized users, resulting in a breach of security. Oracle Net8 supports encrypted passwords.

### DBA Privileges Over the Network

To control DBA privileges over the network choose one of the following options:

- Set remote DBA access to denied in the `/etc/listener.ora` file
- Set a special password in `orapwd` for DBA privileges

### Automatic (ops\$) Logins

Oracle8 supports automatic logins (operating system authorized logins) over the network.

UNIX treats a dollar sign (\$) as the beginning of an environment variable. Therefore, when you specify an operating system authorized (ops\$) login on the command line or in a script, first escape the \$ with a backslash (\). For example, user ID `scott` should specify `ops\scott` when logging in remotely.

Automatic logins are not allowed for the `root` user ID.

---

---

**Note:** Automatic logins by PC, Apple MacIntosh, and OS/2 users are not secure. Anyone can edit the Oracle configuration file and change their user ID. For security reasons, if users of these systems are logging in over the network, Oracle Corporation strongly recommends you disable the ops\$ logins in the `listener.ora` file.

---

---

## Enabling Automatic Logins for Oracle Net8

Automatic and remote DBA logins are not controlled by Oracle Net8. They are controlled by the Oracle8 Server and configured using parameters in the `init.ora` file. Although automatic logins are supported, they are disabled by default. To enable them, set the `REMOTE_OS_AUTHENT` initialization parameter to `true`, then start up the database.

Because `oracle` controls these logins, it is not necessary to run the Oracle Net8 listener as `setuid` to `root`.

---

---

**See Also:** Configuring Oracle Net8 is described in Chapter 6.

---

---

To perform an automatic login with Oracle Net8, create a user called `daemon` in your `/etc/passwd` file. The `daemon` user must not have an `ops$` account in any of the local databases, nor be in any of the DBA groups. That is, there should be no `ops$daemon` account that would allow an outside user to intrude into your local database.

DBA Group ID Keywords

Table 2–11 describes the keywords used in the `/etc/listener.ora` file to enable and control remote logins:

Table 2–11    *Keywords Used to Control Remote Logins*

DBA_GROUP	Use this keyword if the name is constant for all instances serviced by the listener.
DBA_GROUP_sid	Use this keyword for each ORACLE_SID if the listener services more than one \$ORACLE_HOME, and the group IDs are different.
OPS_DOLLAR_LOGIN_ALLOWED OPS_DOLLAR_LOGIN_DENIED	Use these keywords to control remote login. OPS_DOLLAR_LOGIN_DENIED is the default.
REMOTE_DBA_OPS_ALLOWED REMOTE_DBA_OPS_DENIED	Use these keywords to control remote DBA access. REMOTE_DBA_OPS_DENIED is the default.

If the DBA group ID for the database accessed is not the default name (`dba`), you can specify a non-default name.

Set remote login and remote DBA access parameters to the individual `ORACLE_SIDs` of databases on the network, or specify all *sids* at once. For example, either of the following statements are valid:

```
PARAMETER=ALL_SIDS
PARAMETER=sid1[, sidn...]
```

To see which privileges are assigned to the *sids*, enter:

```
$ lsnrctl status
```

Checking Order

The system checks remote login parameters in the following order:

1. Parameters that deny access
2. Parameters that permit access
3. Default value (denied)

These privileges are implemented by manipulating the user ID and group ID of the shadow process forked by the Oracle Net8 listener. For example:

- If `OPS_DOLLAR_LOGIN_DENIED` is `true` for a particular instance, or if the user ID as reported by the client-side operating system has no account on the database host machine, the user ID and group ID are found in the `/etc/passwd` file under the entry for `daemon`.
- If both `OPS_DOLLAR_LOGIN_ALLOWED` and `REMOTE_DBA_OPS_ALLOWED` are `true` for a particular `ORACLE_SID`, and if the user ID as reported by the client operating system does have an account on this system, the user ID and group ID are found in `/etc/passwd` for this user ID.
- If `OPS_DOLLAR_LOGIN_ALLOWED` is `true` for a particular `ORACLE_SID`, but `REMOTE_DBA_OPS_ALLOWED` is `false`, then, if the user ID has DBA privileges, the process has the user ID and group ID of `daemon`. Otherwise, the process has the user ID and group ID of this user.

---

---

**Note:** `REMOTE_DBA_OPS_ALLOWED` is `false` by default. Oracle Corporation recommends that you do not change this value. When this parameter is set to `false`, users with DBA privileges cannot make operating system authorized logins over the network. They can, however, proceed with ordinary (password-protected) network logins.

---

---

## Security and Remote Passwords

You can access or administer a database from a remote machine, such as a personal computer, without operating system accounts. User validation is accomplished by using an Oracle8 password file, created and managed by the `orapwd` utility. You can also use password file validation on systems that support operating system accounts.

Local password files are in the `$ORACLE_HOME/dbs` directory and contain the username and password information for a single database. If there are multiple `$ORACLE_HOME` directories on a machine, each has a separate password file.

### Running orapwd

The `orapwd` utility exists in `$ORACLE_HOME/bin` and is run by the *oracle* software owner. Invoke `orapwd` by entering:

```
$ orapwd file=$ORACLE_HOME/dbs/orapwsid password=password entries=max_users
```

This syntax is described in Table 2–12.

**Table 2–12**    *Syntax for Executing orapwd*

file	is the name of the file where password information is written. The name of the file must be <code>orapwsid</code> , and you must supply the full pathname. Its contents are encrypted and not user-readable. This parameter is mandatory.
password	is the initial password you selected for INTERNAL and SYS. You can change this password after you create the database using an ALTER USER statement. This parameter is mandatory.
entries	is the maximum number of users allowed to connect to the database as SYSDBA or SYSOPER. This parameter is mandatory only if you want this password file to be EXCLUSIVE.

---

**Note:** You must create a new password file if you ever need to increase the maximum number of users. Therefore, set *max\_users* to a higher number than you expect to require.

---

## orapwd Example

```
$ orapwd file=/u01/app/oracle/product/8.0.6/dbs/orapwV806 \
password=manager entries=30
```

---

---

**See Also:** *Oracle8 Server Administrator's Guide.*

---

---

## Shared Password File for Multiple Databases

The default password file `/dbs/orapwd` should be used when the initialization parameter `REMOTE_LOGIN_PASSWORDFILE` is set to `SHARED` for multiple databases. There is no `sid` specific password file for multiple databases.

## Access to a Database from a Remote PC

When there is an Oracle8 password file, networked PC users can access this database as `INTERNAL`. Non-privileged users can connect to the database by invoking an Oracle application that uses the database. Privileged users who want to perform DBA functions on the database can enter the appropriate Server Manager command from their PC, adding the `dba` user password. For example:

```
SVRMGR> connect internal/dba_password
```

To connect as `OPERATOR`, use the same command with the `OPERATOR` password.

## Remote Authentication

The following `init.ora` parameters, shown in Table 2–13 control the behavior of remote connections through non-secure protocols:

**Table 2–13 Parameters For Controlling Remote Connections**

<code>REMOTE_OS_AUTHENT</code>	enables or disables <code>ops\$</code> connection
<code>OS_AUTHENT_PREFIX</code>	used by <code>ops\$</code> accounts
<code>REMOTE_OS_ROLES</code>	enables or disables roles through remote connections

---

---

**Note:** If `REMOTE_OS_AUTHENT` is set to `true`, users who are members of the `dba` group on the remote machine are able to connect as `INTERNAL` without a password.

---

---

### User-Visible Effects of the Shutdown Mechanism

Clients connected to an Oracle instance while a shutdown takes place will receive one of the following error messages upon subsequent SQL operations.

```
ORA-03113: end-of-file on communication channel
ORA-12571: TNS:packet writer failure
```

## Administering Login Home Directories

To add or move login home directories without modifying programs that refer to them, you must:

- Refer to explicit path names in files designed to store them, for example:  
`/etc/passwd` and `/etc/oratab`
- Refer to group memberships in the `/etc/group` file

It is not necessary to record a pathname except in a central reference file, because a user's home directory can be derived in either of the following ways:

- C shell and Korn shell users can use `~login` to refer to a user's home directory.
- Bourne shell users can construct a simple program to do this. See the sample `1hd` script later in this section.

Similarly, group memberships are computed from `/etc/group`. See the sample `grp` script later in this section.

---

**Note:** Local general-purpose utilities such as these should be stored in the `/usr/local/bin` directory.

---



## Sample lhd Script

```
#!/bin/sh
#
# lhd - print login home directory name for a given user
#
# SYNTAX
# lhd [login]
#
prog=`basename $0`
if [ $# -eq 0 ] ; then
    login=`whoami`

elif [ $# -eq 1 ] ; then
    login=$1
else
    echo "Usage: $prog login" >&2

    exit 2
fi
awk -F: ' $1==login {print $6}' login=$login /etc/passwd
```

## Sample grpx Script

```
#!/bin/sh
# grpx - print the list of users belonging to a given group
#
prog=`basename $0`
if [ $# -ne 1 ] ; then
    echo "Usage: $prog group" >&2
    exit 2
fi
g=$1
# calculate group id of g
gid=`awk -F: ' $1==g {print $3}' g=$g /etc/group`
# list users whose default group id is gid
u1=`awk -F: ' $4==gid {print $1}' gid=$gid /etc/passwd`
# list users who are recorded members of g
u2=`awk -F: ' $1==g {gsub(/,/," "); print $4}' g=$g /etc/group`
# remove duplicates from the union of the two lists
echo $u1 $u2 | tr " " "\012" | sort | uniq | tr "\012" " "
echo
```

**Example 2–1 Using *lhd* and *grp*x Scripts**

This example shows how the administrator can propagate a skeleton `.profile` file to the home directory for each member of a group. If the membership list of the `clerk` group changes, the code does not require modification.

```
$ for u in `grp x clerk` ; do
> cp /etc/skel/.profile `lhd $u`
> done
```

## Building and Running Demonstrations

### Loading PL/SQL Demonstrations

PL/SQL includes a number of sample programs you can load. Demonstration and message files are in the `rdbms` directory. Perform these steps with the Oracle8 Server open and mounted:

1. Invoke Server Manager and connect with the user/password `scott/tiger`:

```
$ cd $ORACLE_HOME/plsql/demo
$ svrmgrl
SVRMGR > connect scott/tiger
```

2. To load the demonstrations, invoke `exampbld.sql` from Server Manager:

```
SVRMGR > @exampbld
```

---

---

**Note:** Build the demonstrations under any Oracle account with sufficient permissions. Run the demonstrations under the same account you used to build them.

---

---

## Running PL/SQL Demonstrations

Table 2–14 lists the kernel demonstrations.

**Table 2–14 Kernel Demonstrations**

examp1.sql	examp5.sql	examp11.sql	sample1.sql
examp2.sql	examp6.sql	examp12.sql	sample2.sql
examp3.sql	examp7.sql	examp13.sql	sample3.sql
examp4.sql	examp8.sql	examp14.sql	sample4.sql
extproc.sql			

Table 2–15 lists the precompiler demonstrations.

**Table 2–15 Precompiler Demonstrations**

examp9.pc	examp10.pc	sample5.pc	sample6.pc
-----------	------------	------------	------------

To run the kernel PL/SQL demonstrations, invoke SQL\*Plus to connect to the kernel, using the same user/password you used to create the demonstrations. Start the demonstration by typing an “at” sign (@) or the word `start` before the demonstration name. For example, to start the `examp1` demonstration, enter:

```
$ sqlplus scott/tiger
SQL> @examp1
```

To build the precompiler PL/SQL demonstrations, enter:

```
$ cd $ORACLE_HOME/plsql/demo
$ make -f demo_plsql.mk demos
```

If you want to build a single demonstration, enter its name as the argument in the `make` command. For example, to make the `examp9.pc` executable, enter:

```
$ make -f demo_plsql.mk examp9
```

To start the `examp9` demonstration from your current shell, enter:

```
$ examp9
```

In order to run the `extproc` demo, you first have to add the following line to the file, `tnsnames.ora`:

```
(DESCRIPTION=(ADDRESS=(PROTOCOL=ipc)(KEY=plsf)))(CONNECT_DATA=(SID=extproc))
```

and the following line to the file, `listener.ora`:

```
SC=(SID_NAME=extproc)(ORACLE_HOME=/vobs/oracle)(PROGRAM=extproc)
```

then from your current `SQL*Plus` session, enter:

```
SQL> connect scott/tiger
Connected.
SQL> connect system/manager
Connected.
SQL> grant create library to scott;
Statement processed.
SQL> connect scott/tiger
Connected.
SQL> create library demolib as
'$ORACLE_HOME/plsql/demo/extproc.so';
Statement processed.
```

Then finally, to run the tests:

```
SQL> connect scott/tiger
Connected.
SQL> @extproc
```

## SQL\*Loader Demonstrations

SQL\*Loader demonstrations require that:

- User `scott/tiger` has `CONNECT` and `RESOURCE` privileges
- `EMP` and `DEPT` tables exist and are empty

To create and run a demonstration:

1. Connect to the database as the user/password `scott/tiger` from Server Manager (line mode).
2. Run the `ulcasen.sql` corresponding to the demonstration you want to run.
3. As `scott/tiger`, invoke the demonstration from the command line:  

```
$ sqlldr scott/tiger ulcasen
```

As `scott/tiger`, run the SQL\*Loader demonstrations in the following order:

- `n`    `ulcase1`: Follow steps 1 - 3 above.
- `n`    `ulcase3`: Follow steps 1 - 3 above.
- `n`    `ulcase4`: Follow steps 1 - 3 above.
- `n`    `ulcase5`: Run the `ulcase*.sql` script as `scott/tiger`, then enter the following at the command line:  
  
      `$ sqlldr scott/tiger ulcase*`
- `n`    `ulcase2`: Invoke the demonstration (you do not have to run the `ulcase2.sql` script).
- `n`    `ulcase6`: Run the `ulcase6.sql` script as `scott/tiger`, then enter the following at the command line:  
  
      `$ sqlldr scott/tiger ulcase1 DIRECT=true`
- `n`    `ulcase7`: Run the `ulcase6.sql` script as `scott/tiger`, then enter the following at the command line:  
  
      `$ sqlldr scott/tiger ulcase7`

## Administering SQL\*Loader

Oracle8 Server incorporates SQL\*Loader functionality. Demonstration and message files are in the `rdbms` directory.

### File Processing Option

The SQL\*Loader release control file includes the following additional file processing option strings, the default being `str`, which takes no argument:

```
[ "str" | "fix n" | "var n" ]
```

`str` (the default) specifies a stream of records, each terminated by a newline character, which are read in one record at a time.

`fix` indicates that the file consists of fixed-length records, each of which is *n* bytes long, where *n* is an integer value.

`var` indicates that the file consists of variable-length records, each of which is *n* bytes long, where *n* is an integer value specified in the first five characters of the record.

If the file processing options are not selected, the information is processed by default as a stream of records (`str`). You might find that `fix` mode yields faster performance than the default `str` mode because it does not need to scan for record terminators.

### Newlines in Fixed Length Records

When using the `fix` option to read a file containing fixed-length records, where each record is terminated by a newline, include the length of the newline (one character) when specifying the record length to SQL \*Loader.

For example, to read the following file:

```
AAA newline
BBB newline
CCC newline
```

specify `fix 4` instead of `fix 3` to account for the additional newline character.

If you do not terminate the last record in a file of fixed records with a newline character, do not terminate the other records with a newline character either. Similarly, if you terminate the last record with a newline, terminate all records with a newline.

---

---

**WARNING:** Certain text editors, such as `vi`, automatically terminate the last record of a file with a newline character. This leads to inconsistencies if the other records in the file are not terminated with newline characters.

---

---

### Removing Newlines

Use the `position(x:y)` function in the control file to discard the newlines from fixed length records rather than loading them. To do this, enter the following in your control file:

```
load data
infile xyz.dat "fix 4"
into table abc
( dept position(01:03) char )
```

When this is done, newlines are discarded because they are in the fourth position in each fixed-length record.

## Oracle Security Server

---

**See Also:** For information on the Oracle Security Server, see the *Oracle Security Server Guide*.

---

## Oracle8 Server SQL Reference

### CREATE CONTROLFILE Parameters

Use the parameter values in Table 2–16 to determine the size of control files for a database.

**Table 2–16** *Determining the Size of Control Files*

Parameter	Default Value	Maximum Value
MAXDATAFILES	30	65534
MAXINSTANCES	1	63
MAXLOGFILES	16	255
MAXLOGMEMBERS	2	5
MAXLOGHISTORY	100	65534





---

# Tuning Oracle8 on DYNIX/ptx

- „ The Importance of Tuning
- „ DYNIX/ptx Tools
- „ SQL Scripts
- „ Tuning Memory Management
- „ Tuning Disk I/O
- „ Monitoring Disk Performance
- „ Tuning CPU Usage
- „ Tuning Oracle Resource Contention
- „ Tuning Block Size and File Size
- „ Tuning the DYNIX/ptx Buffer Cache Size
- „ Tuning Resource Contention for Oracle Parallel Server
- „ Using Trace and Alert Files
- „ Raw Devices

## The Importance of Tuning

Oracle8 is a highly optimizable software product. Frequent tuning optimizes system performance and prevents data bottlenecks. Although this chapter is written from the perspective of single-processor systems, most of the performance tuning tips provided here are also valid when using the Oracle Parallel Server options.

### Before Tuning the System

Before tuning the system, observe its normal behavior using the DYNIX/ptx tools described in “DYNIX/ptx Tools” in the next section.

---

---

**See Also:** *Oracle8 Parallel Server Concepts and Administration.*  
*Oracle8 Tuning.*

---

---

## DYNIX/ptx Tools

DYNIX/ptx provides performance monitoring tools that can be used to assess database performance and determine database requirements.

In addition to providing statistics for `oracle` processes, these tools provide statistics for CPU usage, interrupts, swapping, paging, and context switching for the entire system.

---

---

**See Also:** DYNIX/ptx tools are described in the operating system documentation.

---

---

### sar

The `sar` command is used to monitor swapping, paging, disk, and CPU activity, depending on the switches you supply with the command.

The following statement displays a summary of paging activity ten times, at ten second intervals:

```
$ sar -p 10 10
```

Sample output from the `sar -p` command is shown in Table 3–1.

**Table 3–1 Output from the `sar -p` command**

	15:29:13	vflt/s	pflt/s	pgfil/s	rclm/s
	15:29:23	1846.30	0.00	0.00	0.00
	15:29:33	2516.77	0.00	0.20	0.00
	15:29:43	1457.10	0.00	0.00	0.00
	15:29:53	2318.36	0.00	0.00	0.00
	15:30:03	1566.43	0.00	0.40	0.00
	15:30:14	1519.58	0.00	45.25	0.00
	15:30:24	1435.20	0.00	0.00	0.00
	15:30:34	1520.38	0.00	0.00	0.00
	15:30:44	1806.49	0.00	0.00	0.00
	15:30:54	1727.05	0.00	0.40	0.00
	Average	1771.55	0.00	4.62	0.00

## swap

The `swap -l` utility reports information about swap space usage. A shortage of swap space can result in the system hanging and slow response time. Sample output from the `swap -l` command is shown in Table 3–2.

**Table 3–2 Output from the `swap -l` command**

swapfile	dev	swaplo	blocks	free
/dev/dsk/c0t3d0s1	32,25	8	197592	162136

Other DYNIX/ptx monitoring facilities available include:

- Monitor
- lmstat
- ptx/PEP

## SQL Scripts

### utlbbstat and utlestat SQL Scripts

The `utlbbstat` and `utlestat` SQL scripts are used to monitor Oracle database performance and tune the Shared Global Area (SGA) data structures. For information regarding these scripts, see *Oracle8 Server Tuning*. On DYNIX/ptx, the scripts are located in `$ORACLE_HOME/rdbms/admin/`.

## Tuning Memory Management

Start the memory tuning process by tuning paging and swapping space to determine how much memory is available.

The Oracle buffer manager ensures that the more frequently accessed data is cached longer. Monitoring the buffer manager and tuning the buffer cache can have a significant influence on Oracle performance. The optimal Oracle buffer size for your system depends on the overall system load and the relative priority of Oracle over other applications.

### Allocate Sufficient Swap Space

Swapping causes significant UNIX overhead and should be minimized. Use `sar -w` on DYNIX/ptx to check for swapping.

If your system is swapping and you need to conserve memory:

- Avoid running unnecessary system daemon processes or application processes
- Decrease the number of database buffers to free some memory
- Decrease the number of UNIX file buffers, especially if you are using raw devices

Procedures for adding swap space vary between UNIX implementations. On DYNIX/ptx use `swap -l` to determine how much swap space is currently in use. Use `swap -a` to add swap space to your system. Consult your IBM DYNIX/ptx documentation for further information

Start with swap space two to four times your system's random access memory (RAM). Use a higher value if you plan to use CASE, Oracle Applications, or Oracle Office. Monitor the use of swap space and increase it as necessary.

## Control Paging

Paging may not present as serious a problem as swapping, because an entire program does not have to reside in memory in order to run. A small number of page-outs may not noticeably affect the performance of your system.

To detect excessive paging, run measurements during periods of fast response or idle time to compare against measurements from periods of slow response.

Use `sar -p` to monitor paging. The following columns from `sar -p` output are important:

- `vflt/s` indicates the number of address translation page faults. Address translation faults occur when a process references a valid page not in memory.
- `rc1m/s` indicates the number of valid pages that have been reclaimed and added to the free list by page-out activity. This value should be zero.

If your system consistently has excessive page-out activity, consider the following solutions:

- Install more memory
- Move some of the work to another system
- Configure your kernel to use less memory

## Hold the SGA in a Single Shared Memory Segment

Although this performance gain is minor, you cannot start the database without configuring sufficient shared memory.

You may need to reconfigure the UNIX kernel to increase shared memory. The UNIX kernel parameters for shared memory include `SHMMAX`, `SHMMNI`, and `SHMSEG`. In order to ensure that the SGA resides in a single shared memory segment, set the value of `SHMAX` to 4294967295 (4 GB).

The size of the SGA can be estimated using the following steps:

1. Multiply `DB_BLOCK_BUFFERS` by `DB_BLOCK_SIZE`.
2. Add the result of Step 1 to `SORT_AREA_SIZE`.
3. Add the result of Step 2 to `SHARED_POOL_SIZE`.
4. Add the result of Step 3 to `LOG_BUFFER`.

You can also use the UNIX utility `ipcs` to monitor the status of shared memory.

---

---

**See Also:** “Configure UNIX Kernel for Oracle” in Chapter 2 of the *Oracle8 Installation Guide for IBM DYNIX/ptx*.

---

---

## Optimize Number of Database Buffers

Potential performance benefit: 0-200 percent.

The `DB_BLOCK_SIZE` parameter also determines the size of the database buffers in the SGA. The `DB_BLOCK_BUFFERS` parameter is the memory parameter with the most direct effect on system performance.

Use the System Statistics Monitor in Server Manager to check the *hit ratio*. The hit ratio for the buffer cache is defined as:

$$\text{Hit Ratio} = \frac{\text{Logical Reads} - \text{Physical Reads}}{\text{Logical Reads}}$$

where  $\text{Logical Reads} = db\ block\ gets + consistent\ gets$

If your hit ratio is less than 60 or 70 percent, increase the number of buffers in the cache by raising `DB_BLOCK_BUFFERS`.

---

---

**See Also:** Using the `X$KCBRBH` table as an alternate method for estimating the number of buffers based on statistics gathered from a running system is described in *Oracle8 Server Tuning*. The System Statistics Monitor is described in the *Oracle Server Manager User's Guide*.

---

---

## Use Indirect Database Buffers

The maximum SGA size supported by DYNIX/ptx is 3 GB if direct database buffers are used, but this can be increased to take advantage of large memory systems by using indirect buffers. DYNIX/ptx provides the ability to indirectly map blocks of memory, which can allow the database block buffer section of the SGA to grow up to a theoretical maximum of 31 GB.

Since enabling indirect database buffers involves a slight overhead of mapping and unmapping buffers as they are needed, this feature is disabled by default, and is only recommended when SGA sizes of 3 GB or greater are required for a higher buffer cache hit ratio.

To enable indirect database buffers, set the init.ora parameter `USE_INDIRECT_DATA_BUFFERS = true`. The SGA must be locked into physical memory (see page 4-10) in order to use this feature.

## Optimize Number of Redo Buffers

The redo log space statistic is the number of times a user process waits for space in the redo buffer. Use the Server Manager System Statistics display to monitor redo buffers.

The value in the Total column for redo log space requests should be near zero, or at least not increasing. A non-zero value indicates that processes are waiting for space in the buffer. In this case, consider increasing the size of the redo log buffer in increments of 5 percent.

The size of the redo log buffer is determined by the `init.ora` `LOG_BUFFER` parameter. The value of this parameter is expressed in bytes.

## Optimize the Shared Pool Size

The `init.ora` `SHARED_POOL_SIZE` parameter sets the size of the shared pool in bytes. A modified least-recently-used algorithm gives precedence to data dictionary cache entries. This means that tuning the library cache also ensures that enough memory is available for the data dictionary.

Use the `V$SGASTAT` table to monitor the shared pool, checking the free space in particular. Sample `V$SGASTAT` query:

```
SELECT * FROM v$sgastat ORDER BY bytes desc
NAME                                BYTES
-----
sql area                            1370876
free memory                          867036
library cache                        785224
db_block_buffers                     409600
dictionary cache                     275740
...
```

The shared pool is often set too large. If the free memory area is as large as the example above, reduce the size of the shared pool. Execute repeated queries to see if any of the values are increasing.



## Verify Data Dictionary Cache Effectiveness

For optimal performance when parsing SQL statements, the data dictionary cache must be large enough to hold the most frequently accessed data. Data dictionary cache misses generate recursive calls and degrade database performance.

In the Server Manager Statistic display, the Total column shows the number of recursive calls since you started up the database. If the Oracle8 Server does not continue making recursive calls after startup, your data dictionary cache is probably large enough for your dictionary data. If the number of recursive calls accumulates while your application is running, you may need to increase the size of the data dictionary cache.

---

---

**Note:** If your dictionary cache seems too small, query the V\$ROWCACHE table to check cache activity.

---

---

## Allocate Adequate Library Cache Space

Potential performance benefit: 0-50 percent.

The library cache contains shared SQL and PL/SQL areas. Even if SQL can be reused, it will not be if the library cache is too small. Find out if library cache misses are affecting performance by querying the V\$LIBRARYCACHE table.

Monitor the statistics in the V\$LIBRARYCACHE over a period of time with the following query:

```
SELECT SUM(pins) "Executions",
       SUM(reloads) "Cache Misses while Executing"
FROM V$LIBRARYCACHE;
```

The query returns output similar to the following:

```
Executions Cache Misses while Executing
-----
320871                               549
```

The sum of the PINS (first column) indicates that SQL statements, PL/SQL blocks, and object definitions were accessed for execution a total of 320,871 times. The sum of the RELOADS (last column) indicates that 549 of those executions resulted in library cache misses. Total reloads should be near zero, and the ratio should be below 1 percent.

If the ratio of RELOADS to PINS is greater than 1 percent, allocate additional memory to the library cache by increasing the `init.ora` parameter `SHARED_POOL_SIZE`.

## Lock Large PL/SQL Blocks into the Shared Pool

Occasionally, seldom-used shared objects should be locked into the shared pool. This often helps when the library cache latch is a bottleneck. Use the `dbms_shared_pool` utility package to determine the size of objects in the shared pool. See the documentation in the comments of the PL/SQL script `dbmspool.sql`, installed in the `$ORACLE_HOME/rdbms/admin` directory.

Use the procedure below to install PL/SQL, enable the server output buffer, and run the `sizes()` procedure:

1. Navigate to `$ORACLE_HOME/rdbms/admin`.
2. Invoke SQL\*Plus and execute the following sequence of commands:

```
@dbmspool
@prvtpool
set serveroutput on size xxx
begin
  sys.dbms_shared_pool.sizes (minsize);
end;
/
```

A setting of 20000 is sufficient for `xxx`, and 20 for `minsize`. After you identify the frequently used shared objects, you can run the procedure keeping `VARCHAR2`, flag `CHAR` `DEFAULT 'P'` as often as necessary to lock objects into the shared pool; `dbms_shared_pool.sizes` gives a list of objects in the shared pool larger than `minsize`.

For example, keep (`obj_name`, "P") pins `obj_name` in the shared pool.

The `unkeep` procedure unlocks objects.

---

---

**Suggestion:** Build a SQL\*Plus script to do this as part of database startup.

---

---

## Optimize the Session Cache Cursors

If you use Oracle Forms applications extensively (or other programs that close and reopen session cursors), set the Oracle Server to save session cursors in the library cache. This improves performance significantly.

Set the `SESSION_CACHED_CURSORS` initialization parameter to the maximum number of session cursors you want to cache. Monitor cache performance and adjust the cache size

based on the hit ratio. For example, the following SQL statement retrieves data for tuning caching session cursors:

```
> SELECT value, name
      FROM V$sysstat WHERE statistic# IN (122, 123)
VALUE      NAME
-----
12675 session cursor cache hits
12766 session cursor cache count
2 rows selected.
```

## Tuning Disk I/O

I/O bottlenecks are the easiest performance problems to identify. Balance I/O evenly across all available disks to reduce disk access times. For smaller databases and those not using the Parallel Query option, ensure that different datafiles and tablespaces are distributed across the available disks.

## Use Logical Volumes

Potential performance benefit: 0 - 500 percent.

You can use a Logical Volume Manager (LVM) to stripe data across multiple disk drives.

While an LVM is preferable, Oracle8 allows data files to be striped without an LVM. This is done with the **DATAFILE** keyword of the **CREATE TABLE** command. Performance is usually better with an LVM, which uses a smaller stripe size and tends to distribute I/O randomly and automatically.

## Choose the Appropriate File System or Raw Devices

UNIX file systems have different characteristics, and the techniques they use to access data can have a substantial impact on database performance. File system choices on DYNIX/ptx are:

- `ufs`: the Unix File System
- `efs`: the Enhanced File System, derived from Veritas
- `cfs`: the Clustered File System, an extension of `efs` to support clustered systems
- raw device: no file system

Using raw partitions or an Enhanced File System (EFS)/Clustered File System (CFS) instead of a UNIX File System (UFS) can improve performance because database file I/O bypasses the UNIX buffer cache and eliminates file system overhead, resulting in fewer instructions per I/O. Raw devices, CFS, and EFS also allow use of asynchronous I/O, so they are usually the best choice for high performance requirements.

If ease of administration or use of Veritas features such as caching are the primary considerations, then `efs` would be the best choice.

Oracle Parallel Server requires either raw devices or `cfs`.

---

---

**WARNING:** Apply patch 238289fix from IBM to ptx/EFS. If this patch is not applied, you will have problems with Asynchronous I/O support for both EFS and CFS.

---

---

Using `ufs` will limit the maximum file size to less than 2 GB and does not provide any performance benefits.

## Tune the Database Writer to Increase Write Bandwidth

Oracle offers solutions to prevent database writer (DBWR) activity from becoming a bottleneck:

- Use asynchronous I/O
- Use I/O slaves
- Use multiple DBWR processes

## Asynchronous I/O

Asynchronous I/O allows processes to proceed with the next operation without having to wait after issuing a write and therefore improves system performance by minimizing idle time. DYNIX/ptx supports Asynchronous I/O to both raw and CFS and EFS.

## I/O Slaves

I/O Slaves are specialized processes whose only function is to perform I/O. They are new with Oracle8, and replace Multiple DBWRs (they are a generalization of Multiple DBWRs and can be deployed by other processes as well), and can operate whether or not asynchronous I/O is available. I/O Slaves come with a new set of initialization parameters which allow a degree of control over the way they operate. These are shown in Table 3–3.

**Table 3–3 Initialization Parameters for I/O Slaves**

Parameter	Range of Values	Default Value
DISK_ASYNC_IO	TRUE/FALSE	TRUE
TAPE_ASYNC_IO	TRUE/FALSE	TRUE
BACKUP_DISK_IO_SLAVES	TRUE/FALSE	FALSE
BACKUP_TAPE_IO_SLAVES	TRUE/FALSE	FALSE
DBWR_IO_SLAVES	0 - 999	0
LGWR_IO_SLAVES	0 - 999	0
ARCH_IO_SLAVES	0 - 999	0
DB_WRITER_PROCESSES	1-10	1

There may be times when the use of asynchronous I/O is not desirable or not possible. The first two parameters in Table 3–3, `DISK_ASYNC_IO` and `TAPE_ASYNC_IO`, allow asynchronous I/O to be switched off respectively for disk and tape devices. Because the number of I/O Slaves for each process type defaults to zero, no I/O Slaves will be deployed unless specifically set.

`DBWR_IO_SLAVES` should only be set to greater than 0 if `ASYNC I/O` (that is, `DISK_ASYNC_IO`, or `TAPE_ASYNC_IO`) has been disabled, otherwise `DBWR` will become a bottleneck. In this case the optimal value on `DYNIX/ptx` for `DBWR_IO_SLAVES` should be 4. In the case of `LGWR_IO_SLAVES`, it is not recommended to deploy more than 9 slaves.

`DB_WRITER_PROCESSES` replaces the parameter `DB_WRITERS`, and specifies the initial number of database writer processes for an instance. If you use `DBWR_IO_SLAVES`, only one database writer process will be used, regardless of the setting for `DB_WRITER_PROCESSES`.

### Multiple DBWR Processes

`DB_WRITER_PROCESSES` replaces the parameter `DB_WRITERS`, and specifies the initial number of database writer processes for an instance. If you use `DBWR_IO_SLAVES`, only one database writer process will be used, regardless of the setting for `DB_WRITER_PROCESSES`.

## Separate Indexes from Tables

If an index and a table it refers to are on the same drive, all the I/O associated with an indexed search is concentrated on the same disk. Indexes and tables should be stored on separate drives to distribute I/O loads.

To move the indexes owned by user *u* from tablespace `OLD` into tablespace `NEW`, perform the following steps:

1. Calculate the size needed to hold the indexes and create `NEW`.
2. Grant resource on `NEW` to *u*.
3. Export the segments owned by *u* by specifying `full=n`, `rows=n`, and `indexes=y`.
4. Drop the indexes owned by *u*.
5. Import the segments owned by *u* by specifying `full=n`, `rows=n`, `indexes=y`, and `indexfile=filename`.
6. Edit *filename*, changing `OLD` to `NEW`.

7. Execute *filename* from SQL\*Plus.

---

**See Also:** export and import utilities are described in *Oracle8 Server Utilities*.

---

## Place Redo Logs on their Own Disk Device

Potential performance benefit: 0-15 percent.

If your Oracle applications involve heavy INSERT and UPDATE activity, you can maximize Oracle performance by locating your redo logs on disks that support no other disk activity. Also, if you have enabled the ARCHIVELOG option, place each redo log on a separate disk to minimize disk contention between the LGWR process (writing to the current redo log) and the ARCH process (reading from the closed redo log).

Place redo logs on raw devices to further enhance performance. Redo logs should be among the first files to be put on raw devices for the following reasons:

- Redo files are written and read sequentially, maximizing the benefits of raw devices.
- Asynchronous read and write is likely to be available for raw devices.
- The size of redo files is fixed, minimizing raw device administrative costs.

## Look for Large Disk Request Queues

A request queue shows how long the I/O requests on a particular disk device must wait to be serviced. Request queues are caused by a high volume of I/Os to that disk or by I/Os with long average seek times. Ideally, disk request queues should be at or near zero. The “Resp Time” field in the File I/O Monitor in Server Manager shows how long requests are waiting.

## Move Hot Files to Other Disks

Distribute frequently accessed “hot” files to less active disk devices to balance I/O. You can move an entire file from an active disk to a less active disk, or stripe a hot file across multiple disks so that part of the file is on each disk.

## Reduce I/O to Hot Files

Potential performance benefit: 0 - 50 percent.

If there is only one hot file on a disk device and the file is responsible for the large request queue, moving it to another disk will not help.

If the Oracle file or tablespace in question contains data from multiple segments (such as tables and indexes), move the heavily accessed segments to separate tablespaces and to separate files.

A physical device for a database segment can be specified only at the tablespace level. If only one segment is involved, consider table striping to place the segment data into multiple files in a single tablespace.

## Table Striping

Table striping is the process of dividing the data for a large table into small portions and storing these portions in separate data files on separate disks. This permits multiple processes to access different portions of the table concurrently without disk contention. Striping is particularly helpful in optimizing random access to tables with many rows.

---

**See Also:** The `ALLOCATE EXTENT DATAFILE` parameters of the `ALTER TABLE` command is described in the *Oracle8 Server SQL Reference*.

---

## Check for Excessive Database Fragmentation

The fragmentation of Oracle data structures requires the CPU to piece together the elements of a single logical I/O from multiple physical I/Os. The extra overhead degrades response time.

### Extent Fragmentation

Database segments may include multiple non-contiguous extents of disk space. This increases I/O time due to non-sequential disk reads or split I/Os. A split I/O occurs when a single I/O request must be split into two or more physical I/Os because the requested data spans non-contiguous extents on the disk.



## Tablespace Fragmentation

Oracle tablespaces are composed of several individual files. The Oracle segments (such as tables and indexes) within a tablespace are composed of many individual extents, resulting in tablespace fragmentation. Sometimes this sort of fragmentation is desirable, as in the case of table striping, but in most cases it is not. Each time a database segment is dropped, it causes tablespace fragmentation. Tablespace fragmentation causes inefficient use of free space.

Tablespace fragmentation prevents Oracle from taking advantage of its multi-block read capability. A fragmented tablespace file also wastes database space if the segment extents are larger than the contiguous free extents.

Tablespace fragmentation can be identified with the following SQL statement:

```
SELECT * FROM DBA_EXTENTS;
```

Free space can be queried with the following SQL statement:

```
SELECT * FROM DBA_FREE_SPACE;
```

Free space fragmentation takes two forms:

- *bubbling*: small bubbles of non-contiguous free space formed when an extent lying between active extents is dropped
- *honeycombing*: free space sectioned into contiguous pieces whenever adjacent extents are dropped

High recursive calls values in `utlbstat` and `utlestat` reports suggest tablespace fragmentation (assuming the data dictionary cache has been properly tuned).

---

**See Also:** “Avoiding a Database Reorganization” by Craig A. Shallahamer. The paper is available on the World Wide Web at <http://www.europa.com/~orapub>.

---

## When Disk I/O Optimizations Fail

When disk I/O optimizations fail to eliminate I/O bottlenecks, it may be necessary to move some applications to another system or add more disk drives and controllers to your system.

Disk Performance Issues

Oracle block sizes should either match disk block sizes, or be a multiple of disk block sizes.

If possible, do a file system check on the partition before using it for database files, then make a new file system to ensure that it is clean and unfragmented. Distribute disk I/O as evenly as possible and separate log files from database files.

Monitoring Disk Performance

To monitor disk performance, use `sar -b` and `sar -u`.

Important `sar -b` columns for disk performance are listed in Table 3–4.

**Table 3–4 Important `sar -b` Columns for Disk Performance**

<code>bread/s</code> , <code>bwrit/s</code>	blocks read and blocks written  (important for file system databases)
<code>pread/s</code> , <code>pwr/s</code>	partition reads and partition writes  (important for raw partition database systems)

An important `sar -u` column for disk performance is `%wio`, the percentage of CPU time waiting on blocked I/O.

Key indicators are:

- The sum of `bread`, `bwrit`, `pread` and `pwr/s` indicates the state of the disk I/O subsystem. The higher the sum, the greater the potential for disk I/O bottlenecks. The larger the number of physical drives, the higher the sum threshold number can be. A good default value is no more than 40 for two drives and no more than 60 for four to eight drives.
- The `%rcache` should be greater than 90 and `%wcache` should be greater than 60. Otherwise, the system may be disk I/O bound.
- If `%wio` is consistently greater than 20, the system is I/O bound.

# Tuning CPU Usage

## Keep All Oracle Users/Processes at the Same Priority

Oracle is designed to operate with all users and background processes operating at the same priority level. Changing priorities causes unexpected effects on contention and response times.

For example, if the log writer process (LGWR) gets a low priority, it is not executed frequently enough and LGWR becomes a bottleneck. On the other hand, if LGWR has a high priority, user processes may suffer poor response time.

## Use Processor Affinity/Binding on Multi-Processor Systems

In a multi-processor environment, use processor affinity/binding if it is available on your system. Processor binding prevents a process from migrating from one CPU to another, allowing the information in the CPU cache to be better utilized. You can bind a server shadow process to make use of the cache since it is always active, and let background processes flow between CPUs. Some platforms employ process binding automatically.

## Reorganize Usage Patterns

If the system is overused during peak periods, look for ways to redistribute loads to off-peak times. For example, use batch processes, perform backups overnight, and move applications to other systems.

## Use a Client/Server Configuration

If your system is CPU-bound, move applications to a separate system to reduce load on the CPU. For example, you can off-load foreground processes such as Oracle Forms to a client machine to free CPU cycles on the database server machine.

### Use the Post-Wait Driver

Potential performance benefit: 0-10 percent.

Oracle processes usually use semaphores to coordinate access to shared resources. If a shared resource is locked, a process suspends and waits for the resource to become available.

One way to improve shared resource coordination is to use a post-wait driver instead of semaphores, if it is available on your system. A post-wait driver is a faster, less expensive synchronization mechanism than a semaphore. The Oracle Post-Wait driver implements an optimized mechanism of inter-process communication, without the overhead of signal handlers or semaphores. It improves performance for Oracle8 and can be used with DYNIX/ptx. To enable this feature, set the `init.ora` parameter `USE_POST_WAIT_DRIVER = true`. Because DYNIX/ptx allows batch posting with the post\_wait driver, it is also recommended to set `_USE_VECTOR_POST = true`.

### Use Single-Task Linking for Large Exports/Imports and SQL\*Loader Jobs

If you need to transfer large amounts of data between the user and Oracle8 (for example, using `export/import`), it is efficient to use single-task architecture. To make the single-task import (`impst`), export (`expst`), and SQL\*Loader (`sqlldrst`) executables, use the `ins_rdbms.mk` makefile, which can be found in the `$ORACLE_HOME/rdbms/lib` directory.

The following example makes the `impst`, `expst`, and `sqlldrst` executables:

```
% cd $ORACLE_HOME/rdbms/lib
% make -f ins_rdbms.mk expst impst sqlldrst
```

---

---

**Note:** Linking Oracle executables as a single-task allows a user process to directly access the entire SGA. In addition, running single-task requires more memory because the oracle executable text is no longer shared between the front-end and background processes.

---

---

## Tuning Oracle Resource Contention

If your database is performing poorly and the problem is not caused by CPU or disk contention, the problem may be Oracle resource contention.

---

**See Also:** *Oracle8 Server Utilities.*

---

## Tune UNIX Kernel Parameters

You can improve performance by keeping the UNIX kernel as small as possible. The UNIX kernel typically pre-allocates physical RAM, leaving less memory available for other processes, such as `oracle`.

Traditionally, kernel parameters such as `NBUF`, `NFILE`, and `NOFILES` were used to adjust kernel size. However, most UNIX implementations dynamically adjust those parameters at run time, even though they are present in the UNIX configuration file.

Look for memory mapped video drivers, networking drivers, and disk drivers. They can often be de-installed, yielding more memory for use by other processes.

---

---

**WARNING:** Remember to make a backup copy of your UNIX kernel. See your hardware vendor documentation for additional details.

---

---

## Use V\$ Tables to Isolate Contention

Use the `V$SYSTEM_EVENT` table for a snapshot of database activity.

The statistics in `V$SYSTEM_EVENT` indicate how `oracle` is using its time, and allows you to identify potential problems. Query the table with the following SQL statement:

```
SELECT * FROM V$SYSTEM_EVENT ORDER BY TIME_WAITED;
```

A well-tuned database experiences waits, and the presence or absence of an event in this table does not necessarily indicate a problem. It is normal to see events such as *client message*, *pmon timer*, *smon timer*, *rdbms ipc message*, and *rdbms ipc reply*. The number of rows in this table changes dynamically. If there is no information to report on an event, the event will not appear in the table.

`V$SYSTEM_EVENT` is a cumulative table; it is also useful to look at a table measuring events as they occur. Use the `V$SESSION_WAIT` table by entering:

```
SELECT sid, event, p1text, p1, p2text, p2
FROM V$SESSION_WAIT;
```

This query provides a snapshot of the sequence of events. Observing how an event frequency changes with the load on the database provides insight into both the Oracle operation and the nature of the SQL statement being executed.

The sample output from querying `V$SESSION_WAIT` is:

SID	EVENT	P1TEXT	P1	P2TEXT	P2
1	pmon timer		0		0

```
2      buffer busy waits      file#      7      block#    792
10     latch free              address    8.05E08  number    8
...
```

**Isolate the Segment Causing Contention**

If you determine that Oracle resource contention is a problem, isolate the segment causing contention. You may decide the number of *buffer busy* waits are a problem. Use the *block number* and *file number* to determine the type of contention by entering:

```
SELECT segment_name, segment_type, block_id, blocks
FROM dba_extents
HERE file_id=7 AND (792 between block_id and
block_id+blocks);
```

The sample output is:

SEGMENT_NAME	SEGMENT_TYPE	BLOCK_ID	BLOCKS
-----	-----	-----	-----
COT1	TABLE	752	50
1 row selected.			

The output indicates the contention is occurring in a table segment, rather than an index, cluster, or rollback segment. Because you have the file number and block number, you can obtain additional information from the X\$BH table by entering:

```
SELECT class
FROM X$BH
WHERE dbafile=7 AND dbablk=792;
```

This query provides the class number of the block, which can be interpreted using the following table:

**Table 3-5 Block Type and Class**

Class	Block Type
0	System rollback segment
1	Data block
2	Sort block
3	Save Undo block
4	Segment header block
5	Save Undo segment header block
6	Free List block
7	Extent map block
8	Bitmap block
9	Bitmap Index block
10 + (n*2)	Undo segment header block
11 + ((n*2) +1)	Undo segment block

## Reduce Latch Free Contention

If the output from V\$SYSTEM\_EVENT indicates the value of *latch free* is causing contention, use the output from V\$SESSION\_WAIT to determine the source of the contention. The latch number is given in the *P2* field, and can be identified by entering:

```
SELECT latch#, name
FROM V$LATCH
WHERE latch#=8;
```

Sample output is:

```
LATCH#    NAME
-----
      8    8 cache buffers chains
1 row selected.
```

The *cache buffer chains* latch often experiences contention, as do the *cache buffer lru chain* latch and the *cache buffer handles* latch. These latches typically indicate that raising the number of SGA buffers is necessary.

## Reduce Rollback Segment Contention

Database data files have segments allocated for rollback information. Since the database blocks that make up rollback segments are accessed frequently, rollback segments may be subject to contention.

Use this SQL statement to determine how often requests for space in a rollback segment cause delays.

```
SELECT name, gets, waits, ((gets-waits)*100)/gets hits
FROM v$rollstat s, v$rollname n
WHERE s.usn = n.usn;
```

The hit rate should be more than 95 percent.

- When there are too few rollback segments, add more
- When users are not properly assigned, assign users who run large transactions to large rollback segments.
- Common symptom of insufficient rollback space is the error message, “Snapshot too old”.



## Reduce Redo Log Buffer Latch Contention

Heavy access to the redo log buffer can result in contention for the redo log buffer latches. Examine the activity of the redo log buffer latches through the Server Manager Latch Display.

If the ratio of *misses* to *gets* for a particular latch exceeds 10 percent, contention for that latch might affect performance. Each *Sleep* indicates a delay for the process requesting the latch.

---

---

**Note:** Systems with multiple CPUs may be able to tolerate more contention without performance reduction.

---

---

You can reduce contention for the *redo allocation* latch. Minimize the time that any single process holds the latch by decreasing the value of the `initsid.ora` `LOG_SMALL_ENTRY_MAX_SIZE` parameter.

To reduce contention for *redo copy* latches in multi-processor environments:

- Add more latches by increasing the value of `LOG_SIMULTANEOUS_COPIES`
- Use twice as many redo copy latches as CPUs available to your Oracle8 instance

## Reduce Parallel Query/Parallel DML Contention

Tune parallel queries to avoid excessive CPU usage and prevent exhausting the supply of available query servers. Use the `V$Q_SYSSTAT` view to determine the number of active query servers by entering:

```
SELECT *
FROM V$PQ_SYSSTAT
WHERE statistic = "Servers Busy":STATISTIC
```

The sample output is:

```
VALUE
-----
Servers Busy          70
```

If the value of servers busy reaches the value set for `PARALLEL_MAX_SERVERS`, it may be that some parallel queries are being processed sequentially.

Run `sar -u` along with the previous query to observe CPU loading. Observe these measurements over a significant period of time. The following table summarizes tuning actions based on the ratio of *servers busy* to `PARALLEL_MAX_SERVERS` compared to CPU utilization.

**Table 3-6 Recommended Tuning Actions Based on CPU Utilization and PARALLEL\_MAX SERVERS**

Busy/Max Servers	CPU use heavy (95 - 100%)	CPU use OK (60 - 80%)	CPU use light (0 - 30%)
1.0 often	Aggressively decrease parallelism in tables and queries; tune system	Decrease parallelism in tables and queries	Increase MAX servers; tune system
1.0 rarely	Identify queries when maximized; tune system	Increase MAX Servers - watch; decrease parallelism in tables and queries	Increase MAX servers
.3 - .7	If Query Servers using >40% CPU, decrease MAX servers; tune system	Tuned	Increase parallelism in tables and queries; increase MAX servers
0 - .2	Tune system; consider adding processors	Consider lowering MAX servers	Increase parallelism in tables and queries

## Tune Spin Count

In multi-processor environments, performance can be improved by tuning the `SPIN_COUNT` initialization parameter.

A process continues to request a latch until it obtains one. When the number of requests reaches `$SPIN_COUNT`, the process fails to acquire the latch, sleeps, then tries to acquire the latch again. Because a latch is a low-level lock, a process does not hold it long. It is less expensive to use CPU time by spinning a process than it is to make a process sleep.

To check the contention level of the latch, monitor the *miss rate* and *sleep rate* from the `utl1bstat` and `utlestat` scripts. Try reducing the sleep rate by tuning the spin count. If the contention level is high, increase the spin count to allow processes to spin more before acquiring latches. However, since increasing the spin count increases CPU usage, system throughput may decline at some point.

## Tuning Block Size and File Size

---

---

**WARNING:** To change block size, you must create a new database. Experiment with block size before transferring your data to the new database, to determine the most efficient configuration.

---

---

### Tune Block Size

Although data storage space is often measured in megabytes, the DYNIX/ptx operating system and Oracle8 Server each perform input and output in units of data storage called *blocks*. The size of the operating system blocks is not necessarily equal to Oracle blocks.

The Oracle8 server block size can be set when creating a database by changing the `DB_BLOCK_SIZE` parameter in the `initsid.ora` file.

Changing the Oracle block size can change database performance, depending on the disk hardware, file system, and application. The default size of blocksize should be adequate for most circumstances, but some performance benefits can be gained by modifying it. But beware: to change block size, you must create a new database. Experiment with block size before transferring your data to the new database, to determine the most efficient configuration.

---

---

**WARNING:** To change block size, you must create a new database. Experiment with block size before transferring your data to the new database, to determine the most efficient configuration.

---

---

### Specifying Oracle Block Size

On DYNIX/ptx, the default Oracle block size is 4KB and the maximum block size is 16KB.

You can set the actual block size to 2 KB, 4 KB, 8 KB or 16 KB.

The optimal block size is typically the default, but varies with the applications. To create a database with an different Oracle block size, add the following line to the `initsid.ora` file:

```
db_block_size=new_block_size
```

## Tuning the DYNIX/ptx Buffer Cache Size

To take full advantage of raw devices, adjust the size of the Oracle8 buffer cache and, if memory is limited, the DYNIX/ptx buffer cache.

The DYNIX/ptx buffer cache is provided by the operating system. It holds blocks of data in memory while they are being transferred from memory to disk, or vice versa.

The Oracle8 buffer cache is the area in memory that stores the Oracle database buffers. Since Oracle8 can use raw devices, it does not need to use the DYNIX/ptx buffer cache.

When moving to raw devices, increase the size of the Oracle8 buffer cache. If the amount of memory on the system is limited, make a corresponding decrease in the DYNIX/ptx buffer cache size.

The DYNIX/ptx command `sar` may help you determine which buffer caches should be increased or decreased. The `sar` command syntax is shown in Table 3–7.

**Table 3–7    *sar* Command Syntax**

<code>sar -b</code>	reports the DYNIX/ptx buffer cache activity
<code>sar -w</code>	reports the DYNIX/ptx swapping activity
<code>sar -u</code>	reports CPU utilization
<code>sar -r</code>	reports memory utilization
<code>sar -p</code>	reports the DYNIX/ptx paging activity

**Adjusting Cache Size**

- Increase Oracle8 cache size as long as the cache hit ratio goes up.
- Decrease cache sizes if the swapping/paging activity becomes high.

## Tuning Resource Contention for Oracle Parallel Server

This section describes optimization techniques designed to minimize Distributed Lock Manager (DLM) bottlenecks.

Database concurrence in an Oracle Parallel Server (OPS) system is maintained across the processors using a DLM. Managing resources using the DLM is less efficient than using the shared memory model within a single database instance.

### Avoid Index Contention

Index tables are used extensively and may be a source of contention in your database if a sequence generator is used to create primary keys for database records. The sequence numbers are typically consecutive and, when used as keys to add data, cause entries in the same index blocks. This can result in contention for the index blocks. Solve this problem by pre-pending a value to the sequence. Select a value to distribute indexes to different blocks.

---

---

**Note:** Index contention can be a problem for Oracle8 running on a Symmetric Multi-Processor (SMP), but is more likely to become a bottleneck on Oracle Parallel Server.

---

---

### Avoid Free List Contention

Blocks available for insert operations are kept on a list in the table header. Insert-intensive applications experience contention for the table header block. Solve this problem by creating multiple free lists and multiple free list groups. Free list headers are kept in different blocks.

### Avoid Lock Contention

An application will not scale well if there is excessive lock contention. Lock contention can be measured by fields from the V\$SYSSTAT table (CLASS=32). CLASS is a column in the V\$SYSSTAT table, and the '32' identifies global locks.

Lock Conversion Ratio =  $\frac{\text{Consistent Gets} - \text{Async Lock Converts}}{\text{Consistent Gets}}$

The lock conversion ratio should be 95 percent or higher for the application to scale well. If there is excessive lock contention, the application must be re-evaluated and possibly re-designed for OPS.

Although the application being executed by `oracle` is the most common source of lock contention, sometimes insufficient locks have been allocated, or were poorly allocated. For

example, an OLTP application requires more locks than a decision support application. Allocate locks appropriately with the `init.ora` parameters.

### Localize Disk I/O

Keep the rollback segments and redo logs for an instance on the disks connected to that node. This should be part of your overall strategy of partitioning data so each node uses data without contention.

### Monitor Contention

Many statistics can indicate OPS contention. Examine the following tables to determine OPS contention:

- V\$SESSION\_WAIT
- V\$SESSION\_EVENT
- V\$SYSTEM\_EVENT
- V\$SQL\_AREA
- V\$CACHE
- V\$PING

Generally, lock conversions are the most important factor. Lock conversions imply disk I/O and delays while the lock is acquired and converted. Proper application partitioning is the only way to avoid lock conversions.

## Using Trace and Alert Files

This section describes the trace (or dump) and alert files the Oracle Server creates to diagnose and resolve operating problems.

### Trace File Names

The format of a trace file name is *processname\_sid\_unixpid.trc*, where:

**Table 3–8 Format Key to Process Name**

<i>processname</i>	is a three- or four-character process name showing which Oracle8 process the trace file is from (for example, PMON, DBWR, ORA, or RECO)
<i>sid</i>	is the instance system identifier
<i>unixpid</i>	is the UNIX process ID number
<i>.trc</i>	is a file name extension appended to all trace file names

A sample trace file name is *lgwr\_TEST\_1237.trc*.

### Alert Files

The *alert\_sid.log* file is associated with a database and is located in the directory specified by the *initSID.ora* parameter **BACKGROUND\_DUMP\_DEST**. The default value is *\$ORACLE\_HOME/rdbms/log*.

## Raw Devices

### Disadvantages of Raw Devices

Raw devices have the following disadvantages when used on DYNIX/ptx:

- They may not solve problems with ULIMIT that can arise when exporting tables larger than a megabyte (such as another disk partition).
- When raw devices and operating system files are mixed within an Oracle8 database, the operating system files must still be within the value of the ULIMIT parameter.
- They may not solve problems with ULIMIT that can arise when reading the contents of the Oracle distribution media onto the disk.
- Small client systems usually cannot use sufficiently large raw device partitions. Disk partitions usually come in odd sizes that may hinder good database architecture.
- If a particular disk drive has intense I/O activity and performance would benefit from movement of an Oracle data file to another drive, it is likely that no acceptably sized section exists on a drive with less I/P activity. Moving data files around, a common advantage of UNIX, may not be possible with raw devices.
- Adding space to a tablespace can be a difficult process in a raw device environment. Occasionally, all raw partitions are assigned data files at initial configuration time, leaving no raw storage to accommodate normal tablespace growth.

### Criteria for Using Raw Devices

These factors should be considered when deciding on raw devices:

- Oracle8 Parallel Server (OPS) installation
- Raw disk partition availability

### Oracle8 Parallel Server Installation

Each instance of OPS has individual log files. Therefore, in addition to the partitions required for the tablespaces and control files, each instance requires a minimum of three partitions for the log files. All the files must be on disks that can be shared by all nodes of a DYNIX/ptx cluster.

UNIX clusters do not provide access to a shared file system between all nodes of a cluster. As a result, all files associated with a database must be built on raw devices.



### **Raw Disk Partition Availability**

Use raw devices for Oracle files if your site has at least as many raw disk partitions as Oracle tablespaces.

If the raw disk partitions are already formatted, match tablespace size to partition size as closely as possible to avoid wasting space.

## **Guidelines for Using Raw Devices**

When creating raw disk partitions, observe these guidelines:

- Three partitions for the log files of each instance
- One partition each for the following datafiles: SYSTEM, ROLLBACK, TEMP, USERS, TOOLS
- At least three partitions for data files

### **Configuration Planning**

With logical volumes, you can create logical disks based on raw partition availability, because logical disks can be moved on more than one disk. The disk drives do not have to be reformatted to obtain logical disk sizes.

### **Dynamic Performance Tuning**

Disk performance can be optimized when the database is online by moving hot spots to cooler drives. Most hardware vendors who provide the logical disk facility also provide a graphical user interface that can be used for tuning.

### **Mirroring and Online Disk Replacement**

Mirroring of logical volumes is possible and should be used to protect against loss of data. If one copy of a mirror fails, dynamic re-synchronization is possible. Some vendors also provide the ability to replace drives online in conjunction with the mirroring facility.

**For Parallel Server:** Logical volumes are available for drives associated with a single UNIX machine, as well as those that can be shared with more than one machine of a UNIX cluster. The latter allows for all files associated with the Oracle Parallel Server to be placed on these shared logical volumes.

## Setting Up Raw Devices

---

---

**WARNING:** Do not attempt to set up raw devices without the help of an experienced system administrator and specific knowledge about the machine you are using.

---

---

To set up raw devices on your system:

1. (This step is for Oracle Parallel Server only.) Make sure the partitions you are adding are on a shared disk.
2. Determine the names of the free disk partitions.

A free partition is one that is not used for a DYNIX/ptx file system. That means that the partition follows these restrictions:

- n It is not listed when you execute the `/etc/mount` command.
- n It is not in use as a swap device.
- n It does not overlap a swap partition.
- n It is not in use by other DYNIX/ptx applications (for example, other instances of Oracle).
- n It does not overlap the DYNIX/ptx file system.
- n It does not use a space already used by the file system.

To find out whether a partition is free, obtain a complete map of the starting locations and sizes of the partitions on the device and check for free space. Note that some partitions may contain file systems that are currently not mounted and are not listed in the `/etc/mount` output.

---

---

**Attention:** Make sure that the partition does *not* start at Cylinder 0.

---

---

3. Set up the raw device for use by the Oracle8 Server.

Begin by verifying that the disk is partitioned. If not, use the operating system `format` utility to partition it.

Next, make sure that the partition is owned by the *oracle* software owner. If necessary, use *chown* to change its ownership on the block and character files for the device. For example:

```
$ chown oracle /devices/ionmu@f,e0000000/ \
sbus@f,e0001000/espdma@f,400000/esp@f,800000/ \
sd@5,0:a
$ chown oracle /devices/ionmu@f,e0000000/\
sbus@f,e0001000/espdma@f,400000/esp@f,800000/ \
sd@5,0:a,raw
```

Use *chmod* to make the partition accessible only by the *oracle* software owner. For example:

```
$ chmod 600 /devices/ionmu@f,e0000000/ \
sbus@f,e0001000/espdma@f,400000/esp@f,800000/ \
sd@5,0:a
$ chmod 600 /devices/ionmu@f,e0000000/ \
sbus@f,e0001000/espdma@f,400000/esp@f,800000/ \
sd@5,0:a,raw
```

4. Create a symbolic link to the raw devices you require. For example:

```
$ ln -s /devices/ionmu@f,e0000000/sbus@f,e0001000\
/espdma@f,400000/esp@f,800000/sd@5,0:a,raw /oracle_data/datafile.dbf
```

Make sure you use the character special device, not the block special device. If this is correct, the following command,

```
$ ls -lL datafile
```

should return,

```
crw----- oracle dba datafile
```

(the flags used in the above command are: L = show symbolic links, and l = long listing).

---

---

**Note:** This symbolic link must be set up on each node of the Parallel Server. Check that no two symbolic links point to the same raw device.

---

---

5. Create or add the new partition to a new database.

From Server Manager, use the SQL statement `CREATE DATABASE` to create the database using the specified raw partition.

---

---

**Note:** The size of an Oracle datafile created in a raw partition must be at least two Oracle block sizes smaller than the size of the raw partition.

---

---

### **Example 3–1**

```
$ svrmgrl
SVRMGR> create database sid
SVRMGR> logfile '/oracle_data/log1.dbf' size 100K,
'/oracle_data/log2.dbf' size 100K
SVRMGR> datafile '/oracle_data/datafile.dbf' size 10000K
reuse;
```

If you want to add the partition to a tablespace in an existing Oracle database instead, enter:

```
$ svrmgrl
SVRMGR> alter tablespace tablespace_name add datafile
'/dev/rdsk/c0t1d0s6' size 10000K reuse;
```

You can use the same procedure to set up a raw device for the redo log files.

---

# Administering SQL\*Plus on DYNIX/ptx

- Administering SQL\*Plus
- Using SQL\*Plus
- Restrictions

# Administering SQL\*Plus

## Setup Files

The setup files for SQL\*Plus are `glogin.sql`, the global setup file which defines the site profile, and `login.sql`, which defines the user profile. The `glogin.sql` and `login.sql` files contain either SQL statements or SQL\*Plus commands that you choose to execute at the beginning of each SQL\*Plus session. When you invoke SQL\*Plus, `glogin.sql` is read first, followed by `login.sql`.

## The Site Profile

The Site Profile file is `$ORACLE_HOME/sqlplus/admin/glogin.sql`. SQL\*Plus executes this command file whenever any user starts SQL\*Plus and SQL\*Plus establishes the Oracle connection. The default Site Profile is placed in `$ORACLE_HOME/sqlplus/admin` whenever SQL\*Plus is installed. If a Site Profile already exists, it will be overwritten. An existing Site Profile is deleted whenever SQL\*Plus is de-installed.

## The User Profile

The User Profile file is `login.sql`. SQL\*Plus attempts to execute this command file whenever any user starts SQL\*Plus and SQL\*Plus establishes the Oracle connection. The User Profile is run after the Site Profile. SQL\*Plus always searches the current directory for the User Profile. The environment variable `SQLPATH` may be set to a colon-separated list of directories that SQL\*Plus will search in order.

For example, if the current directory is `/u02/oracle` and `SQLPATH` is set as follows:

```
% echo $SQLPATH
/home:/home/oracle:/u01/oracle
```

SQL\*Plus will first look for `login.sql` in the current directory `/u02/oracle`. If it is not found there, SQL\*Plus will then look in `/home`, `/home/oracle`, and `/u01/oracle`, respectively.

Here is a sample `login.sql` file:

```
set echo off
set feedback 4
set pause on
set pause "PLEASE PRESS RETURN TO CONTINUE"
set message on
set echo on
```

## The PRODUCT\_USER\_PROFILE Table

The SQL script `$ORACLE_HOME/sqlplus/admin/pupbld.sql` may be run as the user `SYSTEM` to create the Product and User Profile tables.

`$ORACLE_HOME/sqlplus/admin/pupbld.sql` may also be run using the shell script `$ORACLE_HOME/bin/pupbld`. To use this script, the environment variables `ORACLE_HOME` and `SYSTEM_PASS` must be set. `SYSTEM_PASS` must be set to the `SYSTEM`'s username and password. For example:

```
% setenv SYSTEM_PASS SYSTEM/manager
% pupbld
```

```
Installing product user profile tables...
```

```
Product user profile tables installed.
```

`pupbld.sql` will only be run by the Installer during SQL\*Plus installation if Create Database Objects was selected.

## Demonstration Tables

SQL\*Plus is shipped with demonstration tables that may be used for testing.

### Default Install

If using Default Install and Create Database Objects, the user `SCOTT` and the demonstration tables will be created automatically.

### Custom Install

When installing SQL\*Plus using Custom Install, if Create Database Objects is selected and you answer 'Yes' to the prompt "Would you like to load the SQL\*Plus Demo Tables?", the Installer will create the user `SCOTT` with the password `TIGER` and create the demonstration tables.

### Creating Demonstration Tables Manually

The SQL script `$ORACLE_HOME/sqlplus/demo/demobld.sql` is used to create the demonstration tables. The file `demobld.sql`, may be run in SQL\*Plus as any user to create the demonstration tables in that schema. For example:

```
% sqlplus scott/tiger
SQL> @?/sqlplus/demo/demobld.sql
```

`$ORACLE_HOME/sqlplus/demo/demobld.sql` may also be run using the shell script `$ORACLE_HOME/bin/demobld` as follows:

```
% demobld scott tiger
```

### Deleting Demonstration Tables

The SQL script `$ORACLE_HOME/sqlplus/demo/demodrop.sql` is used to drop the demonstration tables. The file `demodrop.sql` may be run in SQL\*Plus as any user to drop the demonstration tables from that user's schema. For example:

```
% sqlplus scott/tiger
SQL> @?/sqlplus/demo/demodrop.sql
```

`$ORACLE_HOME/sqlplus/demo/demodrop.sql` may also be run using the shell script `$ORACLE_HOME/bin/demodrop` as follows:

```
% demodrop scott tiger
```

---

---

**Note:** Both SQL scripts `demobld.sql` and `demodrop.sql` drop the tables EMP, DEPT, BONUS, SALGRADE, and DUMMY. You must ensure that a table does not exist with the same name in the desired schema prior to running either script or the table data will be lost.

---

---

## Help Facility

### Default Install

If using Default Install and Create Database Objects, the Help Facility is installed automatically.



## Custom Install

When installing SQL\*Plus, if Create Database Objects is selected and you answer 'Yes' to the prompt "Would you like to load the SQL\*Plus Help Facility?", the Installer will create the Help Facility.

## Installing the Help Facility Manually

The Help Facility may be installed manually using the shell script `$ORACLE_HOME/bin/helpins`. To use this script, the environment variables `ORACLE_HOME` and `SYSTEM_PASS` must be set. `SYSTEM_PASS` must be set to `SYSTEM`'s username and password. For example:

```
$ setenv SYSTEM_PASS SYSTEM/manager
$ helpins
```

```
SQL*Loader: Release 8.0.6.0.0 - Production
```

```
(c) Copyright 2000 Oracle Corporation. All rights reserved.
```

```
Commit point reached - logical record count 828
```

```
SQL*Loader: Release 8.0.6.0.0 - Production
```

```
(c) Copyright 2000 Oracle Corporation. All rights reserved.
```

```
Commit point reached - logical record count 1024
```

```
Commit point reached - logical record count 1207
```

```
SQL*Loader: Release 8.0.6.0.0 - Production
```

```
(c) Copyright 2000 Oracle Corporation. All rights reserved.
```

```
Commit point reached - logical record count 1024
```

```
Commit point reached - logical record count 1304
```

```
Commit point reached - logical record count 2328
```

```
Commit point reached - logical record count 2724
```

```
Commit point reached - logical record count 2835
```

---

---

**See Also:** Refer to the *SQL\*Plus User's Guide and Reference*, and the README file, `$ORACLE_HOME/sqlplus/doc/release.doc`.

---

---

## Using SQL\*Plus

### Using a System Editor from SQL\*Plus

An `ed` or `edit` command entered at the SQL\*Plus prompt calls a default operating system editor, such as `ed`, `emacs`, `ned`, or `vi`. Your `PATH` variable must include the directory of the editor.

The global default editor is usually set by the DBA in `glogin.sql` using the SQL\*Plus `_editor` option. Override this setting by specifying an editor in `login.sql`. Both files are read by SQL\*Plus at startup, the local file taking precedence. The `_editor` option can also be set during a SQL\*Plus session, overriding the setting in either file.

If the `_editor` option is not set, the `EDITOR` and `VISUAL` environment variables specify the SQL\*Plus editor. These variables are not set in `glogin.sql` or `login.sql`. They are set in a user startup file, or at the system prompt. If both are set, the `EDITOR` variable is used.

### Setting the Order of the Editor

SQL\*Plus searches for the default editor in this order:

1. The `_editor` variable during a SQL\*Plus session.
2. The `_editor` variable in `login.sql`.
3. The `_editor` variable in `glogin.sql`.
4. The `EDITOR` environment variable.
5. The `VISUAL` environment variable.

When none of these values are set, SQL\*Plus uses `ed`.

### Setting the `_editor` option

Set the SQL\*Plus `_editor` option by adding the following line to the `login.sql` file:

```
define _editor=editor_name
```

where `editor_name` is a UNIX editor.

### Setting Environment Variables

For the Bourne or Korn shell, set the default editor with an environment variable by entering:

```
$ UNIX_VAR=editor_name; export UNIX_VAR
```

For the C shell, set the default editor with an environment variable by entering:

```
% setenv UNIX_VAR editor_name
```

Environment variable syntax is explained in Table 4–1.

**Table 4–1 Syntax for UNIX Environment Variables**

<i>UNIX_VAR</i>	the EDITOR or VISUAL environment variable
<i>editor_name</i>	the UNIX editor (for example, vi or ed)

## Default Settings

If you call the system editor, the current SQL buffer is placed in the edit buffer and all statements available to the editor can change the SQL statement. SQL\*Plus uses the `afiedt.buf` temporary file. When you exit the editor, the changed SQL buffer is returned to SQL\*Plus.

## Running Operating System Commands from SQL\*Plus

An exclamation point (!) in the first position after the SQL\*Plus prompt indicates subsequent character strings are passed to a sub-shell. The SHELL environment variable selects the shell you use to execute operating system commands. The default shell is `/bin/sh (sh)`. If the shell cannot be executed, an error message is displayed.

Use the following SQL\*Plus commands to perform specific tasks:

- Enter `[!]+[command]` to execute one operating system command. After the command executes, control returns to SQL\*Plus.
- Enter `[!]+[Return]` to execute *more than one* operating system command. When you finish, enter `[Ctrl]+[d]` to return to SQL\*Plus.

## Interrupting SQL\*Plus

While running SQL\*Plus:

- You can stop the scrolling record display and terminate a SQL statement by pressing [Ctrl]+[c] on BSD machines or [Delete] on System V machines.
- If you are at the SQL\*Plus prompt, pressing [Interrupt] displays another SQL\*Plus prompt.

## Using the SPOOL Command

The default filename extension for files generated by the SPOOL command is .lst. To change the extension, specify a spool file containing a period (.).

For example:

```
SQL> SPOOL query.lis
```

## Restrictions

### COPY Command

The COPY command in SQL\*Plus is supported without restrictions on different machines running the same version of the operating system.

COPY may also work between machines running different versions of the operating system. If COPY fails, test the connection using rcp or ftp. Restrictions in vendor-supplied networking software may prevent rcp, ftp, or COPY from functioning properly between systems.

---

---

**Note:** The rlogin command does not send or receive large packets of data and is not an adequate test for connections.

---

---

If COPY does not function between systems, create a database link to the system and user ID indicating the table you want to copy. To do this, enter:

```
SQL> create table newtable as \  
(SELECT * FROM table@database_link_name)
```

This selects the rows and columns from the original table on the remote system and enters them in the new table on the local system.

## Resizing Windows

The default value for SQL\*Plus `LINE SIZE` is 80 and for `PAGESIZE` is 25. These variables do not automatically adjust for window size.

## Return Codes

UNIX return codes use only one byte, which is not enough space to return an Oracle error code. The range for a return code is 0 to 255.



---

# Using Oracle Precompilers and the Oracle Call Interface on DYNIX/ptx

- n Overview of Oracle Precompilers
- n Pro\*C/C++
- n Pro\*COBOL
- n Pro\*FORTRAN
- n Oracle Call Interface
- n Oracle Precompiler and Oracle Call Interface Linking and Makefiles
- n Thread Support
- n Static and Dynamic Linking with Oracle Libraries
- n Using Signal Handlers
- n XA Functionality

# Overview of Oracle Precompilers

Oracle precompilers are application design tools used to combine SQL statements from an Oracle database with programs written in a high-level language. Oracle precompilers are compatible with ANSI SQL and are used to develop open, customized applications that run with Oracle8, or any other ANSI SQL DBMS.

## Relinking Precompiler Executables

All precompiler executables are relinked using the makefile, `$ORACLE_HOME/precomp/lib/ins_precomp.mk`. The make command uses the following convention:

```
$ make -f ins_precomp.mk relink EXENAME=executable
```

This command will create the new executable in the `$ORACLE_HOME/precomp/lib` directory, and then move it to `$ORACLE_HOME/bin`. In order to create the new executable without it being moved to `$ORACLE_HOME/bin`, use the following command:

```
$ make -f ins_precomp.mk executable
```

where the name of the executable in respect of the product being used, can be determined from Table 5–1.

**Table 5–1 Products and Their Corresponding Executable Names**

Product	Executable
Pro*C/C++	proc
Pro*COBOL v1.8.28	procobl8, or rtsora
Pro*COBOL v8.0.6	procob, or rtsora
Pro*FORTRAN	profor
Object Type Translator	ott

For example, to relink the Pro\*C/C++ executable, use the following command:

```
$ cd $ORACLE_HOME/precomp/lib
$ make -f ins_precomp.mk relink EXENAME=proc
```



## Precompiler Configuration Files

There are five .cfg system configuration files in \$ORACLE\_HOME/precomp/admin. These are described in Table 5–2.

**Table 5–2 System Configuration Files**

Product	Configuration File
Pro*C/C++ v8.0.6	pcscfg.cfg
Pro*COBOL v8.0.6	pcbcfg.cfg
Pro*COBOL v1.8.28	pcccob.cfg
Pro*FORTRAN v1.8.28	pccfor.cfg
Object Type Translator v8.0.6	ottcfg.cfg

## Issues Common to All Precompilers

### Uppercase to Lowercase Conversion

In languages other than C, your compiler converts an uppercase function or subprogram name to lowercase. This can cause “No such user exit” errors. In this case, verify that the function or subprogram name in your option file matches the case in the `iapxtb` table.

### Vendor Debugger Programs

Precompilers and vendor-supplied debuggers may be incompatible. Oracle Corporation does not guarantee that a program run under a debugger will run the same way under an operating system.

### Value of `ireclen` and `oreclen`

The `ireclen` and `oreclen` parameters do not have maximum values.

## Supplemental Documentation

The following documents provide additional information about precompiler and interface features:

- n *Programmer's Guide to the Pro\*C/C++ Precompiler*
- n *Programmer's Guide to the Pro\*COBOL Precompiler*
- n *Pro\*FORTRAN Supplement to Oracle Precompilers*
- n *Programmer's Guide to the Oracle Call Interface*
- n *Oracle8 Server Application Developer's Guide*

## Pro\*C/C++

For additional information regarding Pro\*C/C++ release 8.0.6, see the README file, `$ORACLE_HOME/precomp/doc/proc2/readme.doc`.

## Administering Pro\*C/C++

### System Configuration File

The system configuration file for Pro\*C/C++ is `$ORACLE_HOME/precomp/admin/pcscfg.cfg`.

## Using Pro\*C/C++

Prior to using Pro\*C/C++, verify that the correct version of the Operating System compiler is properly installed. The required version is documented in Chapter 1 of the *Oracle8 Installation Guide for IBM DYNIX/ptx*.

### Demonstration Programs

Demonstration programs are provided to show the various functionality of the Pro\*C/C++ precompiler. There are three types of demonstration programs: C, C++, and Object programs - the latter demonstrate the new Oracle8 Object features. All the demonstration programs are located in `$ORACLE_HOME/precomp/demo/proc`, and all of them assume that the demonstration tables created by `$ORACLE_HOME/sqlplus/demo/demobld.sql` exist in the SCOTT schema with the password TIGER.

For further information on building the demonstration programs using SQL\*Plus, see "Demonstration Tables" on page 4-3 of this book. For further information on the demonstration programs see the *Programmer's Guide to the Pro\*C/C++ Precompiler*.

The makefile, `$ORACLE_HOME/precomp/demo/proc/demo_proc.mk`, should be used to create the demonstration programs. For example, to precompile, compile, and link the `sample1` demonstration program, enter the following command.

```
$ make -f demo_proc.mk sample1
```

Alternatively, the following command may be used, which achieves exactly the same result, only with more explicit syntax.

```
$ make -f demo_proc.mk build OBJS=sample1.o EXE=sample1
```

By default, all programs are dynamically linked with the client shared library, `$ORACLE_HOME/lib/libclntsh.so`.

To create all Pro\*C/C++ C demonstration programs, enter the following command:

```
$ make -f demo_proc.mk samples
```

To create all Pro\*C/C++ C++ demonstration programs, enter this command:

```
$ make -f demo_proc.mk cppsamples
```

To create all Pro\*C/C++ Object demonstration programs, enter this command:

```
$ make -f demo_proc.mk object_samples
```

Some demonstration programs require a SQL script, found in `$ORACLE_HOME/precomp/demo/sql`, to be run. In order to build such a demonstration program and run the corresponding SQL script, the `make` macro argument, `RUNSQL=run`, must be included on the command line. For example, to create the `calldemo` demonstration program and run the required `$ORACLE_HOME/precomp/demo/sql/calldemo.sql` script, use the following command syntax:

```
$ make -f demo_proc.mk calldemo RUNSQL=run
```

As another example, to create all Object demonstration programs and run all corresponding required SQL scripts, enter the following command:

```
$ make -f demo_proc.mk object_samples RUNSQL=run
```

The SQL scripts may also be run manually, if desired.

User Programs

The makefile, \$ORACLE\_HOME/precomp/demo/proc/demo\_proc.mk, may be used to create user programs. The general syntax for linking a user program with demo\_proc.mk is as follows:

```
$ make -f demo_proc.mk target OBJS="objfile1 objfile2 ..." \
    EXE=exename
```

For example, to create the program, myprog, from the Pro\*C/C++ source myprog.pc, use one of the following commands, depending on the source and type of executable desired:

For C source, dynamically linked with client shared library:

```
$ make -f demo_proc.mk build OBJS=myprog.o EXE=myprog
```

For C source, statically linked:

```
$ make -f demo_proc.mk build_static OBJS=myprog.o EXE=myprog
```

For C++ source, dynamically linked with client shared library:

```
$ make -f demo_proc.mk cppbuild OBJS=myprog.o EXE=myprog
```

For C++ source, statically linked:

```
$ make -f demo_proc.mk cppbuild_static OBJS=myprog.o EXE=myprog
```

For IBM DYNIX/ptx issues on the use of shared libraries, refer to the IBM DYNIX/ptx documentation from IBM.

Pro\*COBOL

There are two versions of Pro\*COBOL included with this release: Pro\*COBOL 8.0.6, and Pro\*COBOL 1.8.28. Table 5–3 shows the naming differences between these two versions.

Table 5–3 Pro\*COBOL naming differences

	Pro*COBOL 8.0.6	Pro*COBOL 1.8.28
Executable	procob	procobl8
Demo Directory	procob2	procob
Makefile for MicroFocus COBOL	demo_procob.mk	demo_procobl8.mk

Pro\*COBOL supports statically linked, dynamically linked, or dynamically loadable programs. Dynamically linked programs use the Oracle client shared library, `$ORACLE_HOME/lib/libclntsh.so`. Dynamically loadable programs use the `rtsora` executable.

For additional information regarding Pro\*COBOL 8.0.6, see the README file, `$ORACLE_HOME/precomp/doc/procob2/readme.doc`. For additional information regarding Pro\*COBOL 1.8.27, see the README file, `$ORACLE_HOME/precomp/doc/prolx/readme.txt`.

## Administering Pro\*COBOL

### System Configuration File

The system configuration file for Pro\*COBOL 8.0.6 is `$ORACLE_HOME/precomp/admin/pcbcfg.cfg`.

The system configuration file for Pro\*COBOL 1.8.28 is `$ORACLE_HOME/precomp/admin/pcccob.cfg`.

## Environment Variables

### MicroFocus COBOL Compiler

The MicroFocus COBOL Compiler requires the environment variables `COBDIR` and `LD_LIBRARY_PATH`. `COBDIR` must be set to the directory where the compiler is installed. For example:

```
$ setenv COBDIR /opt/cobol
```

`LD_LIBRARY_PATH` must include the directory `$COBDIR/coblib`. For example, to append `$COBDIR/coblib` to `LD_LIBRARY_PATH`:

```
$ setenv LD_LIBRARY_PATH ${LD_LIBRARY_PATH}:$COBDIR/coblib
```

If `LD_LIBRARY_PATH` does not contain `$COBDIR/coblib`, you will receive the following error when compiling a program:

```
dynamic linker: rts32: open libfhutil.so.2.0: No such file or directory
```

## Using Pro\*COBOL

Prior to using Pro\*COBOL, verify that the correct version of the COBOL compiler is properly installed. The required version for your operating system is documented in Chapter 1 of the *Oracle8 Installation Guide for IBM DYNIX/ptx*.

### The Oracle Run Time System

Oracle provides its own complete run time system, called *rtsora*, to run dynamically loadable Pro\*COBOL programs. The *rtsora* run time system should be used in place of the MicroFocus provided *cobrun* run time system when running dynamically loadable Pro\*COBOL programs. If you attempt to run a Pro\*COBOL program with *cobrun*, you will receive the following error:

```
$ cobrun sample1.gnt
Load error : file 'SQLADR'
error code: 173, pc=0, call=1, seg=0
173      Called program file not found in drive/directory
```

### Demonstration Programs

Demonstration programs have been provided that show various functionality of the Pro\*COBOL precompiler. All programs are located in either `$ORACLE_HOME/precomp/demo/procob` or `$ORACLE_HOME/precomp/demo/procob2`, depending on the Pro\*COBOL version. All programs assume that the demonstration tables created by `$ORACLE_HOME/sqlplus/demo/demobld.sql` exist in the SCOTT schema with the password TIGER. For further information on building the demonstration programs using SQL\*Plus, see “Demonstration Tables” on page 4-3 of this book.

---

**See Also:** For further information on the demonstration programs see the *Programmer's Guide to the Pro\*COBOL Precompiler*.

---

The demonstration makefile should be used to create the sample programs. The demonstration makefile for Pro\*COBOL 8.0.6 is `$ORACLE_HOME/precomp/demo/procob2/demo_procob.mk`. The demonstration makefile for Pro\*COBOL 1.8.28 is `$ORACLE_HOME/precomp/demo/procob/demo_procob18.mk`. For example, to precompile, compile, and link the `sample1` demonstration program for Pro\*COBOL 8.0.6, use the following command:

```
$ cd $ORACLE_HOME/precomp/demo/procob2
```

```
$ make -f demo_procob.mk sample1
```

Alternatively, the following command may be used, which achieves exactly the same result, only with more explicit syntax.

```
$ make -f demo_procob.mk build COBS=sample1.cob EXE=sample1
```

By default, all programs are dynamically linked with the client shared library, `$ORACLE_HOME/lib/libclntsh.so`.

To create all Pro\*COBOL demonstration programs, enter the following command:

```
$ make -f demo_procob.mk samples
```

To create a dynamically loadable `sample1.gnt` program to be used with `rtsora`, enter this command:

```
$ make -f demo_procob.mk sample1.gnt
```

Then use `rtsora` to run the program as follows:

```
$ rtsora sample1.gnt
```

Some demonstration programs require a SQL script found in `$ORACLE_HOME/precomp/demo/sql` to be run. In order to build such a demonstration program and run the corresponding SQL script, the make macro argument, `RUNSQL=run`, must be included on the command line.

For example, to create the `sample9` demonstration program and run the required `$ORACLE_HOME/precomp/demo/sql/sample9.sql` script, use the following command syntax:

```
$ make -f demo_procob.mk sample9 RUNSQL=run
```

The SQL scripts may also be run manually, if desired.

## User Programs

The demonstration makefile may be used to create user programs. Be sure to use the appropriate makefile depending on the Pro\*COBOL version and COBOL compiler used. The general syntax for linking a user program with the demonstration makefile is:

```
$ make -f demo_procob.mk target COBS="cobfile1 cobfile2 ..." \  
EXE=exename
```

For example, to create the program, `myprog`, from the Pro\*COBOL source `myprog.pco`, use one of the following commands, depending on the type of executable and use of shared library resources desired:

For a dynamically linked executable with client shared library:

```
$ make -f demo_procob.mk build COBS=myprog.cob EXE=myprog
```

For a statically linked executable without client shared library:

```
$ make -f demo_procob.mk build_static COBS=myprog.cob EXE=myprog
```

For a dynamically loadable module usable with `rtsora`

```
$ make -f demo_procob.mk myprog.gnt
```

### FORMAT Precompiler

The `FORMAT` precompiler option specifies the format of input lines for COBOL. If you specify `FORMAT=ANSI`, the default, columns 1 to 6 contain an optional sequence number, column 7 indicates comments or continuation lines, paragraph names begin in columns 8 to 11, and statements begin in columns 12 to 72.

If you specify `FORMAT=TERMINAL`, columns 1 to 6 are dropped, making column 7 the leftmost column.

## Pro\*FORTRAN

For additional information regarding Pro\*FORTRAN 1.8.28, see the `README` file, `$ORACLE_HOME/precomp/doc/prolx/readme.txt`.

## Administering Pro\*FORTRAN

### System Configuration File

The system configuration file for Pro\*FORTRAN is `$ORACLE_HOME/precomp/admin/pccfor.cfg`.

## Using Pro\*FORTRAN

Prior to using Pro\*FORTRAN, verify that the correct version of the compiler is properly installed. The required version for your operating system is specified in Chapter 1 of the *Oracle8 Installation Guide for IBM DYNIX/ptx*.



## Demonstration Programs

Demonstration programs are provided to show the various functionality of the Pro\*FORTRAN precompiler. All programs are located in `$ORACLE_HOME/precomp/demo/profor`, and all of them assume that the demonstration tables created by `$ORACLE_HOME/sqlplus/demo/demobld.sql` exist in the SCOTT schema with the password TIGER.

For further information on building the demonstration programs using SQL\*Plus, see “Demonstration Tables” on page 4-3 of this book.

---

**See Also:** For further information on the demonstration programs see the *Pro\*FORTRAN Supplement to Oracle Precompilers*.

---

The makefile, `$ORACLE_HOME/precomp/demo/profor/demo_profor.mk`, should be used to create the demonstration programs. For example, to precompile, compile, and link the `sample1` demonstration program, enter the following command.

```
$ make -f demo_profor.mk sample1
```

Alternatively, the following command may be used, which achieves exactly the same result, only with more explicit syntax.

```
$ make -f demo_profor.mk build FORS=sample1.pfo EXE=sample1
```

By default, all programs are dynamically linked with the client shared library, `$ORACLE_HOME/lib/libclntsh.so`.

To create all Pro\*FORTRAN demonstration programs, enter the following command:

```
$ make -f demo_profor.mk samples
```

Some demonstration programs require a SQL script, found in `$ORACLE_HOME/precomp/demo/sql`, to be run. In order to build such a demonstration program and run the corresponding SQL script, the make macro argument, `RUNSQL=run`, must be included on the command line. For example, to create the `sample11` demonstration program and run the required `$ORACLE_HOME/precomp/demo/sql/sample11.sql` script, use the following command syntax:

```
$ make -f demo_profor.mk sample11 RUNSQL=run
```

The SQL scripts may also be run manually, if desired.

## User Programs

The makefile, `$ORACLE_HOME/precomp/demo/profor/demo_profor.mk`, may be used to create user programs. The general syntax for linking a user program with `demo_profor.mk` is as follows:

```
$ make -f demo_profor.mk target FORS="forfile1 forfile2 ..." \  
    EXE=exename
```

For example, to create the program, `myprog`, from the Pro\*FORTRAN source `myprog.pfo`, use one of the following commands, depending on the type of executable desired:

For dynamically linked executable with client shared library:

```
$ make -f demo_profor.mk build FORS=myprog.f EXE=myprog
```

For a statically linked executable:

```
$ make -f demo_profor.mk build_static FORS=myprog.f EXE=myprog
```

# Oracle Call Interface

## Using the Oracle Call Interface

Prior to using the Oracle Call interface (OCI), verify that the correct version of the compiler is properly installed. The required version for DYNIX/ptx is specified in Chapter 1 of the *Oracle8 Installation Guide for IBM DYNIX/ptx*.

## Demonstration Programs

Demonstration programs have been provided that show various functionality of the OCI. There are two types of demonstration programs: C and C++. All the demonstration programs are located in `$ORACLE_HOME/rdbms/demo`. Many of the demonstration programs assume that the demonstration tables created by `$ORACLE_HOME/sqlplus/demo/demobld.sql` exist in the SCOTT schema with the password TIGER.

For further information on building the demonstration programs using SQL\*Plus, see “Demonstration Tables” on page 4-3 of this book.

---

**See Also:** For further information on the demonstration programs see the *Programmer's Guide to the Oracle Call Interface* and the program source for details of each program.

---

The makefile, `$ORACLE_HOME/rdbms/demo/demo_rdbms.mk`, should be used to create the demonstration programs. For example, to compile and link the `cdemo1` demonstration program, enter the following command:

```
$ make -f demo_rdbms.mk cdemo1
```

Alternatively, the following command may be used, which achieves exactly the same result, only with more explicit syntax:

```
$ make -f demo_rdbms.mk build OBJS=cdemo1.o EXE=cdemo1
```

By default, all programs are dynamically linked with the client shared library, `$ORACLE_HOME/lib/libclntsh.so`.

To create all OCI C demonstration programs, enter the following command:

```
$ make -f demo_rdbms.mk demos
```

To create all OCI C++ demonstration programs, enter this command:

```
$ make -f demo_rdbms.mk c++demos
```

Some demonstration programs require a SQL script, located in `$ORACLE_HOME/rdbms/demo`, to be run manually prior to executing the program. In most cases, the SQL script name is the same as the program name with a `.sql` extension. For example, the SQL script for the program `oci02` is `oci02.sql`.

Read the comments at the beginning of the program to determine the required SQL script, if any.

## User Programs

The makefile, `$ORACLE_HOME/rdbms/demo/demo_rdbms.mk`, may be used to create user programs. The general syntax for linking a user program with `demo_rdbms.mk` is:

```
$ make -f demo_rdbms.mk target OBJS="objfile1 objfile2 ..." \
    EXE=exename
```

For example, to create the program `myprog` from the C source `myprog.c`, use one of the following commands depending on the type of executable desired:

For C source, dynamically linked with client shared library:

```
$ make -f demo_rdbms.mk build OBJS=myprog.o EXE=myprog
```

For C source, statically linked:

```
$ make -f demo_rdbms.mk build_static OBJS=myprog.o EXE=myprog
```

To create the program `myprog` from the C++ source `myprog.cc`

For C++ source, dynamically linked with client shared library:

```
$ make -f demo_rdbms.mk buildc++ OBJS=myprog.o EXE=myprog
```

For C++ source, statically linked:

```
$ make -f demo_rdbms.mk buildc++_static OBJS=myprog.o EXE=myprog
```

## Oracle Precompiler and Oracle Call Interface Linking and Makefiles

### Custom Makefiles

It is recommended that the provided `demo_product.mk` makefiles be used to link user programs as described in the specific product sections of this chapter. If it is necessary to modify the provided makefile, or if you decide to use a custom written makefile, the following should be noted:

- Do not modify the ordering of the Oracle libraries.  
Oracle libraries are included on the link line more than once so all symbols will be resolved during linking. There are two reasons for this:
  1. Oracle libraries are mutually referential, meaning that functions in library A call functions in library B, and functions in library B call functions in library A.
  2. The DYNIX/ptx linker is a one-pass linker, meaning that the linker will search a library exactly once at the point it is encountered in the link line.
- If you add your own library to the link line, it should be added to the beginning or to the end of the link line.  
User libraries should not be placed between the Oracle libraries.
- If you choose to use a make utility such as `nmake` or `GNU make`, you should be aware of how macro and suffix processing differs from the make utility provided with DYNIX/ptx. Oracle makefiles have been tested and are supported with the DYNIX/ptx make utility.
- Oracle library names and the contents of those libraries are subject to change between releases. Always use the `demo_product.mk` makefile that ships with the current release as a guide to determine which libraries are necessary.

## Undefined Symbols

A common error when linking a program is undefined symbols, similar to the following:

```
$ make -f demo_proc.mk sample1
Undefined                          first referenced
   symbol                          in file
sqlcex                             sample1.o
sqlglm                             sample1.o
ld: fatal: Symbol referencing errors. No output written to sample1
```

This error occurs when the linker cannot find a definition for a referenced symbol. Generally, the remedy for this type of problem is to ensure that the library or object file containing the definition exists on the link line and that the linker is searching the correct directories for the file.

Oracle provides a utility called `symfind` to assist in locating a library or object file where a symbol is defined. Here is example output of `symfind` locating the symbol `sqlcex`:

```
$ symfind sqlcex

SymFind - Find Symbol <sqlcex> in <*>.a, .o, .so
-----
Command:          /u01/app/oracle/product/8.0.6/bin/symfind sqlcex
Local Directory:  /u01/app/oracle/product/8.0.6
Output File:      (none)
Note:             I do not traverse symbolic links
                  Use '-v' option to show any symbolic links

Locating Archive and Object files ...
[11645] | 467572 | 44 | FUNC | GLOB | 0 | 8 | sqlcex
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
[35] | 0 | 44 | FUNC | GLOB | 0 | 5 | sqlcex
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
./lib/libcintsh.so
./lib/libsql.a
```

## Thread Support

The Oracle libraries provided with this release are not thread safe, and do not support multi-threaded applications.

## Static and Dynamic Linking with Oracle Libraries

Precompiler and OCI applications can be linked with Oracle Libraries either statically or dynamically. With static linking, the libraries and objects of the whole application are

linked together into a single executable program. As a result, application executables can become fairly large.

With dynamic linking, the executing code partly resides in the executable program, and also resides in libraries that are linked by the application dynamically at run-time. Libraries that are linked at run-time are called dynamic or shared libraries. There are two primary benefit of dynamic linking:

- 1. Smaller disk requirements**

Different applications, or different invocations of the same application, can use the same shared or dynamic library. As a result, the application disk requirements are reduced.

- 2. Smaller main memory requirements**

The same shared or dynamic library image (for example, the in-memory copy), can be shared by different applications. This means that a library needs to be loaded only once into the main memory and then multiple applications can use the same library. As a result, main memory requirements are reduced.

## Oracle Shared Library

The Oracle shared library is `$ORACLE_HOME/lib/libclntsh.so`. If the Oracle provided `demo_product.mk` makefile is used to link an application, the Oracle shared library is used by default.

It may be necessary to set the environment variable `LD_LIBRARY_PATH` so the runtime loader can find the Oracle shared library at process start-up. If you receive the following error when starting an executable, `LD_LIBRARY_PATH` must be set to the directory where the Oracle shared library exists:

```
% sample1
dynamic linker: sample1: open libclntsh.so.1.0: No such file or directory
```

Set `LD_LIBRARY_PATH` as follows:

```
% setenv LD_LIBRARY_PATH $ORACLE_HOME/lib
```

The Oracle shared library is created automatically during installation. If there is a need to recreate the Oracle shared library, exit all client applications using the Oracle shared library, including all Oracle client applications like SQL\*Plus and Recovery Manager, and run the following command logged in as the *oracle* user:

```
% cd $ORACLE_HOME/rdbms/lib
% make -f ins_rdbms.mk client_sharedlib
```

## Using Signal Handlers

This section describes signals Oracle8 uses for two-task communication, and explains how to set up your own signal handlers.

### Signals

Signals are installed in a user process when you connect to the database, and are de-installed when you disconnect.

Oracle8 uses the following signals for two-task communications:

**Table 5–4 Signals for Two-Task Communications**

SIGCONT	used by the pipe two-task driver to send out-of-band breaks from the user process to the <code>oracle</code> process.
SIGINT	used by all two-task drivers to detect user interrupt requests. SIGINT is not caught by <code>oracle</code> ; it is caught by the user process.
SIGPIPE	used by the pipe driver to detect end-of-file on the communications channel. When writing to the pipe, if no reading process exists, a SIGPIPE signal is sent to the writing process. SIGPIPE is caught by both the <code>oracle</code> process and the user process.
SIGCLD	used by the pipe driver. SIGCLD is similar to SIGPIPE, but only applies to user processes, not <code>oracle</code> processes. When an <code>oracle</code> process dies, the UNIX kernel sends a SIGCLD to the user process ( <code>wait()</code> is used in the signal handler to see if the server process died). SIGCLD is not caught by <code>oracle</code> ; it is caught by the user process.
SIGTERM	used by the pipe driver to signal interrupts from the user side to the <code>oracle</code> process. This occurs when the user presses the interrupt key [Ctrl]+[c]. SIGTERM is not caught by the user process; it is caught by <code>oracle</code> .
SIGIO	used by Oracle Net8 protocol adapters to indicate incoming networking events.
SIGURG	used by the Oracle Net8 TCP/IP drivers to send out-of-band breaks from the user process to the <code>oracle</code> process.

The listed signals affect Pro\*C or other precompiler applications. You can install one signal handler for SIGCLD (or SIGCHLD) and SIGPIPE when connected to the `oracle` process. You can have multiple signal handlers for SIGINT as long as the `osnsui()` routine is called to set this up. You can install as many signal handlers as you want for other signals. If you are not connected to the `oracle` process, you can have multiple signal handlers.

### Sample Signal Routine

The following example shows how you can set up your own signal routine and the catching routine. For SIGINT, use `osnsui()` and `osncui()` to register and delete signal-catching routines.

```
/* user side interrupt set */
word osnsui( /*_ word *handlp, void (*astp), char * ctx, _*/)
/*
** osnsui: Operating System dependent Network Set
**User-side
** Interrupt. Add an interrupt handling procedure
**astp.
```



```

** Whenever a user interrupt(such as a ^C) occurs,
**call astp
** with argument ctx. Put in *handlep handle for this
**handler so that it may be cleared with osncui.
** Note that there may be many handlers; each should
** be cleared using osncui. An error code is
**returned if an error occurs.
*/

/* user side interrupt clear */
word osncui( /*_ word handle _*/ );
/*
** osncui: Operating System dependent Clear User-side
**Interrupt.
** Clear the specified handler. The argument is the
**handle obtained from osnsui. An error code is
** returned if an error occurs.
*/

```

The following is a template for using `osnsui()` and `osncui()` in an application program:

```

/*
** My own user interrupt handler.
*/
void sig_handler()
{
...
}

main(argc, argv)
int arc;
char **argv;
{

    int handle, err;
    ...

    /* set up my user interrupt handler */
    if (err = osnsui(&handle, sig_handler, (char *) 0))
    {
        /* if the return value is non-zero, an error has occurred
        Do something appropriate here. */
        ...
    }
}

```

```
...
/* clear my interrupt handler */
if (err = osncui(handle))
{
/* if the return value is non-zero, an error has occurred
Do something appropriate here. */
...
}
...
}
```

## XA Functionality

When building a TP-monitor XA application, ensure that the TP-monitors libraries (that define the symbols `ax_reg` and `ax_unreg`) are placed in the link line before Oracle's client shared library. This link restriction is required only when using XA's dynamic registration (Oracle XA switch `xaoswd`).

The Oracle8 Server does not support Oracle7 7.1.6 XA calls (although it does support 7.3 XA calls), hence TP-monitor XA applications using 7.1.6 XA calls must be relinked with the Oracle8 XA library. The Oracle8 XA calls are defined in both the shared library `$ORACLE_HOME/lib/libclntsh.so` and the static library `$ORACLE_HOME/lib/libclient.a`.

---

# Configuring Oracle Net8

- » Supplemental Documentation
- » Core Net8 Products and Features
- » Oracle Net8 Protocol Adapters
  - » The BEQ Protocol Adapter
  - » The IPC Protocol Adapter
  - » The RAW Protocol Adapter
  - » The TCP/IP Protocol Adapter
  - » The SPX/IPX Protocol Adapter
- » Oracle Enterprise Manager (OEM) Intelligent Agent
- » Oracle Advanced Networking Option

# Supplemental Documentation

The following documents provide a full discussion of Oracle Net8 features:

- *Oracle Net8 Administrator's Guide*
- *Oracle Networking Quick Reference Card for Net8*
- *Oracle Advanced Networking Option Administrator's Guide*
- *Oracle Security Server Guide*
- *Oracle Cryptographic Toolkit Programmer's Guide*

## Supplementary Information in README Files

Table 6–1 shows the location of README files for various bundled products. The README files describe changes since the last release.

**Table 6–1 Location of README Files for Oracle Products**

Product	README File
Net8	<code>\$ORACLE_HOME/network/doc/README.Net8</code>
Advanced Networking Option	<code>\$ORACLE_HOME/network/doc/README.ANO</code>
Oracle Intelligent Agent	<code>\$ORACLE_HOME/network/doc/README.oemagent</code>
Oracle Security Server	<code>\$ORACLE_HOME/network/doc/README.Security</code>
Oracle Names Server	<code>\$ORACLE_HOME/network/install/names/doc/README.doc</code>
Oracle SPX/IPX Protocol Adapter	<code>\$ORACLE_HOME/network/install/spxpa/doc/README.doc</code>
Oracle TCP/IP Protocol Adapter	<code>\$ORACLE_HOME/network/install/tcppa/doc/README.doc</code>

# Core Net8 Products and Features

---

**See Also:** Sample files can be found in the *Oracle Net8 Administrator's Guide*.

---

## Net8 Files and Utilities

### Location of Net8 Configuration Files

The default directory for global Oracle Net8 and Connection Manager files is `/etc` on DYNIX/ptx.

Oracle Net8 and Connection Manager search for global files in the following order:

1. The directory specified by the environment variable, `TNS_ADMIN`, if set.
2. The `/etc` directory.
3. `$ORACLE_HOME/network/admin`.

If your files are not in the default directory, use the `TNS_ADMIN` environment variable in the startup files of all network users to specify a different location:

For the C shell, enter:

```
% setenv TNS_ADMIN new_default
```

For each system level configuration file, users may have a corresponding local private configuration file (stored in the user's home directory). The settings in the private file override the settings in the system level file. The private configuration file for `sqlnet.ora` is `$HOME/.sqlnet.ora`. The private configuration file for `tnsnames.ora` is `$HOME/.tnsnames.ora`. Syntax for these files is identical to that of the corresponding system files.

### Sample Configuration Files

Examples of the `cman.ora`, `listener.ora`, `names.ora`, `sqlnet.ora`, and `tnsnames.ora` configuration files are located in `$ORACLE_HOME/network/admin/samples`.

### The adapters Utility

To display installed Oracle Net8 adapters, enter:

```
% adapters
```

To display adapters linked with a specific executable, enter:

```
% adapters executable
```

For example, the following command displays the adapters linked with the `oracle` executable:

```
% adapters oracle
Protocol Adapters linked with oracle are:
  BEQ Protocol Adapter
  IPC Protocol Adapter
  TCP/IP Protocol Adapter
  RAW Protocol Adapter
Net8 Naming Adapters linked with oracle are:
  Oracle TNS Naming Adapter
  Oracle Naming Adapter
Advanced Networking Option/Network Security products linked with oracle are:
  Oracle Security Server Authentication Adapter
```

## Oracle Connection Manager

---

---

**See Also:** For information on the Oracle Connection Manager see the *Net8 Administrator's Guide*.

---

---

## Multi-Threaded Server

---

---

**See Also:** For information on the Multi-Threaded Server see the *Oracle8 Server Concepts* and *Oracle8 Administrator's Guide*.

---

---

## Oracle Names

---

---

**See Also:** For information on Oracle Names see the *Oracle Net8 Administrator's Guide*.

---

---

## Net8 Assistant

The Net8 Assistant (`$ORACLE_HOME/bin/net8asst.sh`) requires Ptx Server Environment Java® Platform (ptx/JSE) v1.1.0. To use Net8 Assistant, the ptx/JSE v1.1.0 must be installed on the system.

The environment variables `JRE_HOME`, `LD_LIBRARY_PATH` and `CLASSPATH` should be set properly to use Net8 Assistant.

If JSE is installed under `/opt/jse1.1`, to set the environment variables under `ksh`, use the following:

```
export JRE_HOME=/opt/jse1.1
export LD_LIBRARY_PATH=$JRE_HOME/lib/ptx/green_threads:
export CLASSPATH=$JRE_HOME/lib/classes.jar:
```

---

---

**See Also:** For further information on Net8 Assistant see the *Oracle Net8 Administrator's Guide*.

---

---

## Oracle Net8 Protocol Adapters

The supported Protocol Adapters for Net8 version 8.0.6 on DYNIX/ptx are BEQ Protocol Adapter, IPC Protocol Adapter, RAW Protocol Adapter, TCP/IP Protocol Adapter, SPX/IPX Protocol Adapter.

Prior to installing the TCP/IP or SPX/IPX Net8 Protocol Adapters, the appropriate operating system software must be installed and configured. Refer to the *Oracle8 Installation Guide for IBM DYNIX/ptx* for requirements details. The BEQ and IPC Net8 Protocol Adapters do not have any specific operating system requirement.

The IPC, TCP/IP, or SPX/IPX Net8 Protocol Adapters each have a protocol-specific ADDRESS specification that is used for Net8 configuration files and for the `MTS_LISTENER_ADDRESS` database initialization parameter (`init.ora`). See the ADDRESS specification heading under each Protocol Adapter section in this chapter for details.

Table 6–2 shows a summary of ADDRESS specifications for each Protocol Adapter.

**Table 6–2 ADDRESS Specification Summary**

Protocol Adapter	ADDRESS Specification
BEQ	(ADDRESS = (PROTOCOL = BEQ) (PROGRAM = ORACLE_HOME/bin/oracle) (ARGV0 = oracleORACLE_SID) (ARGS = ' (DESCRIPTION=(LOCAL=YES) (ADDRESS=(PROTOCOL=BEQ))) ' ) (ENVS = 'ORACLE_HOME=ORACLE_HOME,ORACLE_SID=ORACLE_SID' ) )
IPC	(ADDRESS = (PROTOCOL=IPC) (KEY=key) )
RAW	N/A
TCP/IP	(ADDRESS = (PROTOCOL=TCP) (HOST=hostname) (PORT=port_id) )
SPX/IPX	(ADDRESS = (PROTOCOL=SPX) (SERVICE=servicename) )



# The BEQ Protocol Adapter

## Overview of the BEQ Protocol Adapter

The BEQ Protocol Adapter, is both a communications mechanism and a process spawning mechanism. If a service name is not specified, either directly by the user on the command line or the login screen, or indirectly through an environment variable such as TWO\_TASK, then the BEQ Protocol Adapter will be used. In which case, a dedicated server will always be used, and the multi-threaded server will never be used. This dedicated server is started automatically by the BEQ Protocol Adapter, which waits for the server process to start and attach to an existing SGA. If the startup of the server process is successful, the BEQ Protocol Adapter then provides inter-process communication via UNIX pipes.

An important feature of the BEQ Protocol Adapter is that no network Listener is required for its operation, since the adapter is linked into the client tools and directly starts its own server process with no outside interaction. However, the BEQ Protocol Adapter can only be used when the client program and the Oracle8 server reside on the same machine. The BEQ Protocol Adapter is always installed, and always linked into all client tools and to the Oracle8 server.

## Specifying a BEQ ADDRESS

The BEQ Protocol Adapter connection parameters are part of the ADDRESS keyword-value pair. You can enter the parameters in any order.

```
(ADDRESS =  
  (PROTOCOL = BEQ)  
  (PROGRAM = ORACLE_HOME/bin/oracle)  
  (ARGV0 = oracleORACLE_SID)  
  (ARGS = '(DESCRIPTION=(LOCAL=YES)(ADDRESS=(PROTOCOL=BEQ)))')  
  (ENVS = 'ORACLE_HOME=ORACLE_HOME,ORACLE_SID=ORACLE_SID')  
)
```

Syntax for BEQ Protocol Adapter connection parameters is described in Table 6–3.

**Table 6–3 Syntax for BEQ Protocol Adapter Connection Parameters**

PROTOCOL	Specifies the adapter to be used. The value is <code>beq</code> and may be specified in either uppercase or lowercase.
PROGRAM	The full path to the oracle executable.
ARGV0	The name of the process as it appears in a <code>ps</code> listing. The recommended value is <code>oracleORACLE_SID</code> .
ARGS	<code>' (DESCRIPTION= (LOCAL=YES) (ADDRESS= (PROTOCOL=BEQ)) ) '</code>
ENVS	Environment specification where <code>ORACLE_HOME</code> is the full path to the <code>ORACLE_HOME</code> directory of the database to connect, and <code>ORACLE_SID</code> is the system identifier of the database to connect.

**Example 6–1 BEQ ADDRESS Specifying a Client**

The following is an example of an BEQ ADDRESS:

```
(ADDRESS =  
  (PROTOCOL = BEQ)  
  (PROGRAM = /u01/app/oracle/product/8.0.6/bin/oracle)  
  (ARGV0 = oracleV806)  
  (ARGS = ' (DESCRIPTION= (LOCAL=YES) (ADDRESS= (PROTOCOL=BEQ)) ) ' )  
  (ENVS = 'ORACLE_HOME=/u01/app/oracle/product/8.0.6,ORACLE_SID=V806' )  
)
```

The ADDRESS is commonly part of a larger construct such as a connect descriptor or configuration file.

# The IPC Protocol Adapter

## Overview of the IPC Protocol Adapter

The IPC Protocol Adapter, is similar to the BEQ Protocol Adapter in that it can only be used when the client program and the Oracle8 server reside on the same machine. The IPC Protocol Adapter differs from the BEQ Protocol Adapter in that it can be used with dedicated server and multi-threaded server configurations. The IPC Protocol Adapter requires a network listener for its operation. The IPC Protocol Adapter is always installed, and always linked in to all client tools and to the Oracle8 server.

For the IPC Protocol Adapter, the location of the UNIX Domain Socket (IPC) file on UNIX systems changed after Oracle7 version 7.1. Thus, if you have Oracle7 version 7.1 installed on the same machine as Oracle8, and you attempt to make an IPC connection between the two instances, the connection may fail. The solution to this problem is to make a symbolic link between the directory where the IPC file used to be (`/var/tmp/o`) and where it now resides (`/var/tmp/.oracle`).

## Specifying an IPC ADDRESS

The IPC Protocol Adapter connection parameters are part of the ADDRESS keyword-value pair. You can enter the parameters in any order.

```
(ADDRESS=
  (PROTOCOL=IPC)
  (KEY=key)
)
```

Syntax for IPC Protocol Adapter connection parameters is described in Table 6–4.

**Table 6–4 Syntax for IPC Protocol Adapter Connection Parameters**

PROTOCOL	Specifies the adapter to be used. The value is <code>ipc</code> and may be specified in either uppercase or lowercase.
KEY	Service name of database or database identifier (SID).

### Example 6–2 IPC ADDRESS Specifying a Client

The following is an example of an IPC ADDRESS:

```
(ADDRESS=
  (PROTOCOL=IPC)
  (KEY=PROD)
)
```

The ADDRESS is commonly part of a larger construct such as a connect descriptor or configuration file.

## The RAW Protocol Adapter

When data is transferred back and forth between a client and a server, Net8 adds its own header information onto every packet (a block of information sent over the network). Through the Raw Transport feature, Net8 can now minimize transmitting header information on each packet going over the network.

After the connection is established, two types of information flow over the network: data and break handling. The connection packets need to have the Net8 header information on them to establish the connection correctly. However, after the connection is established, all data packets for sending or receiving data, or packets for breaking or resetting the connection, are stripped of their Net8 header information and passed directly to the operating system, bypassing Net8 NT and Oracle Protocol Adapter layers. The performance of the connection is increased because of fewer protocol stack layers for the data to flow through and fewer bytes that are transmitted over the network.

This feature is transparently turned on whenever it is appropriate. That is, if no existing features require that header information be transmitted, the headers are stripped off. For example, encryption and authentication require certain information to be sent along with each packet of information; so, raw transport would not be turned on in this scenario.

This feature requires no configuration to be enabled. Net8 determines if the conditions are met and then transparently switches to Raw Transport mode.

# The TCP/IP Protocol Adapter

## Overview of the TCP/IP Protocol Adapter

Oracle Corporation recommends that you reserve a port for your Oracle Net8 listener in the `/etc/services` file of each node on the network that defines the Oracle Net8 listener port. The port is commonly 1521. The entry should list the listener name and the port number, for example:

```
listener      1521/tcp
```

where *listener* is the name of the listener, as defined in `listener.ora`.

Reserve more than one port to start more than one listener.

## Specifying a TCP/IP ADDRESS

The TCP/IP Protocol Adapter connection parameters are part of the ADDRESS keyword-value pair. You can enter the three parameters in any order.

```
(ADDRESS=
  (PROTOCOL=TCP)
  (HOST=hostname)
  (PORT=port_id)
)
```

Syntax for TCP/IP Protocol Adapter connection parameters is described in Table 6–5.

**Table 6–5 Syntax for TCP/IP Protocol Adapter Connection Parameters**

PROTOCOL	Specifies the adapter to be used. The value can be uppercase or lowercase. The default is <code>tcp</code> .
HOST	The host name or the host IP address.
PORT	The TCP/IP port. Either a number or the name specified in the <code>/etc/services</code> file. Oracle Corporation recommends a value of 1521.

### Example 6–3 TCP/IP ADDRESS Specifying a Client

Following is an example of the TCP/IP ADDRESS specifying a client on the MADRID host:

```
(ADDRESS=
  (PROTOCOL=TCP)
  (HOST=MADRID)
  (PORT=1521)
```

)

The last field could be specified by name, for example, (PORT=listener). The ADDRESS is commonly part of a larger construct such as a connect descriptor or configuration file.

## The SPX/IPX Protocol Adapter

This section describes the SPX/IPX Protocol Adapter.

### The `ntisbsd` Broadcast Daemon

A client uses a name and translates the name into an SPX address to identify a server and communicate with it. The netware bindery is a directory service that provides the translation mechanism. When a server is registered with the bindery, it periodically notifies the bindery of its address. This is done using the Server Advertising Protocol (SAP).

The server broadcasts a SAP packet in an IPX datagram every 60 seconds. This SAP packet contains all relevant addressing information. Any client can then query its nearest server for the address of the required server.

The Oracle SPX/IPX Protocol Adapter broadcasts using the `ntisbsd` broadcast daemon in `$ORACLE_HOME/bin`. The `ntspctl` utility starts and stops `ntisbsd`.

### The `ntspctl` Utility

The `ntspctl` utility contains functions to register and remove names, and to query a bindery. It can also be used to stop and start the broadcast daemon. (The `listener` automatically uses the daemon to auto-register service names in use.)

Example 6-4 demonstrates several uses of the `ntspctl` utility.

#### **Example 6-4** *Using the `ntspctl` Utility*

The `ntspctl` utility reads commands from the command line. If parameters are missing, it prompts for them.

To start `ntspctl`, enter:

```
$ ntspctl
```

Output similar to the following is displayed:

```
ntspxctl for IBM DYNIX/ptx: Version 8.0.6.0.0 -  
Production on Fri Jul 3 11:41:40 2000
```

To start the broadcast daemon, enter:

```
ntspxctl> startup
```

Output similar to the following is displayed:

```
ntisbsdsm started at Fri Jul 3 11:43:47 2000
```

A system message is displayed if the daemon has already been started.

Startup of the broadcast daemon should be automated, so it is always started when the machine is started. Automate daemon startup by adding an entry to the `/etc/inittab` file. For example, to start the `ntisbsdsm` on system startup add the following line to `/etc/inittab` :

```
ntspxctl:2:once:/u/oracle/bin/ntisbsdsm &
```

where `/u/oracle` is the full path to `$ORACLE_HOME`.

To register a name for testing, enter `register` and the name:

```
ntspxctl> register test
```

This creates a socket owned by `ntisbsdsm`, and registers it.

A message similar to the following is displayed:

```
Name test successfully registered  
test address 00eee045:000000000001:4454
```

To check the status of `ntisbsdsm`, enter:

```
ntspxctl> status
```

A message similar to the following is displayed:

```
ntisbsdsm started at Fri Jul 3 11:43:47 2000  
Tracing is off  
Pid: 14784 test
```

A shorter status can be obtained by entering:

```
ntspxctl> summary
```

A message similar to the following is displayed:

```
ntisbsdsm started at Fri Jul 3 11:43:47 20000
Tracing is off
1 names are registered
```

## SPX/IPX Protocol Adapter Command Summary

Table 6–6 shows the help command summary for the SPX/IPX Protocol Adapter.

**Table 6–6** *help Command Summary*

<code>register name</code>	Register entry.
<code>remove name</code>	Remove entry.
<code>shutdown [force]</code>	Shut down ntisbsdsm.
<code>startup</code>	Get status summary.
<code>traceon</code>	Activate trace.
<code>traceoff</code>	Deactivate trace.
<code>status</code>	Get full status.
<code>getname name / hex_number</code>	Query name services.
<code>exit</code>	Exit program.
<code>help [command]</code>	Print command information.
<code>!</code>	Shell escape.

## The getname Command

The getname command asks the Novell system for names. It does not involve the broadcast daemon.

Enter:

```
getname name servicetype
```

A message similar to the following is displayed:

```
getname name servicetype (address number_of_hops)
```



The syntax for the `getname` command is explained in Table 6–7.

**Table 6–7 Syntax for the `getname` Command**

<code>name</code>	The name you entered.
<code>servicetype</code>	A number assigned by Novell. Oracle has the number 103.
<code>address</code>	The address of the name you entered.
<code>number_of_hops</code>	The number of hops to the destination, displayed in hexadecimal. The value 10 means the name is deregistered. If SAP queries are not supported, the value is 0000.

To see all possible names, enter:

```
getname * *
```

Example 6–5 shows names obtained using the `getname` command.

**Example 6–5 Using the `getname` Command**

```
ntspxctl> getname test*
test servertype x0103 address 00eee045:000000000001:
    4465 hops 0000
ntspxctl> getname * 103
LSNR servertype x0103 address 00eee053:000000000001:
    502c hops 0000
IBM6 servertype x0103 address 00eee058:000000000001:
    507f hops 0000
DESK servertype x0004 address 00eee055:000000000001:
    5451 hops 0000
DESK servertype x0107 address 00eee055:000000000001:
    5104 hops 0000
CXY4 servertype x009e address 00eee055:000000000001:
    5063 hops 0000
IBM2 servertype x0004 address 00eee057:000000000001:
    5451 hops 0000
```

To stop `ntisbsdsm`, enter:

```
ntspxctl> shutdown
```

The daemon will not be stopped if names are still registered. A message similar to the following is displayed:

```
1 names are registered
```

```
ntisbsdsm not stopped
```

To remove a name, enter `remove` and the name. Following is an example for the name `test`:

```
ntspxctl> remove test
```

A message similar to the following is displayed:

```
Name testremoved.  
ntspxctl> shutdown  
ntisbsdsm stopped
```

To force a stop, enter:

```
ntspxctl> shutdown force
```

A message similar to the following is displayed:

```
ntisbsdsm stopped
```

## Specifying the SPX/IPX ADDRESS

After the SPX/IPX protocol and Oracle SPX/IPX Protocol Adapter are installed on your system, you can use the SPX/IPX parameters with the TNS connect descriptors to identify SPX/IPX community nodes.

The SPX/IPX Protocol Adapter parameters are part of the ADDRESS keyword-value pairs.

```
(ADDRESS=  
  (PROTOCOL=SPX)  
  (SERVICE=servicename)  
)
```

Table 6–8 explains the syntax for the SPX/IPX Protocol Adapter connection.

**Table 6–8 Syntax for SPX/IPX Protocol Adapter Connection**

PROTOCOL	Specifies the adapter name. For SPX/IPX, the value is <code>spx</code> .
SERVICE	A unique name (up to 30 characters) identifying an application on the network. The service is named during startup and is available to the entire network. Client references to the service are made using lookup in the bindery, a network directory.

Example 6–6 shows an SPX/IPX ADDRESS specifying service MAILDB1 on a remote server.

**Example 6–6** SPX/IPX Protocol Adapter Connection

```
(ADDRESS=
  (PROTOCOL=SPX)
  (SERVICE=MAILDB1)
)
```

This ADDRESS is commonly part of a larger construct such as a connect descriptor or configuration file.

## Oracle Enterprise Manager (OEM) Intelligent Agent

### Agent Service Discovery and Auto-Configuration

---

---

**See Also:** The *Oracle Enterprise Manager Configuration Guide*.

---

---

### Debugging tcl Scripts

The executable `oratclsh` is provided for debugging your `tcl` scripts. Before executing `oratclsh`, set the environment variable `TCL_LIBRARY` to point to `$ORACLE_HOME/network/agent/tcl`.

---

---

**See Also:** The *Oracle Enterprise Manager Application Developer's Guide* for additional details.

---

---

## Oracle Advanced Networking Option

### **.bak Files**

During Oracle Advanced Networking Option installation, three .bak files are created: naeet.o.bak, naect.o.bak, and naedhs.o.bak. They are located in \$ORACLE\_HOME/lib. These files are required for relinking during Oracle Advanced Networking Option deinstall and should not be deleted.

### **Security and Single Sign-On**

---

---

**See Also:** For details on configuring Security and Single Sign-On, see the *Oracle Advanced Networking Option Administrator's Guide*, Release 8.0.

---

---

---

# Running Oracle Data Cartridge Demos on DYNIX/ptx

- n Issues Common to Data Cartridges
- n Oracle8 Time Series Cartridge
- n Oracle8 Visual Information Retrieval Cartridge
- n Oracle8 Image Cartridge

## Issues Common to Data Cartridges

Data cartridges require a C compiler to build their sample applications after installation.

---

---

**See Also:** For more information about Oracle data cartridges, refer to the *Oracle Enterprise Manager Configuration Guide* and the documentation for the individual cartridge.

---

---

## Oracle8 Time Series Cartridge

---

---

**See Also:** For additional information, refer to the *Oracle Time Series Cartridge User's Guide*.

---

---

## Installing Time Series Cartridge Demos

Time Series Cartridge demo files are stored in `$ORACLE_HOME/ord/ts/demo`. This directory contains several demos, each in its own subdirectory:

- The `usage` demo shows how to build a time series object and call time series functions. This `usage` demo should be run first, to create sample time series schemas used by the other demos.
- The `extend` demo shows how to write and add customized time series functions.
- The `proc` demo demonstrate accessing data stored with the cartridge using Pro\*C.
- The `oci` demo demonstrates access data via the Oracle Call Interface.
- The `dev2k` demo includes a Developer 2000 form which retrieves data using the cartridge.

Refer to the `README` files in the demo directories for more information about each demo.

## Oracle8 Visual Information Retrieval Cartridge

---

---

**See Also:** For additional information, refer to the *Oracle Visual Information Retrieval Cartridge User's Guide*.

---

---

## Creating the Demo

1. Create the Visual Information Retrieval (VIR) demo directory.

```
% svrmgrl
SVRMGRL> connect internal;
SVRMGRL> create or replace directory virdemodir
          as '$ORACLE_HOME/ord/vir/demo';
```

**2. Grant privileges on the directory to PUBLIC.**

```
SVRMGRL> grant read on directory virdemodir to public
          with grant option;
SVRMGRL> exit;
```

**3. Make the virdemo program.**

```
% cd $ORACLE_HOME/ord/vir/demo
% make -f demo_ordvir.mk virdemo
```

## Oracle8 Image Cartridge

---

---

**See Also:** For additional information, refer to the *Oracle Image Cartridge User's Guide*.

---

---

### Creating the Demo

1. Create the Image Cartridge demo directory.

```
% svrmgrl
SVRMGR> connect internal;
SVRMGR> create or replace directory imgdemodir
        as '$ORACLE_HOME/ord/img/demo';
```

2. Grant privileges on the directory to PUBLIC.

```
SVRMGR> grant read on directory imgdemodir to public
        with grant option;
SVRMGR> exit;
```

3. Make the imgdemo program.

```
% cd $ORACLE_HOME/ord/img/demo
% make -f demo_ording.mk imgdemo
```



---

---

# Index

## Symbols

---

\$  
    using, 2-25  
?  
    example of use, 2-10  
@  
    to denote \$ORACLE\_SID, 2-10  
\_editor  
    setting, 4-6

## A

---

adapters utility, 6-3  
ADDRESS specification  
    protocol adapters, 6-6  
administering  
    SQL, 4-2  
advanced networking option, 6-18  
    .bak files, 6-18  
    security and single sign-on, 6-18  
afiedt.buf, 4-7  
asynchronous I/O, 3-13  
    using, 3-13  
automatic login  
    listener.ora file, 2-25  
    non-UNIX systems, 2-25  
    remote\_os\_roles, 2-29

## B

---

BEQ protocol adapter, 6-7  
    ADDRESS, 6-7  
    overview, 6-7

    syntax for connection parameters, 6-8  
binding processes, 3-19  
BSD, 2-12  
buffer cache size  
    tuning, 3-27  
buffer manager, 3-4  
buffers  
    tuning, 3-6

## C

---

C  
    Pro\*C/C++, 5-4  
cache  
    size  
        tuning, 3-27  
cartridge demos  
    Image Cartridge, 7-4  
    installing for Time Series, 7-2  
    Visual Information Retrieval Cartridge, 7-2  
catching routine  
    example, 5-18  
CATPROC.SQL, 1-16  
CDE tools, 2-9  
changing databases, 2-6  
chmod command  
    example, 2-24  
cluster  
    estimating size, 2-14  
COBOL  
    Pro\*COBOL, 5-6  
command interpreter, 2-12  
commands  
    expst, 3-20

- impst, 3-20
- ipcs, 3-6
- orapwd, 2-28
- sar, 3-4, 3-5
- vmstat, 3-4, 3-5
- common environment
  - oraenv file, 2-5
  - setting, 2-5
- configuration files
  - Net8, 6-3
  - precompiler, 5-3
- CONNECT INTERNAL
  - security, 2-24
- control remote DBA access
  - REMOTE\_DBA\_OPS\_ALLOWED, 2-26
  - REMOTE\_DBA\_OPS\_DENIED, 2-26
- control remote login, 2-26
  - OPS\_DOLLAR\_LOGIN\_DENIED, 2-26
- COPY command
  - SQL\*Plus, 4-8
- coraenv, 2-9
- CPU usage
  - priority level of processes, 3-19
  - processor binding, 3-19
  - single-task architecture, 3-20
  - tuning, 3-19
- cursors
  - tuning session cache, 3-10

## D

---

- daemon user
  - security, 2-26
- data cartridges
  - running demos for, 7-1
- data dictionary cache
  - tuning, 3-9
- database
  - administrator
    - permissions for executables, 2-22
  - files
    - authorization, 2-24
    - security, 2-24
    - tuning buffers, 3-6
  - database I/O
    - DBWR tuning, 3-12
  - database writer
    - tuning, 3-12
  - dba group
    - members, 2-22
    - relinking, 2-23
  - DBA group ID
    - keywords, 2-26
    - non-default name, 2-26
  - DBA\_GROUP
    - /etc/listener.ora file, 2-26
  - DBA\_GROUP\_sid
    - /etc/listener.ora file, 2-26
  - DBWR
    - tuning, 3-12
  - debugger programs, 5-3
  - default
    - directory, 2-11, 2-12
    - linker option, 2-11
    - printer, 2-11, 2-12
  - default settings
    - system editor, 4-7
  - demonstration
    - precompiler, 2-33
    - the procedural option, PL/SQL, 2-32
  - demonstration programs
    - oracle call interface, 5-12
    - Pro\*C/C++, 5-4
    - Pro\*COBOL, 5-8
  - demonstration tables
    - creating manually, 4-4
    - deleting, 4-4
    - SQL\*Plus, 4-3
  - demos
    - creating for Image Cartridge, 7-4
    - creating for Visual Information Retrieval Cartridge, 7-2
    - installing for Time Series Cartridge, 7-2
  - disk
    - monitoring performance, 3-18
    - quotas, 2-15
  - disk I/O
    - asynchronous I/O, 3-13
    - I/O slaves, 3-13
    - request queues, 3-15

- tuning, 3-11
- tuning for parallel server, 3-30
- tuning the database writer, 3-12
- disk performance
  - issues, 3-18
- DISPLAY
  - environment variable, 2-11
- Distributed Lock Manager
  - tuning, 3-29
- DLM
  - tuning, 3-29
- driver
  - post-wait, 3-20
- dynamic and static linking
  - oracle libraries, 5-15

## E

---

- echo command, 2-4
- editor
  - setting the order of, 4-6
  - SQL\*Plus, 4-6
- environment variables, 5-7
  - DISPLAY, 2-11
  - HOME, 2-11
  - LANG, 2-11
  - LANGUAGE, 2-11
  - LD\_LIBRARY\_PATH, 2-11
  - LDOPTS, 2-11
  - LDPATH, 2-11
  - LPDEST, 2-11
  - MicroFocus COBOL compiler, 5-7
  - NLS\_LANG, 2-8
  - ORA\_NLS, 2-8
  - ORACLE\_HELP, 2-8
  - ORACLE\_HOME, 2-8
  - ORACLE\_SID, 2-9
  - ORACLE\_TERM, 2-9
  - ORACLE\_TERMINAL, 2-9
  - ORACLE\_TRACE, 2-9
  - ORAENV\_ASK, 2-9
  - PATH, 2-11
  - PRINTER, 2-12
  - Pro\*COBOL, 5-7
  - setting for SQL\*Plus, 4-6

- SHELL, 2-12
- TERM, 2-12
- TMPDIR, 2-12
- TNS\_ADMIN, 2-10, 6-3
- TWO\_TASK, 2-10
- TZ, 2-13
- XENVIRONMENT, 2-12
- exclamation point
  - SQL\*Plus prompt, 4-7
- executable program, 2-11
- exports
  - tuning, 3-20
- expst command, 3-20
- extent fragmentation, 3-16

## F

---

- file names
  - default extensions in SQL\*Plus, 4-8
- files
  - trace files, 3-31
- FORMAT precompiler
  - Pro\*COBOL, 5-9, 5-10
- fragmentation
  - tuning, 3-16
- free list
  - tuning, 3-29

## G

---

- getname command, 6-14
- glogin.sql, 4-2
- groups
  - sample script, 2-31

## H

---

- help facility
  - SQL\*Plus, 4-4
- help file, 2-8
- hit ratio
  - for buffer cache, 3-6
- HOME, 2-11
- home directory, 2-11

## I

---

### I/O

- DBWR tuning, 3-12
- disk request queues, 3-15
- tuning, 3-11
- tuning for parallel server, 3-30

### I/O slaves, 3-13

### Image Cartridge, 7-4

- creating the demo for, 7-4

### imports

- tuning, 3-20

### impst command, 3-20

### index size

- calculating, 2-14

### indexes

- tuning, 3-14, 3-29

### initialization parameters

- BACKGROUND\_DUMP\_DEST, 2-16
- BITMAP\_MERGE\_AREA\_SIZE, 2-16, 2-18
- COMMIT\_POINT\_STRENGTH, 2-16
- CONTROL\_FILES, 2-16
- CREATE\_BITMAP\_AREA\_SIZE, 2-16, 2-18
- DB\_BLOCK\_BUFFERS, 2-16, 2-18
- DB\_BLOCK\_SIZE, 2-16
- DB\_FILE\_DIRECT\_IO\_COUNT, 2-16
- DB\_FILE\_MULTIBLOCK\_READ\_COUNT, 2-16
- DB\_FILES, 2-16
- defaults, 2-15
- DISTRIBUTED\_TRANSACTIONS, 2-16
- HASH\_AREA\_SIZE, 2-16
- HASH\_MULTIBLOCK\_IO\_COUNT, 2-16
- LOCK\_SGA, 2-16
- LOCK\_SGA\_AREAS, 2-16
- LOG\_ARCHIVE\_BUFFER\_SIZE, 2-16
- LOG\_ARCHIVE\_BUFFERS, 2-16
- LOG\_ARCHIVE\_DEST, 2-16
- LOG\_ARCHIVE\_FORMAT, 2-16
- LOG\_BUFFER, 2-16
- LOG\_CHECKPOINT\_INTERVAL, 2-16
- LOG\_SMALL\_ENTRY\_MAX\_SIZE, 2-16
- MTS\_LISTENER\_ADDRESS, 2-17
- MTS\_MAX\_DISPATCHERS, 2-16
- MTS\_MAX\_SERVERS, 2-16
- MTS\_SERVERS, 2-16

### NLS\_LANGUAGE, 2-17

### NLS\_TERRITORY, 2-17

### OBJECT\_CACHE\_MAX\_SIZE\_PERCENT, 2-17

### OBJECT\_CACHE\_OPTIMAL\_SIZE, 2-17

### OPEN\_CURSORS, 2-17

### OS\_AUTHENT\_PREFIX, 2-17

### PROCESSES, 2-17

### REMOTE\_OS\_AUTHENT, 2-25

### SHARED\_POOL\_SIZE, 2-17

### SHOW PARAMETERS command, 2-15

### SORT\_AREA\_SIZE, 2-17

### SORT\_READ\_FAC, 2-17

### SORT\_SPACEMAP\_SIZE, 2-17

### USER\_DUMP\_DEST, 2-17

### input and output, 2-11

### interrupting SQL\*Plus, 4-8

### IPC protocol adapter, 6-9

### ADDRESS, 6-9

### overview, 6-9

### ipcs command, 3-6

### ireclen, 5-3

## K

---

### kernel

- tuning UNIX parameters, 3-21

### keywords

- DBA group ID, 2-26

## L

---

### LANGUAGE

- environment variable, 2-11

### language, 2-8

### latches

- contention, 3-24

### ld, 2-11

### LD\_LIBRARY\_PATH

- environment variable, 2-11

### LDOPTS

- environment variable, 2-11

### LDPATH

- environment variable, 2-11

### library cache

- tuning, 3-9

- limits
  - resource, 2-15
- linking
  - single-task, 3-20
- locks
  - tuning for parallel server, 3-29
- Logical Volume Manager, 3-11
- logical volumes
  - tuning, 3-11
- login home directories
  - administering, 2-30
  - sample script, 2-31
- login.sql, 4-2
- LPDEST
  - environment variable, 2-11

## M

---

- MAXDATAFILES parameter, 2-37
- MAXINSTANCES parameter, 2-37
- MAXLOGFILES parameter, 2-37
- MAXLOGHISTORY parameter, 2-37
- MAXLOGMEMBERS parameter, 2-37
- memory
  - estimating usage, 2-13
  - SGA tuning, 3-5
  - shared, 2-19
  - tuning, 3-4
  - virtual, 2-13
- memory management, 3-4
  - control paging, 3-5
  - single shared memory segment, 3-5
  - swap space, 3-4
  - UNIX kernel, 3-5
- MicroFocus COBOL compiler, 5-7
- multiple databases
  - shared password file, 2-29
- multiple signal handlers, 5-18
- multi-thread server, 6-4

## N

---

- National Language Support (NLS)
  - variable, 2-8
- Net8

- adapters utility, 6-3
- advanced networking option, 6-18
- automatic logins, 2-25
- BEQ protocol adapter, 6-7
- files and utilities, 6-3
- IPC protocol adapter, 6-9
- multi-thread server, 6-4
- Net8 assistant, 6-5
- oracle connection manager, 6-4
- oracle enterprise manager intelligent agent, 6-17
- oracle names, 6-4
- products and features, 6-3
- protocol adapters, 6-5
- RAW protocol adapter, 6-10
- README files, 6-2
- SPX/IPX protocol adapter, 6-12
- TCP/IP protocol adapter, 6-11
- Net8 assistant, 6-5
- Net8 configuration files
  - location of, 6-3
- network
  - dba privileges, 2-25
  - passwords, 2-25
  - security, 2-25
- NLS\_LANG
  - environment variable, 2-8
- ntisbsdmd broadcast daemon, 6-12
- ntspxtl utility, 6-12

## O

---

- operating system commands
  - running from SQL\*Plus, 4-7
- ORA\_NLS
  - environment variable, 2-8
- Oracle
  - memory usage, 2-14
- oracle account
  - set authorization, 2-24
- oracle call interface, 5-12
  - demonstration programs, 5-12
  - user programs, 5-13
  - using, 5-12
- oracle connection manager, 6-4
- oracle enterprise manager intelligent agent, 6-17

- agent service discovery and auto-configuration, 6-17
- debugging tcl scripts, 6-17
- Oracle environment variables
  - EPC\_DISABLED, 2-8
  - NLS\_LANG, 2-8
  - ORA\_NLS33, 2-8
  - ORACLE\_BASE, 2-8
  - ORACLE\_HELP, 2-8
  - ORACLE\_HOME, 2-8
  - ORACLE\_PATH, 2-8
  - ORACLE\_SID, 2-9
  - ORACLE\_TERM, 2-9
  - ORACLE\_TERMINAL, 2-9
  - ORACLE\_TRACE, 2-9
  - ORAENV\_ASK, 2-9
  - TNS\_ADMIN, 2-10
  - TWO\_TASK, 2-10
- oracle group
  - permissions and executables, 2-23
- oracle libraries
  - oracle shared library, 5-17
  - static and dynamic linking, 5-15
- oracle names, 6-4
- Oracle Parallel Server
  - monitoring, 3-30
  - tuning, 3-29
- oracle precompiler and OCI linking and makefiles, 5-14
  - custom makefiles, 5-14
  - undefined symbols, 5-15
- oracle run time system
  - Pro\*COBOL, 5-8
- Oracle Server
  - accounts, 2-21
- oracle software owner, 2-21
  - special accounts, 2-21
- Oracle System Identifier, 2-9
- ORACLE\_HELP
  - environment variable, 2-8
- ORACLE\_HOME
  - environment variable, 2-8
  - using ? for, 2-10
- ORACLE\_SID
  - environment variable, 2-9
  - suppressing prompt, 2-6
- ORACLE\_TERM

- environment variable, 2-9
- oraenv file
  - description, 2-5
  - moving between databases, 2-6
- ORAENV\_ASK, 2-9
  - setting, 2-6
- orainst, 2-9
- orapwd command, 2-28
- oreclen, 5-3

## P

---

- paging space
  - tuning, 3-4, 3-5
- parallel queries
  - tuning, 3-25
- parallel server
  - monitoring, 3-30
  - tuning, 3-29
- password files
  - shared, 2-29
- passwords
  - remote, 2-28
- PATH, 2-11
- performance
  - table striping, 3-16
- permissions
  - dba group, 2-22
  - granting, 2-23
- PL/SQL
  - demonstrations
    - loading, 2-32
  - large blocks, 3-10
- pool size
  - tuning, 3-8
- post-wait driver
  - tuning, 3-20
- precompiler
  - configuration files, 5-3
  - overview, 5-2
  - relinking executables, 5-2
  - running demonstration, 2-33
  - signals, 5-18
  - uppercase to lowercase conversion, 5-3
  - value of ireclen and oreclen, 5-3

- values, 5-3
- vendor debugger programs, 5-3

## preface

- Send Us Your Comments, ix

## PRINTER, 2-12

## Pro\*C/C++

- administering, 5-4
- demonstration programs, 5-4
- makefiles, 5-5, 5-6
- signals, 5-18
- system configuration file, 5-4
- user programs, 5-6
- using, 5-4

## Pro\*COBOL, 5-6

- administering, 5-7
- demonstration programs, 5-8
- environment variables, 5-7
- FORMAT precompiler, 5-9, 5-10
- naming differences, 5-6
- oracle run time system, 5-8
- system configuration file, 5-7
- user programs, 5-9

## PRODUCT\_USER\_PROFILE Table

- SQL\*Plus, 4-3

## protocol adapters, 6-5

- ADDRESS specification, 6-6

## pupbld.sql, 4-3

# Q

---

## queries

- tuning parallel, 3-25

## question mark

- example of use, 2-10

# R

---

## raw devices, 3-32

- buffer cache size, 3-27
- criteria for using, 3-32
- disadvantages, 3-32
- guidelines for using, 3-33
- oracle8 parallel server installation, 3-32
- raw disk partition availability, 3-32, 3-33
- setting up, 3-33

## RAW protocol adapter, 6-10

## README files

- Net8, 6-2

## redo buffers

- tuning, 3-8

## redo log

- buffer latch contention, 3-25
- raw devices, 3-15
- tuning, 3-15
- tuning buffer size, 3-8

## related documentation, xiv

## relinking precompiler executables, 5-2

## remote

### connect

- as INTERNAL, 2-29
- as OPERATOR, 2-29

### login

- security, 2-26

### parameter

- REMOTE\_OS\_AUTHENT, 2-25

## remote connections parameters

- OS\_AUTHENT\_PREFIX, 2-29
- REMOTE\_OS\_AUTHENT, 2-29
- REMOTE\_OS\_ROLES, 2-29

## resource contention

- kernel parameters, 3-21
- tuning, 3-20

## resource definition, 2-12

## resource limits, 2-15

## restrictions (SQL\*Plus), 4-8

- COPY command, 4-8

## rollback segments

- contention, 3-24

## root

- user, 2-21

# S

---

## sar command, 3-2, 3-4, 3-5, 3-28

## security

- assigning permissions, 2-22
- CONNECT INTERNAL, 2-24
- default group names, 2-23
- file ownership, 2-22
- group accounts, 2-22

- network, 2-25
- Server Manager access, 2-24
- SHUTDOWN command, 2-24
- STARTUP command, 2-24
- two-task architecture, 2-22
- semaphores
  - replacing with post-wait driver, 3-20
- Send Us Your Comments, ix
- Server Manager
  - commands, 2-24
  - security, 2-24
  - SHOW PARAMETERS, 2-15
- session cache cursors
  - tuning, 3-10
- setting up
  - raw devices, 3-33
- setup files
  - SQL\*Plus, 4-2
- SGA
  - estimating size, 3-5
  - relocating, 2-20
  - tuning, 3-5
- shadow process
  - security, 2-22
- shared
  - library loader, 2-11
  - object library, 2-11
- shared memory
  - SGA, 2-19
  - SGA tuning, 3-5
- shared object library, 2-11
- shared pool size
  - tuning, 3-8
- SHELL, 2-12
- SHUTDOWN command, 2-24
  - security, 2-24
- SIGCLD two-task signal, 5-18
- SIGINT two-task signal, 5-18
- SIGIO two-task signal, 5-18
- signal handlers
  - signals, 5-17
  - using, 5-17
- signal routine
  - example, 5-18
- signals
  - creating handlers, 5-17
  - two-task, 5-17
- SIGPIPE two-task signal, 5-18
- SIGTERM two-task signal, 5-18
- SIGURG two-task signal, 5-18
- single shared memory segment, 3-5
- site profile
  - SQL\*Plus, 4-2
- software distribution, 2-8
- special accounts, 2-21
- special groups
  - root, 2-22
- SPOOL command
  - SQL\*Plus, 4-8
  - using, 4-8
- SPX/IPX protocol adapter, 6-12
  - ADDRESS, 6-16
  - command summary, 6-14
  - getname command, 6-14
  - ntisbsdsm broadcast daemon, 6-12
  - ntspxtl utility, 6-12
- SQL
  - administering, 4-2
- SQL scripts, 3-4
  - utlstat and utlestat, 3-4
- SQL\*DBA, 2-9
  - SHOW PARAMETERS, 2-15
- SQL\*Plus
  - COPY command, 4-8
  - default editor, 4-6
  - demonstration tables, 4-3
  - editor, 4-6
  - environment variables, 4-6
  - help facility, 4-4
  - interrupting, 4-8
  - PRODUCT\_USER\_PROFILE Table, 4-3
  - restrictions, 4-8
  - sample setup files, 4-3
  - setup files, 4-2
  - site profile, 4-2
  - SPOOL command, 4-8
  - system editor, 4-6
  - UNIX commands, 4-7
  - user profile, 4-2
  - using, 4-6



- STARTUP command
  - security, 2-24
- static and dynamic linking
  - oracle libraries, 5-15
- sub-shell
  - creating in SQL\*Plus, 4-7
- superuser, 2-21
- swap, 3-3
- swap space
  - tuning, 3-4
- SYS account
  - privileges, 2-21
- SYSDATE
  - and TZ, 2-13
- SYSTEM account
  - privileges, 2-21
- system configuration file
  - Pro\*C/C++, 5-4
  - Pro\*COBOL, 5-7
- system editor
  - default settings, 4-7
  - setting the order of, 4-6
  - SQL\*Plus, 4-6
- System Global Area (SGA)
  - relocating, 2-20
  - requirements, 2-19
- system time
  - setting, 2-13
- System V, 2-11

## T

---

- table striping, 3-11
- tables
  - storing data in separate files, 3-16
  - striping, 3-16
- tablespaces
  - fragmentation, 3-17
- TCP/IP protocol adapter, 6-11
  - ADDRESS, 6-11
  - overview, 6-11
- temporary disk file, 2-12
- TERM
  - environment variable, 2-12
- thread support, 5-15
- Time Series Cartridge, 7-2
  - dev2k demo, 7-2
  - extend demo, 7-2
  - oci demo, 7-2
  - proc demo, 7-2
  - usage demo, 7-2
- Time Series Cartridge demos
  - installing, 7-2
- time zone
  - setting with TZ, 2-13
- TMPDIR, 2-12
- TNS listener
  - configuring for Oracle TCP/IP Protocol Adapter, 6-11
- TNS\_ADMIN
  - environment variable, 2-10
- Toolkit II resource file, 2-9
- tools, 3-2
  - sar, 3-2
  - swap, 3-3
- trace and alert files
  - alert files, 3-31
  - trace file names, 3-31
  - using, 3-27, 3-31
- tracing Bourne shell scripts, 2-9
- tuning
  - avoiding index contention, 3-29
  - block and file size, 3-27
  - buffer cache size, 3-27
  - client/server configuration, 3-19
  - CPU usage, 3-19
  - disk I/O, 3-11
  - distributing hot files, 3-15
  - I/O bottlenecks, 3-11
  - isolating contention, 3-21
  - localizing disk I/O, 3-30
  - memory management, 3-4
  - multi-processor systems, 3-26
  - Oracle resources, 3-20
  - parallel server, 3-29
  - post-wait driver, 3-20
  - reducing fragmentation, 3-16
  - resource contention, 3-20
  - table striping, 3-16
  - trace and alert files, 3-27, 3-31

- usage patterns, 3-19
- TWO\_TASK
  - environment variable, 2-10
- two-task
  - architecture
    - security, 2-22
  - signals, 5-17
- TZ
  - and SYSDATE, 2-13
  - environment variable, 2-13

## U

---

- UNIX
  - security, 2-22
- UNIX commands
  - running from SQL\*Plus, 4-7
- UNIX environment variables used with Oracle8
  - DISPLAY, 2-11
  - HOME, 2-11
  - LANG, 2-11
  - LD\_LIBRARY\_PATH, 2-11
  - LDOPTS, 2-11
  - LDPATH, 2-11
  - LPDEST, 2-11
  - PATH, 2-11
  - PRINTER, 2-12
  - SHELL, 2-12
  - TERM, 2-12
  - TMPDIR, 2-12
  - XENVIRONMENT, 2-12
- UNIX kernel
  - tuning, 3-21
- user interrupt handler, 5-18
- user profile
  - SQL\*Plus, 4-2
- user programs
  - oracle call interface, 5-13
  - Pro\*C/C++, 5-6
  - Pro\*COBOL, 5-9
- users
  - sample script, 2-31
- using SQL\*Plus, 4-6
- utlbstat and utlestat SQL scripts, 3-4

## V

---

- V\$SESSION\_WAIT table, 3-21
- V\$SGASTAT table, 3-8
- V\$SYSTEM\_EVENT table, 3-21
- Visual Information Retrieval Cartridge, 7-2
  - creating the demo, 7-2
- vmstat command, 3-4, 3-5

## W

---

- writer activity
  - tuning, 3-12

## X

---

- X Windows, 2-11, 2-12
- XA functionality, 5-20
- XENVIRONMENT
  - environment variable, 2-12