Oracle9*i*

Administrator's Reference

Release 2 (9.2.0.1.0) for UNIX Systems: AIX-Based Systems, Compaq Tru64 UNIX, HP 9000 Series HP-UX, Linux Intel, and Sun Solaris

May 2002

Part No. A97297-01

ORACLE

Oracle9*i* Administrator's Reference, Release 2 (9.2.0.1.0) for UNIX Systems: AIX-Based Systems, Compaq Tru64 UNIX, HP 9000 Series HP-UX, Linux Intel, and Sun Solaris

Part No. A97297-01

Primary Author: Platform Technologies Division Documentation Team

Contents

3 Administering SQL*Plus and *i*SQL*Plus

4 Using Oracle Precompilers and the Oracle Call Interface

5 Configuring Oracle Net Services

A Tuning for Oracle9*i* on AIX

G Optimal Flexible Architecture

Index

List of Tables

Send Us Your Comments

Oracle9*i* Administrator's Reference, Release 2 (9.2.0.1.0) for UNIX Systems: AIX-Based Systems, Compaq Tru64 UNIX, HP 9000 Series HP-UX, Linux Intel, and Sun Solaris

Part No. A97297-01

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 errors or have other suggestions for improvement, you can email them to us at osdwrite_us@oracle.com. Please indicate the title and part number of the document and the chapter, section, and page number if available. Please indicate whether you would like a reply.

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

Preface

This guide and the *Oracle9i Installation Guide Release 2 (9.2.0.1.0) for UNIX Systems* provide instructions for administering and configuring Oracle9*i* release 2 (9.2.0.1.0) on UNIX systems.

Audience

This document is intended for anyone responsible for administering and configuring Oracle9*i* release 2 (9.2.0.1.0) on UNIX systems.

Oracle9*i* Standard Edition and Oracle9*i* Enterprise Edition

Unless noted otherwise, the information in this document is common to both Oracle9*i* Standard Edition and Oracle9*i* Enterprise Edition.

Terminology

The names for the UNIX operating systems have been shortened in this guide and in the *Oracle9i Installation Guide Release 2 (9.2.0.1.0) for UNIX Systems*. The names are as follows:

Operating System	Abbreviated Name
AIX-Based Systems	AIX **Note:** Where there is a difference between AIX 4.3.3 and 5.1, this is noted in the text.
HP 9000 Series HP-UX	HP

Operating System	Abbreviated Name
Linux Intel (32-bit)	Linux
Sun Solaris (32-bit and 64-bit)	Solaris **Note**: Where there is a difference between 32-bit and 64-bit systems, this is noted in the text.
Compaq Tru64 UNIX	Tru64

Typographic Conventions

The following typographic conventions are used in this guide:

Convention	Description
`monospace`	Monospace type indicates UNIX commands, directory names, usernames, pathnames, and filenames.
italics	Italic type indicates a variable, including variable portions of filenames. It is also used for emphasis and for book titles.
UPPERCASE	Uppercase letters indicate Structured Query Language (SQL) reserved words, initialization parameters, and environment variables.
<cr>	This string indicates a newline character.

Command Syntax

UNIX command syntax appears in `monospace` font and assumes the use of the Bourne shell. The "`$`" character at the beginning of UNIX command examples is the default UNIX command prompt. Do not enter it as part of the command.

Convention	Description

backslash \	A backslash indicates a command that is too long to fit on a single line. Enter the line as displayed (with a backslash) or enter it on a single line without a backslash:
	`dd if=/dev/rdsk/c0t1d0s6 of=/dev/rst0 bs=10b \`
	`count=10000`

braces { }	Braces indicate required items: `.DEFINE {macro1}`
brackets [ ]	Brackets indicate optional items: `cvtcrt` *`termname`* `[`*`outfile`*`]`

Convention	Description
ellipses ...	Ellipses indicate an arbitrary number of similar items: `CHKVAL fieldname value1 value2 ... valueN`
italics	Italic type indicates a variable. Substitute a value for the variable: `library_name`
vertical line |	A vertical line indicates a choice within braces or brackets: `SIZE filesize [K|M]`

Accessing Installed Documentation

Oracle9*i* release 2 (9.2.0.1.0) for UNIX systems documentation includes this guide and the *Oracle9i Installation Guide Release 2 (9.2.0.1.0) for UNIX Systems*. You can install documentation in HTML and PDF (Adobe Portable Document Format, which requires Acrobat Reader) formats. UNIX-specific documentation files are located on the Oracle9*i* CD-ROM. Generic documentation files are located on the Online Generic Documentation CD-ROM. The exact location of the documentation files is determined according to the following rules:

- If the ORACLE_DOC environment variable is defined in the environment, then the files are installed in the directory defined by the environment variable.
- If the ORACLE_DOC environment variable is not defined but the ORACLE_BASE environment variable is defined, then the files are installed in the `$ORACLE_BASE/doc` directory.
- If neither the ORACLE_DOC environment variable nor the ORACLE_BASE environment variable are defined in the environment, then the files are installed in the `$ORACLE_HOME/doc` directory.

To access the documentation, navigate to the documentation directory. If you want to access the HTML documentation, then use a browser to open the `index.htm` file. If you prefer paper documentation, then open and print the PDF files.

Oracle Product Documentation

Oracle9*i* product documentation is on the Oracle9*i* Generic Documentation CD-ROM. Instructions for accessing and installing the documents on the CD-ROM are found in the README file in the top-level directory of the CD-ROM.

Documentation Library

The documentation library on the Generic Documentation CD-ROM includes a Web-based search tool that enables you to search through the complete library of Oracle9*i* documents. You may search for information on a particular product, parameter, filename, procedure, error message, or other area of interest. The tool also makes it possible to construct a "virtual book" that consists of topics and procedures relevant for your needs drawn from the complete documentation library. The library also includes a comprehensive Master Index, as well as lists of SQL and PL/SQL keywords, initialization parameters, catalog views, and data dictionary views.

Related Documentation

If you are unfamiliar with the concepts or terminology associated with relational database management systems, then refer to *Oracle9i Database Concepts* before beginning your installation. Use the Installation Checklist in the *Oracle9i Installation Guide Release 2 (9.2.0.1.0) for UNIX Systems* to ensure that you have the required information and that you have completed the necessary pre-installation steps for a successful installation.

Information on system administration and tuning for a production database system is provided in these documents:

- *Oracle9i Installation Guide Release 2 (9.2.0.1.0) for UNIX Systems*
- *Oracle9i Database Administrator's Guide*
- *Oracle9i Net Services Administrator's Guide*
- *Oracle9i Database Performance Guide and Reference*

Information on upgrading from a previous version of the Oracle Server is provided in *Oracle9i Database Migration*.

Oracle Services and Support

The Oracle corporate Web page is located at:

`http://www.oracle.com`

Oracle Corporation offers a wide range of services from this Web site to help facilitate corporate system solutions, including Oracle Education courses, Oracle Consulting services, and Oracle Worldwide Customer Support. In addition, Oracle Corporation provides free trial software, updates on Oracle products and services, and technical brochures and data sheets.

Oracle Support Services

Technical Support registration and worldwide contact information is available at:

`http://www.oracle.com/support`

At the Oracle support site, templates are available to help you prepare information about your problem before you call so that you may be helped more quickly. You must also provide your CSI (Customer Support Identifier) or SAC (Support Access Code) number if applicable or complete contact details, including any special project information.

Oracle*MetaLink*

Oracle*MetaLink* is an Oracle Corporation Web service that provides technical information. Members of Oracle*MetaLink* can search for updates, alerts, patches, and other information about products, releases, and operating systems, or set preferences to be notified automatically of new information. Oracle*MetaLink* offers a variety of services to assist in setting up and administrating Oracle products, including procedures, scripts, commentary, and tuning and configuration best-practices bulletins. In addition, it offers forums for information sharing among Oracle customers, and direct communication with Oracle Support. Oracle*MetaLink* is available to Product Support Customers at no extra cost. Sign up for free membership for this service at:

`http://www.oracle.com/support/metalink`

Use your CSI or SAC number to register.

Products and Documentation

For U.S.A. customers, Oracle Store is at:

`http://oraclestore.oracle.com/`

Links to Stores in other countries are provided at this site.

Product documentation can be found at:

`http://docs.oracle.com`

Customer Service

Global Customer Service contacts are listed at:

`http://www.oracle.com/support/`

Education and Training

Training information and worldwide schedules are available at:

`http://www.oracle.com/education/`

Oracle Technology Network

Register with the Oracle Technology Network (OTN) at:

`http://otn.oracle.com/`

OTN delivers technical papers, code samples, product documentation, an Oracle on-line community, and self-service developer support. It also offers the Oracle key developer products to enable rapid development and deployment of applications built on Oracle technology.

Documentation Accessibility

Our goal is to make Oracle products, services, and supporting documentation accessible, with good usability, to the disabled community. To that end, our documentation includes features that make information available to users of assistive technology. This documentation is available in HTML format, and contains markup to facilitate access by the disabled community. Standards will continue to evolve over time, and Oracle Corporation is actively engaged with other market-leading technology vendors to address technical obstacles so that our documentation can be accessible to all of our customers. For additional information, visit the Oracle Accessibility Program Web site at `http://www.oracle.com/accessibility/`.

Accessibility of Code Examples in Documentation

JAWS, a Windows screen reader, may not always correctly read the code examples in this document. The conventions for writing code require that closing braces should appear on an otherwise empty line; however, JAWS may not always read a line of text that consists solely of a bracket or brace.

Accessibility of Links to External Web Sites in Documentation

This documentation may contain links to Web sites of other companies or organizations that Oracle Corporation does not own or control. Oracle Corporation neither evaluates nor makes any representations regarding the accessibility of these Web sites.

Support for Hearing and Speech Impaired Customers

Oracle Corporation provides dedicated Text Telephone (TTY) access to Oracle Support Services within the United States of America 24 hours a day, seven days a week.

- For technical questions, call 1.800.446.2398
- For non-technical questions, call 1.800.464.2330

1

Administering Oracle9*i*

This chapter provides information on administering Oracle9*i* on AIX, HP, Linux, Solaris, and Tru64. It contains the following sections:

Overview

You must set Oracle9*i* environment variables, parameters, and user settings for Oracle9*i* to work. This chapter describes the various settings for Oracle9*i* on AIX, HP, Linux, Solaris, and Tru64.

In Oracle9*i* files and programs, a question mark (?) represents the value of the ORACLE_HOME environment variable. For example, Oracle9*i* expands the question mark in the following SQL statement to the full pathname of the Oracle home directory:

```
SQL> ALTER TABLESPACE TEMP ADD DATAFILE '?/dbs/temp02.dbf' SIZE 2M
```

Similarly, the @ sign represents the ORACLE_SID environment variable. For example, to indicate a file belonging to the current instance, enter:

```
SQL> ALTER TABLESPACE tablespace_name ADD DATAFILE tempfile@.dbf
```

Environment Variables

This section describes the most commonly-used Oracle9*i* and UNIX environment variables. You must define some environment variables before installing Oracle9*i*. These environment variables are listed in the O*racle9i Installation Guide Release 2 (9.2.0.1.0) for UNIX Systems.*

To display the current value of an environment variable, use the env command. For example, to display the value of the ORACLE_SID environment variable, enter:

```
$ env | grep ORACLE_SID
```

Oracle9*i* Environment Variables

Table 1–1 provides the syntax for, and examples of, environment variables used by Oracle9*i*.

Table 1–1 Oracle9i Environment Variables on UNIX

Variable	Detail	Definition
EPC_DISABLED	Function	Disables Oracle Trace.
	Syntax	TRUE | FALSE
NLS_LANG	Function	Specifies the language, territory, and character set of the client environment. The character set specified by NLS_LANG must match the character set of the terminal or terminal emulator. The character set specified by NLS_LANG can be different from the database character set, in which case Oracle automatically converts the character set. See the *Oracle9i Globalization Support Guide* for a list of values for this variable.
	Syntax	*language_territory.characterse*t
	Example	`french_france.we8dec`
ORA_NLS33	Function	Specifies the directory where language, territory, character set, and linguistic definition files are stored.
	Syntax	*directory_path*
	Example	`$ORACLE_HOME/ocommon/nls/admin/data`
ORA_TZFILE	Function	Specifies the full pathname to the time zone file. You must set this environment variable if you want to use a time zone from the large time zone file (`$ORACLE_HOME/oracore/zoneinfo/timezlrg.dat`) for data in the database. The large time zone file contains information on more time zones than the default time zone file (`$ORACLE_HOME/oracore/zoneinfo/timezone.dat`). All databases that share information must use the same time zone file. You must stop and restart the database when you change the value of this environment variable.
	Syntax	*directory_path*
	Example	`$ORACLE_HOME/oracore/zoneinfo/timezlrg.dat`
ORACLE_BASE	Function	Specifies the base of the Oracle directory structure for Optimal Flexible Architecture (OFA) compliant databases.
	Syntax	*directory_path*
	Example	`/u01/app/oracle`
ORACLE_HOME	Function	Specifies the directory containing the Oracle software.
	Syntax	*directory_path*
	Example	`$ORACLE_BASE/product/`*`9.2.0.1.0`*

Table 1–1 Oracle9i Environment Variables on UNIX (Cont.)

Variable	Detail	Definition
ORACLE_PATH	Function	Specifies the search path for files used by Oracle applications, such as SQL*Plus (`*.sql`), Oracle Forms (`*.frm`), and Oracle Reports (`*.rpt`). If the full path to the file is not specified, or if the file is not in the current directory, the Oracle application uses ORACLE_PATH to locate the file.
	Syntax	Colon-separated list of directories: *`directory1:directory2:directory3`*
	Example	`/u01/app/oracle/product/9.2.0.1.0/bin:.` **Note:** The period adds the current working directory to the search path.
ORACLE_SID	Function	Specifies the Oracle system identifier.
	Syntax	A string of numbers and letters that must begin with a letter. Oracle Corporation recommends a maximum of eight characters for system identifiers. For more information on this environment variable, see the *Oracle9i Installation Guide Release 2 (9.2.0.1.0) for UNIX Systems*.
	Example	SAL1
ORACLE_TRACE	Function	Enables the tracing of shell scripts during an installation. If this environment variable is set to T, many Oracle shell scripts use the `set -x` command, which prints commands and their arguments as they are run.
	Syntax	T or not T.
ORAENV_ASK	Function	Controls whether the `coraenv` or `oraenv` script prompts for the value of the ORACLE_SID or ORACLE_HOME environment variable. If the value is NO, the scripts do not prompt; otherwise they do.
	Syntax	A string. Specifies the directory containing the Oracle Net Services configuration file.
	Example	NO or not NO.
SQLPATH	Function	Specifies the directory or list of directories that SQL*Plus searches for a `login.sql` file.
	Syntax	Colon-separated list of directories: *`directory1:directory2:directory3`*
	Example	`/home:/home/oracle:/u01/oracle`
TNS_ADMIN	Function	Specifies the directory containing the Oracle Net Services configuration file.

Table 1–1 Oracle9i Environment Variables on UNIX (Cont.)

Variable	Detail	Definition
	Syntax	*directory_path*
	Example	$ORACLE_HOME/network/admin
TWO_TASK	Function	Specifies the default connect identifier to use in the connect string. The connect identifier does not need to be specified in the connect string. For example, if the TWO_TASK environment variable is set to sales, you can connect to a database using the CONNECT *username/password* command rather than the CONNECT *username/password*@sales command.
	Syntax	Any connect identifier.
	Range of Values	Any valid connect identifier that can be resolved with a naming method, such as a tnsnames.ora file or a directory server.
	Example	PRODDB_TCP

Note: To prevent conflicts, do not define environment variables with names that are identical to the names of Oracle Server processes, for example: ARCH, PMON, and DBWR.

UNIX Environment Variables

Table 1–2 provides the syntax for, and examples of, UNIX environment variables used with Oracle9*i*.

Table 1–2 UNIX Environment Variables Used with Oracle9i

Variable	Detail	Definition
ADA_PATH (Solaris 32-bit and AIX only)	Function	Specifies the directory containing the Ada compiler.
	Syntax	*directory_path*
	Example	*/usr/lpp/*powerada
CLASSPATH	Function	Used with Java applications. The setting for this variable differs with each Java application. See the product documentation for your Java application for more information.
	Syntax	Colon-separated list of directories or files: *directory1:directory2:file1:file2*

Table 1–2 UNIX Environment Variables Used with Oracle9i (Cont.)

Variable	Detail	Definition
	Example	There is no default setting. CLASSPATH must include the following directories: `$ORACLE_HOME/JRE/lib:$ORACLE_HOME/product/jlib`
DISPLAY	Function	Used by X-based tools. Specifies the display device used for input and output. See the X Windows documentation of the vendor for details.
	Syntax	*`hostname:server`*`[.`*`screen`*`]` where the *`hostname`* is the computer name (either IP address or alias), *`server`* is the sequentila code number for the server, and *`screen`* is the sequentila code number for the screen. If you have a single monitor, the *`server.screen`* is `0.0`. **Note:** If you have a single monitor, *`screen`* is optional.
	Example	`135.287.222.12:0.0` `bambi:0`
HOME	Function	The user's home directory.
	Syntax	*`directory_path`*
	Example	`/home/oracle`
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 the *Oracle9i Installation Guide Release 2 (9.2.0.1.0) for UNIX Systems* for more information.
LD_OPTIONS	Function	Specifies the default linker options. See the `ld` man pages for more information.
LPDEST (Solaris only)	Function	Specifies the name of the default printer.
	Syntax	*`string`*
	Example	`docprinter`
LDPATH (Solaris only)	Function	Default directories used by the linker to find shared object libraries. See the `ld` man pages for more information on this environment variable.
LD_LIBRARY_PATH	Function	List of directories that the shared library loader searches to locate shared object libraries at runtime. See the `ld` man page for information on this environment variable.
	Syntax	Colon-separated list of directories: *`directory1:directory2:directory3`*
	Example	`/usr/dt/lib:$ORACLE_HOME/lib`

Table 1–2 UNIX Environment Variables Used with Oracle9i (Cont.)

Variable	Detail	Definition
LIBPATH (AIX only)	Function	List of directories that the shared library loader searches to locate shared object libraries at runtime. See the `ld` man page for information on this environment variable.
	Syntax	Colon-separated list of directories: *`directory1:directory2:directory3`*
	Example	`/usr/dt/lib:$ORACLE_HOME/lib`
PATH	Function	Used by the shell to locate executable programs; must include the `$ORACLE_HOME/bin` directory.
	Syntax	Colon-separated list of directories: *`directory1:directory2:directory3`*
	Example	`/bin:/usr/bin:/usr/local/bin: /usr/bin/X11:$ORACLE_HOME/bin:$HOME/bin:.`
		Note: The period adds the current working directory to the search path.
PRINTER	Function	Defines the name of the default printer.
	Syntax	*`string`*
	Example	`docprinter`
SHELL	Function	Specifies the command interpreter used during a host command.
	Syntax	*`shell_path`*
	Range of Values	`/bin/sh`, `/bin/csh`, `/bin/ksh`, or any other UNIX command interpreter.
	Example	`/bin/sh`
SHLIB_PATH (HP 32-bit libraries only)	Function	List of directories that the shared library loader searches to locate shared object libraries at runtime. See the `ld` man page for information on this environment variable.
	Syntax	Colon-separated list of directories: *`directory1:directory2:directory3`*
	Example	`/usr/dt/lib:$ORACLE_HOME/lib`
TEMP and TMPDIR	Function	Specifies the default directory for temporary files; if set, tools that create temporary files create them in this directory.
	Syntax	*`directory_path`*
	Example	`/u02/oracle/tmp`
XENVIRONMENT	Function	Specifies a file containing X-Windows system resource definitions. See your X-Windows documentation for more information.

Setting a Common Environment

This section describes how to use the `oraenv` command to set a common UNIX environment. You can use the `coraenv` command for the C shell in the same way.

oraenv Script File

The `oraenv` script is created during installation. This script sets environment variables based on the contents of the `oratab` file and provides:

- A central means of updating all user accounts with database changes
- A mechanism for switching between Oracle9*i* databases specified in the `oratab` file

You may find yourself frequently adding and removing databases from your development system or your users may be switching between several different Oracle databases installed on the same system. You can use the `oraenv` script to ensure that user accounts are updated and to switch between databases.

The `oraenv` script is usually called from the user's `.profile` file . It is used to set the ORACLE_SID, ORACLE_HOME, and PATH environment variables to include the `$ORACLE_HOME/bin` directory. When switching between databases, users can run the `oraenv` script to set these environment variables.

> **See Also:** *Oracle9i Installation Guide Release 2 (9.2.0.1.0) for UNIX Systems* for more information on setting a common environment.

Local bin Directory

The directory that contains the `oraenv` and `dbhome` scripts is called the local `bin` directory. All database users should have read access to this directory. Include the path of the local `bin` directory in the users' PATH environment variables. If you run the `root.sh` script after installation, the script prompts you for the path of the local `bin` directory and automatically copies the `oraenv` and `dbhome` scripts to the directory. The default local `bin` directory is `/usr/local/bin`. If you do not run the `root.sh` script, you can manually copy the `oraenv` and `dbhome` script from the `$ORACLE_HOME/bin` directory to your local `bin` directory.

Switching Between Databases

To switch from one database or database instance to another, call the `oraenv` script. Reply to the prompt with the value of the ORACLE_SID environment variable of the database to which you are switching. If the local `bin` directory is not included in the PATH environment, provide the full path of the `oraenv` command file. For example:

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

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

Use the `env` command to show the environment variable values that have been exported to the environment. The Bourne shell and Korn shell can set values without exporting them.

For the Bourne or Korn shell, enter:

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

For the C shell, enter:

```
% setenv ORACLE_SID test
```

In the preceding examples, *`test`* is the value of the ORACLE_SID environment variable.

Setting the System Time

The TZ environment variable sets the time zone. It enables you 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.

Oracle Corporation recommends that you do not change your personal TZ value. Using different values of TZ such as GMT+24 might change the date a transaction is recorded. This changed date affects Oracle applications that use SYSDATE, such as Oracle Financials. To avoid this problem, use sequence numbers to order a table instead of date columns.

LD_PRELOAD Environment Variable for Loading Shared Libraries (HP Only)

If you are developing Oracle9*i* applications on HP that use shared libraries and function calls, users of your application must set the LD_PRELOAD environment variable.

Because they require less memory, many programs use shared libraries. In most cases, the `dld.sl` 64-bit HP-UX dynamic loader is invoked automatically when applications using shared libraries start. At run time, the dynamic loader implicitly attaches all shared libraries linked with the program to the process. This includes the HP-UX thread-local storage (TLS) libraries. Programs can also use the `shl_load()` HP-UX function call to:

- Explicitly access the 64-bit HP-UX dynamic loader
- Attach a shared library to a process at run time
- Calculate the addresses of symbols defined within shared libraries
- Detach the library when finished

In laboratory tests, errors occur when an application uses the `shl_load()` function call to attach a shared library that directly or indirectly contains HP-UX TLS libraries. These errors include Oracle shared libraries which are currently linked with the `libpthread.sl` and `libcl.sl` HP-UX TLS libraries, for example `libclntsh.sl`.

In the following example, the `prog.c` program calls the `shl_load()` function to load the `libclntsh.sl` library:

```
shl_load("Oracle_home_directory/rdbms/lib/libclntsh.sl", BIND_IMMMEDIATE |
BIND_VERBOSE | DYNAMIC_PATH | 0L);
```

When the program is executed, it generates the following errors:

```
/usr/lib/pa20_64/dld.sl: Cannot dlopen load module
'/usr/lib/pa20_64/libpthread.1' because it contains thread specific data.

/usr/lib/pa20_64/dld.sl: Cannot dlopen load module '/usr/lib/pa20_64/libcl.2'
because it contains thread specific data.
```

The new LD_PRELOAD environment variable resolves these errors.

To prevent errors occurring when an application uses the `shl_load()` function call to attach a shared library that directly or indirectly contains HP-UX TLS libraries, perform the following steps:

1. Set the value of the LD_PRELOAD environment variable to include a colon-separated or whitespace-separated list of the TLS libraries that your program uses, for example:

   ```
   $ export LD_PRELOAD=/usr/lib/pa20_64/libpthread.1:/usr/lib/pa20_64/libcl.2
   ```

   The dynamic loader treats the libraries specified by the LD_PRELOAD variable as the first libraries in the link line and pre-loads these libraries implicitly at application startup.

2. Enter the following command to execute your program, where *prog* is the name of your program:

   ```
   $ prog
   ```

   The errors shown on the previous page should not appear.

3. Unset the LD_PRELOAD variable to prevent memory overheads:

   ```
   $ unset LD_PRELOAD
   ```

> **See Also:** Your HP-UX system documentation for more information on the LD_PRELOAD environment variable.

Relinking Executables

You can manually relink your product executables using a relink shell script located in the `$ORACLE_HOME/bin` directory. Relinking is necessary after applying any operating system patches or after an operating system upgrade.

> **Note:** Shut down all executables that are running in the ORACLE_HOME that you are relinking, including the listener and Oracle Intelligent Agent. In addition, shut down any applications that are linked with any Oracle shared libraries.

The relink script manually relinks Oracle product executables, depending on the products that have been installed in the Oracle home directory.

To relink product executables, enter the following command, where *argument* is one of the values listed in Table 1–3:

```
$ relink argument
```

Table 1–3 Relink Script Parameters

Value	Description
`all`	Every product executable that has been installed
`oracle`	Oracle server executable only
`network`	`net_client`, `net_server`, `cman`, `names`
`client`	`net_client`, `otrace`, `plsql`
`ctx`	`Oracle Text utilities`
`precomp`	All precompilers that have been installed
`utilities`	All utilities that have been installed
`oemagent`	`oemagent` **Note:** To give the correct permissions to the `dbsnmp` executable, you must run the `root.sh` script after relinking `oemagent` executables.
ldap	`ldap`, `oid`

System Global Area

The System Global Area (SGA) is the Oracle structure that is located 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.

The maximum size of a single shared memory segment is specified by the SHMMAX kernel parameter (SHM_MAX on Tru64). The following table shows the recommended value for this parameter, depending on your platform:

Platform	Recommended Value
AIX	Not applicable.
HP	The size of the physical memory installed on the system. **See Also:** "HP-UX Shared Memory Segments for a 64-Bit Oracle Instance" on page B-2 for information on the SHMMAX parameter on HP.
Solaris	4 GB minus 16 MB. Can be greater than 4 GB on 64-bit systems.

Platform	Recommended Value
Tru64	4 GB minus 16 MB. **Note:** The value of the SHM_MAX parameter must be at least 16 MB for the Oracle instance to start. If your system runs both Oracle8*i* and Oracle9*i* instances, you must set the value of this parameter to 2 GB minus 16 MB.
Linux	Half the size of the physical memory installed on the system.

If the size of the SGA exceeds the maximum size of a shared memory segment (SHMMAX or SHM_MAX), Oracle9*i* attempts to attach more contiguous segments to fulfill the requested SGA size. The SHMSEG kernel parameter (SHM_SEG on Tru64) specifies the maximum number of segments that can be attached by any process.

Set the following initialization file parameters to control the size of the SGA:

- DB_CACHE_SIZE
- DB_BLOCK_SIZE
- JAVA_POOL_SIZE
- LARGE_POOL_SIZE
- LOG_BUFFERS
- SHARED_POOL_SIZE

Use caution when setting values for these parameters. When values are set too high, too much of the computer's physical memory is devoted to shared memory, resulting in poor performance.

Determining the Size of the SGA

You can determine the SGA size in one of the following ways:

- Enter the following SQL*Plus command to display the size of the SGA for a running database:

  ```
  SQL> SHOW SGA
  ```

  The result is shown in bytes.

- Determine the size of the SGA when you start your database instance. The SGA size is displayed next to the heading Total System Global Area.

Intimate Shared Memory (Solaris Only)

On Solaris systems, Oracle9*i* uses Intimate Shared Memory (ISM) for shared memory segments because it shares virtual memory resources among Oracle processes. On Solaris 2.6 and Solaris 7, Oracle9*i* uses ISM by default. ISM causes the physical memory for the entire shared memory segment to be locked automatically.

On Solaris 8, dynamic / pageable ISM (DISM) is available. This enables Oracle9*i* to share virtual memory resources among processes sharing the segment, and at the same time, enables memory paging. The operating system does not have to lock down physical memory for the entire shared memory segment.

Oracle9*i* automatically decides at startup whether to use ISM or DISM, based on the following criteria:

- Oracle9*i* uses DISM if it is available on the system, and if the value of the SGA_MAX_SIZE initialization parameter is larger than the size required for all SGA components combined. This allows Oracle9*i* to lock only the amount of physical memory that is used.
- Oracle9*i* uses ISM if the entire shared memory segment is in use at startup or if the value of the SGA_MAX_SIZE parameter is equal to or smaller than the size required for all SGA components combined.

Regardless of whether Oracle9*i* uses ISM or DISM, it can always exchange the memory between dynamically sizable components such as the buffer cache, the shared pool, and the large pool after it starts an instance. Oracle9*i* can relinquish memory from one dynamic SGA component and allocate it to another component.

Because shared memory segments are not implicitly locked in memory, when using DISM, Oracle9*i* explicitly locks shared memory that is currently in use at startup. When a dynamic SGA operation uses more shared memory, Oracle9*i* explicitly performs a lock operation on the memory that comes in use. When a dynamic SGA operation releases shared memory, Oracle9*i* explicitly performs an unlock operation on the memory that is freed, so that it becomes available to other applications.

Oracle9*i* uses a new utility, `oradism`, to lock and unlock shared memory. If the LOCK_SGA parameter is set to TRUE, Oracle9*i* attempts to lock the entire SGA at startup. In this case, no locks or unlocks are performed when the SGA resizes.

With Oracle9*i* release 2 (9.2.0.1.0), the `oradism` utility is automatically set up during installation. You do not need to perform any configuration tasks to use dynamic SGA.

Note: The process name for the `oradism` utility is `ora_dism_`*`sid`*, where *`sid`* is the system identifier. When using DISM, this process is started during instance startup, and automatically quits when the instance is shut down.

If a message appears in the alert log saying that the `oradism` utility is not set up correctly, verify that the `oradism` utility is located in the `$ORACLE_HOME/bin/` directory and that it has superuser privileges.

Shared Memory on AIX

On AIX, shared memory uses common virtual memory resources across processes. Processes share virtual memory segments through a common set of virtual memory translation resources, for example tables and cached entries, for improved performance.

With Oracle9*i* on AIX, shared memory can be pinned to prevent paging and to reduce I/O overhead. To do this, set the LOCK_SGA parameter to TRUE. Additionally, starting with Oracle9*i* release 2 (9.2.0.1.0) on AIX 5.1, the same parameter activates the large page feature whenever the underlying hardware supports it.

Enter the following command to make pinned memory available to Oracle9*i* on AIX systems:

```
$ vmtune -s 1
```

Enter the following command to set the maximum percentage of real memory available for pinned memory where *`percent_of_real_memory`* is the maximum percent of real memory that you want to set:

```
$ vmtune -M percent_of_real_memory
```

When using the `-M` flag, it is important that the amount of pinned memory exceeds the Oracle SGA size by at least 3 percent of the real memory on the system, allowing free pinnable memory for use by the kernel.

Use the AIX `svmon` command to monitor the use of pinned memory during the operation of the system. Oracle9*i* attempts to pin memory only if the LOCK_SGA parameter is set to TRUE.

Large Page Feature on AIX Power4-Based Systems

On AIX Power4-based systems, you can enable large pages by:

- Using the `-g` and `-L` flags of the `vmtune` command. The `-g` flag specifies the size of the large pages, and should be set to 256 MB (`-g 268435456`) for Power4 processors. The `-L` flag specifies the number of large pages as specified by the `-g` flag to reserve for Oracle9*i*.
- Running the `bosboot` command and then rebooting the system.

Oracle Corporation recommends specifying enough large pages to contain the entire SGA. Starting with Oracle9*i* release 2 (9.2.0.1.0), the Oracle instance attempts to allocate large pages when the LOCK_SGA parameter is set to TRUE. If the SGA size exceeds the size of memory available for pinning, or large pages, the portion of the SGA exceeding these sizes is allocated to ordinary shared memory.

> **See Also:** The AIX documentation for information on the command for enabling and tuning pinned memory and large pages.

Oracle9*i* Memory Requirements

The total memory used by an Oracle9*i* instance is approximately the size of the SGA plus the size of the Program Global Area (PGA). To specify the size of the PGA, set the PGA_AGGREGATE_TARGET parameter.

Database Limits

Table 1–4 lists the default and maximum values for parameters in a CREATE DATABASE or CREATE CONTROLFILE statement.

Note: Interdependencies among these parameters may affect allowable values.

Table 1–4 CREATE CONTROLFILE and CREATE DATABASE Parameters

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

Table 1–5 lists the Oracle9*i* file size limits in bytes specific to UNIX.

Table 1–5 File Size Limits

File Type	Operating System	Maximum SIze
Datafiles	Any	4,194,303 multiplied by the value of the DB_BLOCK_SIZE parameter
Import / Export files and SQL*Loader files	Tru64	< 16TB
	AIX, HP, Linux, Solaris: 32-bit with 32-bit files	2,147,483,647 bytes
	AIX, HP, Linux, Solaris: 32-bit with 64-bit files	Unlimited
	AIX, HP, Linux, Solaris: 64-bit	Unlimited
Control files	Solaris, HP, Linux	20000 database blocks
	AIX	10000 database blocks
	Tru64	19200 database blocks

Operating System Accounts and Groups

Special operating system accounts and groups are required by Oracle9*i*, as follows:

- Oracle software owner account
- OSDBA, OSOPER, and ORAINVENTORY groups

Oracle Software Owner Account

The Oracle software owner account, usually named `oracle`, is the account that you use to install the Oracle software. You can use different Oracle software owner accounts for separate installations of the software. However, you must use the same account that installed the software for all subsequent maintenance tasks on that installation.

Oracle Corporation recommends that the Oracle software owner has the ORAINVENTORY group as its primary group and the OSDBA group as its secondary group.

OSDBA, OSOPER, and ORAINVENTORY Groups

Table 1–6 describes the special UNIX groups required by Oracle9*i*.

Table 1–6 UNIX Groups

Group	Typical Name	Description
OSDBA	`dba`	Operating system accounts that are members of the OSDBA group have special database privileges. Members of this group can connect to the database using the SYSDBA privilege. The Oracle software owner is the only required member of this group. You can add other accounts as required. For more information on the OSDBA group and the SYSDBA privilege, see the *Oracle9i Database Administrator's Guide* and the *Oracle9i Installation Guide Release 2 (9.2.0.1.0) for UNIX Systems*.
OSOPER	`oper`	The OSOPER group is an optional group. Operating system accounts that are members of the OSOPER group have special database privileges. Members of this group can connect to the database using the SYSOPER privilege. For more information on the OSOPER group and the SYSOPER privilege, see the *Oracle9i Database Administrator's Guide* and the *Oracle9i Installation Guide Release 2 (9.2.0.1.0) for UNIX Systems*.

Table 1–6 UNIX Groups (Cont.)

Group	Typical Name	Description
ORAINVENTORY	`oinstall`	All users installing Oracle software on a UNIX system must belong to the same UNIX group, called the ORAINVENTORY group. This group must be the primary group of the Oracle software owner during installations. After the installation, this group owns all of the Oracle files installed on the system.

Oracle9*i* 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 Oracle9*i* 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 in the `oracle` program.

> **See Also:** *Oracle9i Database Administrator's Guide* for more information on security issues.

Groups and Security

Oracle programs are divided into two sets for security purposes: those executable by all (`other`, in UNIX terms), and those executable by DBAs only. Oracle Corporation recommends the following approach to security:

- The primary group for the `oracle` account should be the `oinstall` group.
- The `oracle` account must have the `dba` group as a secondary group.
- Although any user account which requires `dba` privileges can belong to the `dba` group, the only user account which should belong to the `oinstall` group is the `oracle` account.

Security for Database Files

See the O*racle9i Installation Guide Release 2 (9.2.0.1.0) for UNIX Systems* for information on the appropriate permissions for database files.

External Authentication

If you choose to use external authentication, you must use the value of the OS_AUTHENT_PREFIX initialization parameter as a prefix for Oracle usernames. If you do not explicitly set this parameter, the default value on UNIX is ops$, which is case sensitive.

To use the same usernames for both operating system and Oracle authentication, set this initialization parameter to a null string, as follows:

```
OS_AUTHENT_PREFIX=""
```

> **See Also:** *Oracle9i Database Administrator's Guide* for more information on external authentication.

Running the orapwd Utility

You can use a password file to identify users that can use the SYSDBA and SYSOPER privileges when connecting to the database. To create the password file:

1. Log in as the Oracle software owner.
2. Use the $ORACLE_HOME/bin/orapwd utility, which has the following syntax:

   ```
   $ orapwd file=filename password=password entries=max_users
   ```

   The following table describes the values that you must specify in this command:

Variable	Description
filename	Name of the file where password information is written. The name of the file must be orapw*sid* and you must supply the full pathname. Its contents are encrypted and not user-readable. This parameter is mandatory. The password file is typically created in the $ORACLE_HOME/dbs directory.
password	This parameter sets the password for the SYS user. If you use an ALTER USER statement to change the password for the SYS user after you connect to the database, both the password stored in the data dictionary and the password stored in the password file are updated. This parameter is mandatory.
max_users	Maximum number of entries that you require the password file to accept.

See Also: *Oracle9i Database Administrator's Guide* for more information on using the `orapwd` utility.

Password Management

When the Database Configuration Assistant Summary window appears, users must change the SYS and SYSTEM account passwords. You cannot use the default CHANGE_ON_INSTALL and MANAGER passwords.

For security reasons, the Database Configuration Assistant locks most Oracle user accounts after it creates the database. It does not lock the SYS, SYSTEM, or SCOTT accounts. You must unlock any locked accounts and change their passwords before logging into them.

To change the passwords, click the Password Management button in the Database Configuration Assistant Summary window.

Alternatively, use SQL*Plus to connect to the database as SYSDBA and enter the following command:

```
SQL> ALTER USER username IDENTIFIED BY passwd ACCOUNT UNLOCK;
```

Customizing the Initialization File

The default initialization file (`init`*`sid`*`.ora`) is provided with the Oracle9*i* software. The Database Creation Assistant creates it in the `$ORACLE_BASE/admin/`*`sid`*`/pfile` directory. A sample initialization file is located in the `$ORACLE_HOME/dbs` directory.

Table 1–7 lists default initialization parameter values on UNIX. All Oracle9*i* instances assume these values if you do not specify different values for them in the `init`*`sid`*`.ora` file. Oracle Corporation recommends that you include in the `init`*`sid`*`.ora` file only those parameters that differ from the default initialization parameter values.

Use the SHOW PARAMETERS command in SQL*Plus to display the current values of these parameters on the system.

See Also: For Tru64, "CLUSTER_INTERCONNECTS Initialization Parameter (Formerly TRU64_IPC_NET)" on page D-8, and the *Oracle9i Database Reference*, *Oracle9i Database Administrator's Guide*, and the *Oracle9i Performance Guide and Reference* for more information on initialization parameters.

Table 1–7 Initialization Parameters

Parameter	Default Value	Range of Values
BACKGROUND_DUMP_DEST	`?/rdbms/log`	Valid directory name
BITMAP_MERGE_AREA_SIZE	1048576	65536 to unlimited
COMMIT_POINT_STRENGTH	1	0 to 255
CONTROL_FILES	`?/dbs/cntrl`*`sid`*`.dbf`	Valid filenames
CREATE_BITMAP_AREA_SIZE	8388608	65536 to unlimited
DB_BLOCK_SIZE	2048	2048 to 16384 (Linux, Solaris 32-bit) 2048 to 32768 (AIX, HP, Tru64, Solaris 64-bit)
DB_CACHE_SIZE	8 MB	8 MB to unlimited
DB_FILES	200	1 to 2000000
DB_FILE_DIRECT_IO_COUNT	64	0 to 1048576 / block size
DB_FILE_MULTIBLOCK_READ_COUNT	8	1 to the smaller of the following values: ■ The value of DB_CACHE_SIZE divided by 4 ■ 1048576 divided by the value of DB_BLOCK_SIZE
HASH_AREA_SIZE	The value of SORT_AREA_SIZE multiplied by 2	0 to unlimited
HASH_MULTIBLOCK_IO_COUNT	0 (self-tuned)	0 to the smallest of the following values: ■ 127 ■ The value of DB_CACHE_SIZE divided by 4 ■ 1048576 divided by the value of DB_BLOCK_SIZE
JAVA_POOL_SIZE	24 MB	1000000 to 1000000000
LOCK_SGA	FALSE	TRUE, FALSE
LOG_ARCHIVE_DEST	NULL	Valid directory names

Table 1–7 Initialization Parameters (Cont.)

Parameter	Default Value	Range of Values
LOG_ARCHIVE_FORMAT	"%t_%s.dbf"	Valid filenames
LOG_BUFFER	512 KB or (128 KB multiplied by the value of CPU_COUNT, which ever is higher)	66560 to unlimited
LOG_CHECKPOINT_INTERVAL	0	0 to unlimited
MAX_DISPATCHERS	5	1 to maximum number of processes that can be opened by your operating system.
MAX_SHARED_SERVERS	2 multiplied by the value of SHARED_SERVER, if the value of SHARED_SERVERS is greater than 20, otherwise 20	Between the value of SHARED_SERVERS and the value of PROCESSES
SHARED_SERVERS	1, if DISPATCHERS is specified, else 0	Between 1 and PROCESSES
NLS_LANGUAGE	AMERICAN	Valid language names
NLS_TERRITORY	AMERICA	Valid territory names
OBJECT_CACHE_MAX_SIZE_PERCENT	10	0 to unlimited
OBJECT_CACHE_OPTIMAL_SIZE	100 KB	10 KB to unlimited
OPEN_CURSORS	50	1 to unlimited
OS_AUTHENT_PREFIX	ops$	Arbitrary string
PGA_AGGREGATE_TARGET	0 (auto memory management is not set)	10 MB to 4 TB (if set)
PROCESSES	30, if not PARALLEL_AUTOMATIC_TUNING	6 to unlimited
SHARED_POOL_SIZE	64 MB on 64-bit systems, 8 MB on 32-bit systems	4194304 to unlimited
SORT_AREA_SIZE	65536	0 to unlimited

Oracle HTTP Server

To administer the Oracle HTTP Server, you must have access to the local system on which the server is running, and in some cases you must have `root` access.

The Oracle HTTP Server starts automatically on the default port 7777 after installation. To verify that the server is running, enter the following command:

```
$ ps -elf | grep httpd
```

Note: This product includes software developed by the Apache Software Foundation.

Starting and Stopping the Oracle HTTP Server

If you modify the configuration, you must restart the server. You must be logged in as the `root` user to start the server with SSL enabled.

To stop the server, enter the following commands:

```
$ cd $ORACLE_HOME/Apache/Apache/bin
$ su root
# ./apachectl stop
```

To restart the server, enter the following commands:

```
$ cd $ORACLE_HOME/Apache/Apache/bin
$ su root
# ./apachectl {start|startssl}
```

Use the `start` flag to start a non-SSL enabled server or use the `startssl` flag to start an SSL enabled server. The default port for a non-SSL enabled server is 80. The default port for an SSL enabled server is 443.

Note: If you start a non-SSL enabled server and port 80 is not available, the Oracle HTTP server uses the next available port between 7777 and 7877 as its default port.

If you start an SSL enabled server and port 443 is not available, the Oracle HTTP server uses the next available port between 4443 and 4543 as its default port.

Accessing the Default Initial Static Page

The default initial static page contains links to online documentation as well as demonstrations for each of the components. To access the initial static page, use an internet browser to view one of the following URLs:

- For servers without SSL enabled:

  `http://`*`ServerName`*`:7777/`

- For servers with SSL enabled:

  `http://`*`ServerName`*`:80/`

- For servers with SSL enabled (secure, using HTTPS):

  `https://`*`ServerName`*`:4443/`

In the preceding example, *`ServerName`* is configured in the HTTP Server `httpd.conf` configuration file. To locate the appropriate value in the configuration file, enter:

```
$ grep ServerName $ORACLE_HOME/Apache/Apache/conf/httpd.conf
```

Oracle HTTP Server Log Files

A number of log files are generated by the server. Check them periodically to make sure that the server is working correctly. By default, the error log level is set to `warn` in the configuration files. You can change the default error level by editing the appropriate configuration file and restarting the server.

The following log files are generated by the server:

```
$ORACLE_HOME/Apache/Apache/logs/access_log
$ORACLE_HOME/Apache/Apache/logs/error_log
$ORACLE_HOME/Apache/Apache/logs/ssl_engine_log
$ORACLE_HOME/Apache/Jserv/logs/jserv.log
$ORACLE_HOME/Apache/Jserv/logs/mod_jserv.log
```

Demonstration Files

This section describes how to build and run the SQL*Loader and PL/SQL demonstration programs installed with Oracle9*i*.

SQL*Loader Demonstrations

The following SQL*Loader demonstration files are included with Oracle9*i* in the `$ORACLE_HOME/rdbms/demo` directory. Run the demonstrations in numerical order:

`ulcase1`	`ulcase3`	`ulcase5`	`ulcase7`
`ulcase2`	`ulcase4`	`ulcase6`	

To Create and Run a Demonstration

Run demonstrations while logged in as the user SCOTT/TIGER. If the SCOTT/TIGER schema does not exist on your system, run the `$ORACLE_HOME/rdbms/admin/utlsampl.sql` script to create it.

Ensure that:

- The user SCOTT/TIGER has CONNECT and RESOURCE privileges
- The EMP and DEPT tables exist

In the following steps, *n* represents the demonstration number, listed in the previous section. To create and run a demonstration:

1. Run the `ulcase`*n*`.sql` script corresponding to the demonstration you want to run:

   ```
   $ sqlplus SCOTT/TIGER @ulcasen.sql
   ```

2. Load the demonstration data into the objects:

   ```
   $ sqlplus SCOTT/TIGER @ulcasen.sql
   ```

The following list provides additional information on the `ulcase2`, `ulcase6`, and `ulcase7` demonstrations:

- For the `ulcase2` demonstration, you do not have to run the `ulcase2.sql` script.
- For the `ulcase6` demonstration, run the `ulcase6.sql` script, then enter the following command:

  ```
  $ sqlldr SCOTT/TIGER ulcase6 DIRECT=true
  ```

- For the `ulcase7` demonstration, run the `ulcase7s.sql` script, then enter the following command:

  ```
  $ sqlldr SCOTT/TIGER ulcase7
  ```

After running the demonstration, run the `ulcase7e.sql` script to drop the trigger and package used by this demonstration.

PL/SQL Demonstrations

PL/SQL includes a number of demonstration programs that you can load. The Oracle9*i* database must be open and mounted to work with the demonstration programs.

You must build database objects and load sample data before using these programs. To build the objects and load the sample data:

1. Change directory to the PL/SQL demonstrations directory:

   ```
   $ cd $ORACLE_HOME/plsql/demo
   ```

2. Start SQL*Plus and connect as SCOTT/TIGER:

   ```
   $ sqlplus SCOTT/TIGER
   ```

3. Enter the following commands to build the objects and load the sample data:

   ```
   SQL> @exampbld.sql
   SQL> @examplod.sql
   ```

Note: Build the demonstrations as any Oracle user with sufficient privileges. Run the demonstrations as the same Oracle user.

PL/SQL Kernel Demonstrations

The following PL/SQL kernel demonstrations are available:

```
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
```

To compile and run the `examp*n*.sql` or `sample*n*.sql` PL/SQL kernel demonstrations:

1. Start SQL*Plus and connect as SCOTT/TIGER:

```
$ cd $ORACLE_HOME/plsql/demo
$ sqlplus SCOTT/TIGER
```

2. Enter a command similar to the following to run a demonstration, where *demoname*`.sql` is the name of the demonstration:

```
SQL> @demoname
```

To run the `extproc.sql` demonstration:

1. If necessary, add an entry for external procedures to the `tnsnames.ora` file, similar to the following:

```
EXTPROC_CONNECTION_DATA.domain =
   (DESCRIPTION =
       (ADDRESS_LIST =
          (ADDRESS=(PROTOCOL = IPC)( KEY = EXTPROC))
       )
       (CONNECT_DATA =
          (SID = PLSExtProc)
       )
    )
```

2. If necessary, add an entry for external procedures to the `listener.ora` file, similar to the following:

```
SID_LIST_LISTENER =
  (SID_LIST =
     (SID_DESC=
        (SID_NAME=PLSExtProc)
        (ORACLE_HOME=/u01/app/oracle/product/9.2.0.1.0)

(ENVS=EXTPROC_DLLS=/u01/app/oracle/product/9.2.0.1.0/plsql/demo/extproc.so,
LD_LIBRARY_PATH=/u01/app/oracle/product/9.2.0.1.0/plsql/demo)
        (PROGRAM=extproc)
      )
    )
```

> **Note:** The value that you specify for SID_NAME in the `listener.ora` file must match the value that you specify for SID in the `tnsnames.ora` file.

3. Enter the following command to create the `extproc.so` shared object, build the required database objects, and load the sample data:

```
$ make -f demo_plsql.mk extproc.so exampbld examplod
```

   Alternatively, if you have already built the database objects and loaded the sample data, enter the following command:

```
$ make -f demo_plsql.mk extproc.so
```

4. From SQL*Plus, enter the following commands:

```
SQL> CONNECT SYSTEM/MANAGER
SQL> GRANT CREATE LIBRARY TO SCOTT;
SQL> CONNECT SCOTT/TIGER
SQL> CREATE OR REPLACE LIBRARY demolib IS
  2  '$ORACLE_HOME/plsql/demo/extproc.so';
  3  /
```

5. To run the demonstration, enter the following command:

```
SQL> @extproc
```

PL/SQL Precompiler Demonstrations

> **Note:** The `make` commands shown in this section build the required database objects and load the sample data in the SCOTT schema.

The following precompiler demonstrations are available:

`examp9.pc` `examp10.pc` `sample5.pc` `sample6.pc`

To build all of the PL/SQL precompiler demonstrations, enter the following commands:

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

To build a single demonstration, enter its name as the argument in the `make` command. For example, to build the `examp9` demonstration, enter:

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

To run the `examp9` demonstration, enter the following command:

```
$ ./examp9
```

Administering SQL*Loader

SQL*Loader is used by both database administrators and Oracle9*i* users. It loads data from standard operating system files into Oracle database tables.

> **See Also:** *Oracle 9i Database Utilities* for more information on using SQL*Loader.

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

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

The following table describes these processing options:

String	Description
`"str"`	Specifies a stream of records, each terminated by a newline character, which are read in one record at a time. This option is the default.
`"fix n"`	Indicates that the file consists of fixed-length records, each of which is *n* bytes long, where *n* is an integer value.
`"var n"`	Indicates that the file consists of variable-length records, with the length of each record specified in the first *n* characters. If you do not specify a value of *n*, SQL*Loader assumes a value of 5.

If you do not select the file processing option, the information is processed by default as a stream of records (`"str"`). You might find that the `"fix"` option yields faster performance than the default `"str"` option because it does not scan for record terminators.

Newline Characters 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 character, include the length of the newline character (one character) when specifying the record length to SQL*Loader.

For example, to read the following file, specify `"fix 4"` instead of `"fix 3"` to include the additional newline character:

```
AAA<cr>
BBB<cr>
CCC<cr>
```

If you do not terminate the last record in a file of fixed-length 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 character, terminate all records with a newline character.

> **Caution:** 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 Newline Characters

Use the `position(`*x*`:`*y*`)` function in the control file to discard the newline characters from fixed length records rather than loading them.

For example, enter the following lines in your control file to discard newline characters from the fourth position:

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

Using these lines, SQL*Loader discards newline characters because they are in the fourth position in each fixed-length record.

2

Tuning for Oracle9*i* on UNIX

The more your Oracle9*i* applications increase in complexity, the more you must tune the system to optimize performance and prevent data bottlenecks. This chapter describes how to configure your Oracle9*i* installation to optimize its performance. It contains the following sections:

See Also: The following documents and appendices for more information on system tuning:

- *Oracle9i Database Performance Methods*
- *Oracle9i Database Performance Guide and Reference*
- *Oracle9i Real Application Clusters Concepts*
- *Oracle9i Real Application Clusters Administration*
- Appendix A, "Tuning for Oracle9i on AIX"
- Appendix B, "Tuning for Oracle9i on HP"
- Appendix C, "Tuning for Oracle9i on Linux"
- Appendix D, "Tuning for Oracle9i on Tru64"

Importance of Tuning

Oracle9*i* 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-node computer systems, most of the performance tuning tips provided here are also valid when using Oracle9*i* Real Application Clusters and features available with Oracle9*i*.

Before tuning the system, observe its normal behavior using the tools described in "Operating System Tools" on page 2-3.

Types of Performance Bottlenecks

Performance bottlenecks are often caused by the following:

- Memory contention

  Memory contention occurs when processes require more memory than is available. When this occurs, the system pages and swaps processes between memory and disk.

- Disk I/O contention

  Disk I/O contention is caused by poor memory management, poor distribution of tablespaces and files across disks, or a combination of both.

- CPU contention

  Although the UNIX kernel usually allocates CPU resources effectively, many processes compete for CPU cycles and this can cause contention. If you installed Oracle9*i* in a multiprocessor environment, there might be a different level of contention on each CPU.

- Oracle resources contention

  Contention is also common for Oracle resources such as locks and latches.

Operating System Tools

Several operating system tools are available to help you 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, context switching, and I/O for the entire system.

Common Tools

The following sections provide information on common tools:

- vmstat
- sar
- iostat
- swap, swapinfo, swapon, and lsps

> **See Also:** The operating system documentation and UNIX man pages for more information on these tools.

vmstat

Use the `vmstat` command to view process, virtual memory, disk, trap, and CPU activity, depending on the switches you supply with the command. Enter one of the following commands to display a summary of CPU activity eight times, at five-second intervals:

- HP and Solaris:

  ```
  $ vmstat -S 5 8
  ```

- AIX, Linux, and Tru64:

  ```
  $ vmstat 5 8
  ```

The following example shows sample output from the command on Solaris:

```
procs     memory            page            disk          faults      cpu
 r b w   swap  free  si  so pi po fr de sr f0 s0 s1 s3   in   sy   cs us sy id
 0 0 0   1892  5864   0   0  0  0  0  0  0  0  0  0  0   90   74   24  0  0 99
 0 0 0  85356  8372   0   0  0  0  0  0  0  0  0  0  0   46   25   21  0  0 100
 0 0 0  85356  8372   0   0  0  0  0  0  0  0  0  0  0   47   20   18  0  0 100
 0 0 0  85356  8372   0   0  0  0  0  0  0  0  0  0  2   53   22   20  0  0 100
 0 0 0  85356  8372   0   0  0  0  0  0  0  0  0  0  0   87   23   21  0  0 100
 0 0 0  85356  8372   0   0  0  0  0  0  0  0  0  0  0   48   41   23  0  0 100
 0 0 0  85356  8372   0   0  0  0  0  0  0  0  0  0  0   44   20   18  0  0 100
 0 0 0  85356  8372   0   0  0  0  0  0  0  0  0  0  0   51   71   24  0  0 100
```

The `w` column, under the `procs` column, shows the number of potential processes that have been swapped out and written to disk. If the value is not zero, swapping is occurring and your system is short of memory.

The `si` and `so` columns under the `page` column indicate the number of swap-ins and swap-outs per second, respectively. Swap-ins and swap-outs should always be zero.

The `sr` column under the `page` column indicates the scan rate. High scan rates are caused by a shortage of available memory.

The `pi` and `po` columns under the `page` column indicate the number of page-ins and page-outs per second, respectively. It is normal for the number of page-ins and page-outs to increase. Some paging always occurs even on systems with plenty of memory available.

Note: The output from the `vmstat` command differs between platforms. See the man page for information on interpreting the output on your platform.

sar

Use the `sar` command to display cumulative activity counters in the operating system, depending on the switches that you supply with the command. The following command displays a summary of I/O activity ten times, at ten-second intervals:

```
$ sar -b 10 10
```

The following example shows sample output from the command on Solaris:

```
13:32:45 bread/s lread/s %rcache bwrit/s lwrit/s %wcache pread/s pwrit/s
13:32:55       0      14     100       3      10      69       0       0
13:33:05       0      12     100       4       4       5       0       0
13:33:15       0       1     100       0       0       0       0       0
13:33:25       0       1     100       0       0       0       0       0
13:33:35       0      17     100       5       6       7       0       0
13:33:45       0       1     100       0       0       0       0       0
13:33:55       0       9     100       2       8      80       0       0
13:34:05       0      10     100       4       4       5       0       0
13:34:15       0       7     100       2       2       0       0       0
13:34:25       0       0     100       0       0     100       0       0

Average        0       7     100       2       4      41       0       0
```

Note: On Tru64 systems, the `sar` command is available in the UNIX SVID2 compatibility subset, OSFSVID2400.

iostat

Use the `iostat` command to view terminal and disk activity, depending on the switches you supply with the command. The output from the `iostat` command does not include disk request queues, but it shows which disks are busy. This information is valuable when you must balance I/O loads.

The following command displays terminal and disk activity five times, at five-second intervals:

```
$ iostat 5 5
```

The following example shows sample output from the command on Solaris:

```
tty          fd0           sd0           sd1           sd3          cpu
 tin tout Kps tps serv  Kps tps serv  Kps tps serv  Kps tps serv  us sy wt id
   0    1   0   0    0    0   0   31    0   0   18    3   0   42   0  0  0 99
   0   16   0   0    0    0   0    0    0   0    0    1   0   14   0  0  0 100
   0   16   0   0    0    0   0    0    0   0    0    0   0    0   0  0  0 100
   0   16   0   0    0    0   0    0    0   0    0    0   0    0   0  0  0 100
   0   16   0   0    0    0   0    0    2   0   14   12   2   47   0  0  1 98
```

Use the `iostat` command to 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/O requests to that disk or by I/O with long average seek times. Ideally, disk request queues should be at or near zero.

swap, swapinfo, swapon, and lsps

Use the `swap`, `swapinfo`, `swapon`, or `lsps` command to report information on swap space usage. A shortage of swap space can prevent processes from spawning, cause slow response times, or cause the system to stop responding. The following table lists the appropriate command to use for each platform:

Platform	Command
AIX	`lsps -a`
HP	`swapinfo -m`
Linux	`swapon -s`
Solaris	`swap -l`
Tru64	`swapon -s`

The following example shows sample output from the `swap -l` command on Solaris:

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

Linux Tools

On Linux systems, use the `free` command to view information on swap space, memory, and buffer usage. A shortage of swap space can prevent processes from spawning, cause slow response times, or cause the system to stop responding.

Solaris Tools

On Solaris systems, use the `mpstat` command to view statistics for each processor in a multiprocessor system. Each row of the table represents the activity of one processor. The first row summarizes all activity since the last system reboot; each subsequent row summarizes activity for the preceding interval. All values are events per second unless otherwise noted. The arguments are for time intervals between statistics and number of iterations. The following example shows sample output from the `mpstat` command:

```
CPU minf mjf xcal  intr ithr  csw icsw migr smtx  srw syscl  usr sys  wt idl
  0    0   0    1    71   21   23    0    0    0    0    55    0   0   0  99
  2    0   0    1    71   21   22    0    0    0    0    54    0   0   0  99
CPU minf mjf xcal  intr ithr  csw icsw migr smtx  srw syscl  usr sys  wt idl
  0    0   0    0    61   16   25    0    0    0    0    57    0   0   0 100
  2    1   0    0    72   16   24    0    0    0    0    59    0   0   0 100
```

AIX Tools

The following sections list tools available on AIX systems.

> **See Also:** The AIX operating system documentation and man pages for more information on these tools.

AIX System Management Interface Tool

The AIX System Management Interface Tool (SMIT) provides a menu-driven interface to various system administrative and performance tools. Using SMIT, you can navigate through large numbers of tools and focus on the jobs you want to execute.

Base Operation System Tools

The AIX Base Operation System (BOS) contains performance tools that are historically part of UNIX systems or are required to manage the implementation-specific features of AIX. The following table lists the most important BOS tools:

Tool	Description
`lsattr`	Displays the attributes of devices
`lslv`	Displays information about a logical volume or the logical volume allocations of a physical volume
`netstat`	Displays the contents of network-related data structures
`nfsstat`	Displays statistics about Network File System (NFS) and Remote Procedure Call (RPC) activity
`nice`	Changes the initial priority of a process
`no`	Displays or sets network options
`ps`	Displays the status of one or more processes
`reorgvg`	Reorganizes the physical-partition allocation within a volume group
`time`	Displays the elapsed execution, user CPU processing, and system CPU processing time
`trace`	Records and reports selected system events
`vmtune`	Changes the operational parameters of the Virtual Memory Manager and other AIX components

AIX Performance Toolbox

The AIX Performance Toolbox (PTX) contains tools for monitoring and tuning system activity locally and remotely. PTX consists of two main components, the PTX Manager and the PTX Agent. The PTX Manager collects and displays data from various systems in the configuration by using the `xmperf` utility. The PTX Agent collects and transmits data to the PTX Manager by using the `xmserd` utility. The PTX Agent is also available as a separate product called Performance Aide for AIX.

Both PTX and Performance Aide include the following monitoring and tuning tools:

Tool	Description
`fdpr`	Optimizes an executable program for a particular workload
`filemon`	Uses the trace facility to monitor and report the activity of the file system
`fileplace`	Displays the placement of a file's blocks within logical or physical volumes
`lockstat`	Displays statistics about contention for kernel locks
`lvedit`	Facilitates interactive placement of logical volumes within a volume group
`netpmon`	Uses the trace facility to report on network I/O and network-related CPU usage
`rmss`	Simulates systems with various sizes of memory for performance testing
`svmon`	Captures and analyzes information about virtual-memory usage
`syscalls`	Records and counts system calls
`tprof`	Uses the trace facility to report CPU usage at module and source-code-statement levels
`BigFoot`	Reports the memory access patterns of processes
`stem`	Permits subroutine-level entry and exit instrumentation of existing executables

See Also: *Performance Toolbox for AIX; Guide and Reference 1.2 and 2* for more information on PTX, and the *AIX Performance Tuning Guide Version 3.2 and 4* and the *AIX5L Performance Management Guide* for information on the syntax of some of these tools.

HP Tools

The following sections list tools available on HP systems.

Performance Tuning Tools

The following table lists the tools that you can use for additional performance tuning on HP:

> **See Also:** The HP-UX operating system documentation and man pages for more information on PTX.

Tools	Description
`gprof`	Creates an execution profile for programs
`monitor`	Monitors the program counter and calls to certain functions
`netfmt`	Monitors the network
`netstat`	Reports statistics on network performance
`nfsstat`	Reports statistics for each processor
`nettl`	Captures network events or packets by logging and tracing
`prof`	Creates an execution profile of C programs and displays performance statistics for your program, showing where your program is spending most of its execution time
`profil`	Copies program counter information into a buffer
`top`	Displays the top processes on the system and periodically updates the information

HP Performance Analysis Tools

The following HP-UX performance analysis tools are available:

- GlancePlus/UX
- HP PAK

GlancePlus/UX

This HP-UX utility is an online diagnostic tool that measures the system's activities. GlancePlus displays how system resources are being used. It displays dynamic information about the system's I/O, CPU, and memory usage in a series of screens. You can also use the utility to monitor how individual processes are using resources.

HP PAK

HP Programmer's Analysis Kit (HP PAK) currently consists of two tools, Puma and Thread Trace Visualizer (TTV):

- Puma collects performance statistics during a program run. It provides several graphical displays for viewing and analyzing the collected statistics.
- TTV displays trace files produced by the instrumented thread library, `libpthread_tr.sl`, in a graphical format. It allows you to view how threads are interacting and to find where threads are blocked waiting for resources.

HP PAK is bundled with the HP FORTRAN 77, HP FORTRAN90, HP C, HP C++, HP ANSI C++, and HP Pascal compilers.

Tuning Memory Management

Start the memory tuning process by measuring paging and swapping space to determine how much memory is available. After you have determined your system's memory usage, tune the Oracle buffer cache.

The Oracle buffer manager ensures that the more frequently accessed data is cached longer. If you monitor the buffer manager and tune the buffer cache, you can have a significant influence on Oracle9*i* performance. The optimal Oracle9*i* buffer size for your system depends on the overall system load and the relative priority of Oracle over other applications.

Allocate Sufficient Swap Space

Try to minimize swapping because it causes significant UNIX overhead. To check for swapping, enter the `sar` or `vmstat` commands. For information on the appropriate options to use with the `sar` or `vmstat` commands, see the man pages.

If your system is swapping and you must 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.

To determine how much swap space is in use, enter one of the following commands:

Platform	Command
AIX	`lsps -a`
HP	`swapinfo -m`
Linux	`swapon -s`
Solaris	`swap -l`
Tru64	`swapon -s`

To add swap space to your system, enter one of the following commands:

Platform	Command
AIX	`chps` or `mkps`
HP	`swapon`
Linux	`swapon -a`
Solaris	`swap -a`
Tru64	`swapon -a`

Set the swap space to between two and four times the system's physical memory. Monitor the use of swap space and increase it as required.

> **See Also:** Your operating system documentation for more information on these commands.

Control Paging

Paging might not present as serious a problem as swapping, because an entire program does not have to be stored in memory to run. A small number of page-outs might 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 the `vmstat` or `sar` command to monitor paging. See the man pages or your operating system documentation for information on interpreting the results for your platform. The following columns from the output of these commands are important on Solaris:

Column	Description
`vflt/s`	Indicates the number of address translation page faults. Address translation faults occur when a process references a valid page not in memory.
`rclm/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 the SGA to use less memory.

Adjust Oracle Block Size

A UNIX system reads entire operating system blocks from the disk. If the database block size is smaller than the UNIX file system buffer size, I/O bandwidth is inefficient. If you adjust the Oracle database block size to be a multiple of the operating system block size, you can increase performance by up to five percent.

The DB_BLOCK_SIZE initialization parameter sets the database block size. You can change the block size by recreating the database.

To see the current value of the DB_BLOCK_SIZE parameter, enter the SHOW PARAMETERS command in SQL*Plus.

Tuning Disk I/O

Balance I/O evenly across all available disks to reduce disk access times. For smaller databases and those not using RAID, ensure that different datafiles and tablespaces are distributed across the available disks.

Choose the Appropriate File System Type

Depending on the operating system that you use, you can choose from a range of file systems. File systems have different characteristics, and the techniques they use to access data can have a substantial impact on database performance. The following table lists typical file system choices and the platforms on which they are available:

File System	Platform	Description
S5	AIX, HP, Solaris	UNIX System V file system
UFS	AIX, HP, Solaris, Tru64	Unified file system, derived from BSD UNIX
VXFS	AIX, Solaris	Veritas file system
Raw	AIX, HP, Linux, Solaris, Tru64	Raw devices/volumes (no file system)
EXT2	Linux	Extended file system for Linux
AdvFS	Tru64	Advanced file system
CFS	Tru64	Cluster file system
JFS	AIX	Journaled file system
JFS2	AIX 5.1	Journaled file system

The suitability of a file system to an application is usually undocumented. For example, even different implementations of the Unified file system are hard to compare. Performance differences may vary from 0 to 20 percent, depending on the file system you choose.

If you choose to use a file system:

- Make a new file system partition to ensure that the hard disk is clean and unfragmented.
- Perform a file system check on the partition before using it for database files.
- Distribute disk I/O as evenly as possible.
- Separate log files from database files.

Monitoring Disk Performance

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

Table 2–1 describes the columns of the `sar -b` command output that are significant for analyzing disk performance.

Table 2–1 sar -b Output Columns

Columns	Description
`bread/s, bwrit/s`	Blocks read and blocks written per second (important for file system databases)
`pread/s, pwrit/s`	Partitions read and partitions written per second (important for raw partition database systems)

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

> **Note:** Not all Linux distributions display the `%wio` column in the output of the `sar -u` command.

Key indicators are:

- The sum of the `bread`, `bwrit`, `pread`, and `pwrit` columns indicates the level of activity of the disk I/O subsystem. The higher the sum, the busier the I/O subsystem. 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` column value should be greater than 90 and the `%wcache` column value should be greater than 60. Otherwise, the system may be disk I/O bound.
- If the `%wio` column value is consistently greater than 20, the system is I/O bound.

Tuning UNIX Kernel Parameters

You can improve performance by keeping all Oracle users and processes at the same priority. The UNIX kernel typically pre-allocates physical memory, leaving less memory available for other processes such as the Oracle processes.

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.

Note: Remember to make a backup copy of your UNIX kernel. See your operating system documentation for information on making a backup copy.

Tuning the Operating System Buffer Cache

To take full advantage of raw devices, adjust the size of the Oracle9*i* buffer cache and, if memory is limited, the operating system buffer cache.

The operating system buffer cache holds blocks of data in memory while they are being transferred from memory to disk, or from disk to memory.

The Oracle9*i* buffer cache is the area in memory that stores the Oracle database buffers. Because Oracle9*i* can use raw devices, it does not need to use the operating system buffer cache.

If you use raw devices, increase the size of the Oracle9*i* buffer cache. If the amount of memory on the system is limited, make a corresponding decrease in the operating system buffer cache size.

Use the `sar` command to determine which buffer caches you must increase or decrease. For more information on the `sar` command, see the UNIX man pages.

Using Raw Devices/Volumes

The following sections provide information on using raw devices/volumes.

Note: For additional raw device/volume tuning information, see the following appendices:

- Appendix A, "Tuning for Oracle9i on AIX"
- Appendix C, "Tuning for Oracle9i on Linux"
- Appendix D, "Tuning for Oracle9i on Tru64"

Guidelines for Using Raw Devices/Volumes

Raw devices/volumes have the following disadvantages when used on UNIX:

- Raw devices/volumes may not solve problems with file size writing limits.

  Note: To display current file size limits, enter the following command:

  ```
  $ ulimit -a
  ```

- Small client systems might not be able to use sufficiently large raw device/volume partitions.
- If a particular disk drive has intense I/O activity and performance would benefit from movement of an Oracle datafile to another drive, it is likely that no acceptably sized section exists on a drive with less I/O activity. It might not be possible to move files to other disk drives if you are using raw devices/volumes.
- Raw devices/volumes may be more difficult to administer than datafiles stored on a file system.

In addition to the factors described in this section, consider the following issues when deciding whether to use raw devices/volumes:

- Oracle9*i* Real Application Clusters installation

  Each instance of Oracle9*i* Real Application Clusters has it's own 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 cluster.

- Raw disk partition availability

  Use raw devices/volumes for Oracle files only if your site has at least as many raw disk partitions as Oracle datafiles. If disk space is a consideration and the raw disk partitions are already formatted, match datafile size to partition size as closely as possible to avoid wasting space.

  You must also consider the performance implications of using all of the disk space on a few disks as opposed to using less space on more disks.

- Logical volume manager

  The logical volume manager manages disk space at a logical level and hides some of the complexity of raw devices. With logical volumes, you can create logical disks based on raw partition availability. The logical volume manager controls fixed-disk resources by:

  – Mapping data between logical and physical storage

  – Allowing data to span multiple disks and to be discontiguous, replicated, and dynamically expanded

- Dynamic performance tuning

  You can optimize disk performance when the database is online by moving files from disk drives with high activity to disk drives with less activity. Most hardware vendors who provide the logical disk facility also provide a graphical user interface you can use for tuning.

- Mirroring and online disk replacement

  You can mirror logical volumes to protect against loss of data. If one copy of a mirror fails, dynamic resynchronization is possible. Some vendors also provide the ability to replace drives online in conjunction with the mirroring facility.

  For Oracle9*i* Real Application Clusters, you can use logical volumes for drives associated with a single UNIX system, as well as those that can be shared with more than one computer of a UNIX cluster. Shared drives allow for all files associated with the Oracle9*i* Real Application Clusters to be placed on these shared logical volumes.

Raw Device Setup

Keep the following items in mind when creating raw devices:

- When creating the volumes, ensure that the owner is `oracle` and the group is `oinstall`.
- 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.

> **See Also:** Your operating system documentation for more information on creating raw devices, and "Setting Up Raw Devices" on page D-15 for more information on creating raw devices on Tru64 systems.

Using Trace and Alert Files

This section describes the trace (or dump) and alert files that Oracle9*i* creates to help you diagnose and resolve operating problems.

Trace Files

Each server and background process can write to an associated trace file. When a process detects an internal error, it writes information on the error to its trace file. The filename format of a trace file is *processname_unixpid_sid*.trc**,** where:

- *processname* is a three or four-character abbreviated process name identifying the Oracle9*i* process that generated the file (for example, pmon, dbwr, ora, or reco)
- *unixpid* is the UNIX process ID number
- *sid* is the instance system identifier

A sample trace filename is
$ORACLE_BASE/admin/TEST/bdump/lgwr_1237_TEST.trc.

All trace files for background processes are written to the destination directory specified by the BACKGROUND_DUMP_ DEST initialization parameter. If you do not set this initialization parameter, the default directory is
$ORACLE_HOME/rdbms/log.

All trace files for server processes are written to the destination directory specified by the USER_DUMP_DEST initialization parameter. Set the MAX_DUMP_FILE initialization parameter to at least 5000 to ensure that the trace file is large enough to store error information.

Alert Files

The alert_*sid*.log file stores significant database events and messages. Anything that affects the database instance or global database is recorded in this file. This file is associated with a database and is located in the directory specified by the BACKGROUND_DUMP_DEST initialization parameter. If you do not set this initialization parameter, the default directory is $ORACLE_HOME/rdbms/log.

3

Administering SQL*Plus and *i*SQL*Plus

This chapter describes how to use and administer SQL*Plus and *i*SQL*Plus on Oracle9*i*. It contains the following sections:

- Administering Command-Line SQL*Plus
- Administering iSQL*Plus
- Using Command-Line SQL*Plus
- SQL*Plus Restrictions

> **See Also:** *SQL*Plus Users Guide and Reference* for more information on *i*SQL*Plus and SQL*Plus.

Administering Command-Line SQL*Plus

This section describes how to administer command-line SQL*Plus. In the examples in this section, SQL*Plus uses the value of the ORACLE_HOME environment variable wherever a question mark (?) appears.

Using Setup Files

When you start SQL*Plus, it executes the `glogin.sql` site profile set-up file and then executes the `login.sql` user profile set-up file.

Using the Site Profile File

The global site profile file is `$ORACLE_HOME/sqlplus/admin/glogin.sql`. If a site profile already exists at this location, it is overwritten when you install SQL*Plus. If SQL*Plus is removed, the site profile file is deleted.

Using the User Profile File

The user profile file is `login.sql`. SQL*Plus looks for this file in the current directory, and then in the directories you specify using the SQLPATH environment variable. Set this environment variable to a colon-separated list of directories. SQL*Plus searches these directories for the `login.sql` file in the order they are listed.

The options set in the `login.sql` file override those set in the `glogin.sql` file.

> **See Also:** *SQL*Plus User's Guide and Reference* for more information on profile files.

Using the PRODUCT_USER_PROFILE Table

Oracle9*i* provides the PRODUCT_USER_PROFILE table that you can use to disable the SQL and SQL*Plus commands that you specify. This table is created automatically when you choose an installation type that installs a starter database. A started database is not installed, and demonstration tables are not created, in the following circumstances:

- If you select the Software Only database configuration option
- If you do not select the Demo Schemas option during a Custom installation

> **See Also:** *Oracle9i Installation Guide Release 2 (9.2.0.1.0) for UNIX Systems* for information on installation options.

To recreate the PRODUCT_USER_PROFILE table, run the $ORACLE_HOME/sqlplus/admin/pupbld.sql script in the SYSTEM schema:

```
$ORACLE_HOME/sqlplus/admin/pupbld.sql
```

For example, enter the following commands, where *SYSTEM_PASSWORD* is the password of the SYSTEM user:

```
$ sqlplus SYSTEM/SYSTEM_PASSWORD
SQL> @?/sqlplus/admin/pupbld.sql
```

You can also recreate the PRODUCT_USER_PROFILE table manually in the SYSTEM schema using the $ORACLE_HOME/bin/pupbld shell script. This script prompts for the SYSTEM password. If you need to run the pupbld script without interaction, set the SYSTEM_PASS environment variable to the SYSTEM username and password. For example, enter the following command where *SYSTEM_PASSWORD* is the password of the SYSTEM user:

```
$ SYSTEM_PASS=SYSTEM/SYSTEM_PASSWORD; export SYSTEM_PASS
```

Using Demonstration Tables

Oracle9*i* provides demonstration tables that you can use for testing. These demonstration tables are created automatically when you choose an installation type that installs a starter database. A started database is not installed, and demonstration tables are not created, in the following circumstances:

- If you select the Software Only database configuration option
- If you do not select the Demo Schemas option during a Custom installation

> **See Also:** *Oracle9i Sample Schemas* guide for information on demonstration tables and the *Oracle9i Installation Guide Release 2 (9.2.0.1.0) for UNIX Systems* for information on installation options.

Using EMP and DEPT Tables

This section describes how to manually create and delete the EMP and DEPT demonstration tables.

Creating Demonstration Tables Manually

Use the $ORACLE_HOME/sqlplus/demo/demobld.sql SQL script to create the EMP and DEPT demonstration tables. In SQL*Plus, you can use any username to run the demobld.sql script to create the demonstration tables in a schema. For example, enter:

```
$ sqlplus SCOTT/TIGER
SQL> @?/sqlplus/demo/demobld.sql
```

You can also use the $ORACLE_HOME/bin/demobld shell script to run the demobld.sql script, as follows:

```
$ demobld SCOTT TIGER
```

Deleting Demonstration Tables

Use the $ORACLE_HOME/sqlplus/demo/demodrop.sql script to drop the EMP and DEPT demonstration tables. In SQL*Plus, you can use any username to drop the demonstration tables from the user's schema. For example, enter:

```
$ sqlplus SCOTT/TIGER
SQL> @?/sqlplus/demo/demodrop.sql
```

You can also use the $ORACLE_HOME/bin/demodrop shell script to run the demodrop.sql script, as follows:

```
$ demodrop SCOTT TIGER
```

SQL*Plus Command-Line Help

This section describes how to install and remove the SQL*Plus command-line help.

> **See Also:** *SQL*Plus User's Guide and Reference* for more information on the SQL*Plus command-line help.

Installing the SQL*Plus Command-Line Help

There are three ways to install the SQL*Plus command-line help:

- Perform an installation that installs a starter database.

  When you copy a starter database with pre-built datafiles as part of an installation, SQL*Plus automatically installs the SQL*Plus command-line help in the SYSTEM schema.

- Install the command-line help manually in the SYSTEM schema using the `$ORACLE_HOME/bin/helpins` shell script.

  The `helpins` script prompts for the SYSTEM password. If you need to run this script without interaction, set the SYSTEM_PASS environment variable to the SYSTEM username and password. For example, enter the following command where *SYSTEM_PASSWORD* is the password of the SYSTEM user:

  ```
  $ SYSTEM_PASS=SYSTEM/SYSTEM_PASSWORD; export SYSTEM_PASS
  ```

- Install the command-line help manually in the SYSTEM schema using the `$ORACLE_HOME/sqlplus/admin/help/helpbld.sql` script.

  For example, enter the following commands, where *SYSTEM_PASSWORD* is the password of the SYSTEM user:

  ```
  $ sqlplus SYSTEM/SYSTEM_PASSWORD
  SQL> @?/sqlplus/admin/help/helpbld.sql ?/sqlplus/admin/help helpus.sql
  ```

> **Note:** Both the `helpins` shell script and the `helpbld.sql` script drop existing command-line help tables before creating new tables.

Removing the SQL*Plus Command-Line Help

To manually drop the SQL*Plus command-line help tables from the SYSTEM schema, run the `$ORACLE_HOME/sqlplus/admin/help/helpdrop.sql` script. For example, enter the following commands, where *SYSTEM_PASSWORD* is the password of the SYSTEM user:

```
$ sqlplus SYSTEM/SYSTEM_PASSWORD
SQL> @?/sqlplus/admin/help/helpdrop.sql
```

Administering *i*SQL*Plus

*i*SQL*Plus is a browser-based interface that uses the SQL*Plus processing engine in the following three-tier model:

Tier	Description
Client	*i*SQL*Plus user interface, typically a Web browser
Middle	*i*SQL*Plus server, Oracle Net, and Oracle HTTP Server
Database	Oracle9*i* database

Disabling and Re-enabling *i*SQL*Plus

When you install Oracle9*i*, *i*SQL*Plus is enabled by default. This section describes how to disable and re-enable *i*SQL*Plus.

To disable *i*SQL*Plus:

1. Log in to the system running the Oracle HTTP Server as the Oracle software owner (`oracle`).
2. Change directory to the Oracle HTTP Server configuration directory:

   ```
   $ cd $ORACLE_HOME/Apache/Apache/conf
   ```

3. Open the `oracle_apache.conf` configuration file in any text editor.
4. Insert a comment character (#) before the directive that includes the `isqlplus.conf` configuration file, as follows:

   ```
   #include "ORACLE_HOME/sqlplus/admin/isqlplus.conf"
   ```

   In the preceding example, *`ORACLE_HOME`* is the path of the Oracle home directory.
5. Save the file and exit from the text editor.
6. Verify the changes by parsing the Oracle HTTP Server configuration file:

   ```
   $ $ORACLE_HOME/Apache/Apache/bin/apachectl configtest
   ```

   If necessary, fix any errors displayed by this command.
7. Restart the Oracle HTTP Server:

   ```
   $ $ORACLE_HOME/Apache/Apache/bin/apachectl restart
   ```

To re-enable *i*SQL*Plus, repeat the previous procedure, but in step 4, remove the comment character (#).

Editing the *i*SQL*Plus Configuration File

You can change the values of the following *i*SQL*Plus server parameters in the `$ORACLE_HOME/sqlplus/admin/isqlplus.conf` configuration file:

Parameter	Description
iSQLPlusNumberOfThreads	Sets the maximum number of simultaneous HTTP requests that can be handled by the *i*SQL*Plus server.
iSQLPlusLogLevel	Enables *i*SQL*Plus to maintain log files and determines to what level log files are maintained. The default log file location is `$ORACLE_HOME/sqlplus/log/isqlplus/log.xml`. *i*SQL*Plus must be able to write to the `$ORACLE_HOME/sqlplus/log` directory. This directory has owner write permissions. If *i*SQL*Plus is running as the `nobody` user and therefore cannot write to the log file, you must manually give *i*SQL*Plus write permissions on the log file. It is not recommended to grant world write permission to the file except for temporary testing or tracing purposes.
iSQLPlusTimeOutInterval	Sets the time a session can be idle before it is expired. Timing out *i*SQL*Plus sessions helps reduce system load and maximize resources.
iSQLPlusHashTableSize	Sets the maximum number of concurrent *i*SQL*Plus sessions that can be handled by the *i*SQL*Plus server. The default value is derived from the value of the iSQLPlusNumberOfThreads parameter.
iSQLPlusConnectIdList	Sets a drop-down list of databases that users can access in *i*SQL*Plus, in place of the Connection Identifier text field on the Login screen. This allows greater security for *i*SQL*Plus servers in hosted environments.

Parameter	Description
iSQLPlusAllowUserEntMap	Controls whether *i*SQL*Plus permits users to change the SET MARKUP HTML ENTMAP and COLUMN ENTMAP settings. In HTML, entity mapping replaces characters of special significance with printable representations of those characters. By default, entity mapping is enabled, and the value of the iSQLPlusAllowUserEntMap parameter is set to `none`. This value prevents users from changing the entity mapping settings, and prevents the use of user-defined HTML in *i*SQL*Plus output. If you set the value of the iSQLPlusAllowUserEntMap parameter to `all`, users can change entity mapping settings.
`-idle-timeout`	Sets the time the Oracle HTTP Server waits for results from *i*SQL*Plus.

The following example shows a sample FastCgiServer directive in the `isqlplus.conf` file:

```
FastCgiServer ORACLE_HOME/isqlplus -initial-env
iSQLPlusNumberOfThreads=20 -initial-env iSQLPlusTimeOutInterval=30 -initial-env
iSQLPlusLogLevel=warn -idle-timeout 3600
```

In the preceding example, *ORACLE_HOME* is the path of the Oracle home directory.

Security

This section describes security issues specific to *i*SQL*Plus.

> **See Also:** *SQL*Plus User's Guide and Reference* for more information on *i*SQL*Plus security.

Configuring Oracle HTTP Server Authentication for *i*SQL*Plus

Users can access *i*SQL*Plus in two ways:

- As SYSDBA or SYSOPER privileged users
- As normal users, without SYSDBA or SYSOPER privileges

To access *i*SQL*Plus as a SYSDBA or SYSOPER privileged user, you use a different URL to the URL you use to connect as a normal user. By default, the URL used by privileged users is protected using Oracle HTTP Server authentication. If required, you can also use Oracle HTTP Server authentication to protect the URL used by normal users.

To access a protected *i*SQL*Plus URL, users must specify a username and password, which is separate from their Oracle username and password, before they can access the login screen. These usernames and passwords are stored in a password file.

An empty password file for SYSDBA and SYSOPER privileged users is installed when you install Oracle9*i*. Before accessing *i*SQL*Plus as a privileged user for the first time, you must add one or more usernames and passwords to this password file.

Similarly, if you want to use Oracle HTTP Server authentication for normal users, you must create a separate password file and add usernames and passwords to it. You must then modify the *i*SQL*Plus configuration file to use this file to authenticate normal users.

The following sections describe how to add users to password files and if necessary, modify the *i*SQL*Plus configuration file.

Adding Usernames and Passwords to a Password File

To add usernames and passwords to a password file:

1. Log in to the system running the Oracle HTTP Server as the Oracle software owner (`oracle`).
2. Change directory to the `$ORACLE_HOME/Apache/Apache/bin` directory:

   ```
   $ cd $ORACLE_HOME/Apache/Apache/bin
   ```

3. Enter one of the following commands to add a username and password to the appropriate password file:

   Note: You do not need to create the password file for privileged users. This file, `iplusdba.pw`, is installed by default.

   - To create a password file for normal users and add a username and password to it, enter:

     ```
     $ htpasswd -c $ORACLE_HOME/sqlplus/admin/filename.pw username
     ```

- To add a username and password to an existing password file, enter:

  ```
  $ htpasswd $ORACLE_HOME/sqlplus/admin/filename.pw username
  ```

  In the previous examples, *filename*.pw is the name of the password file that you want to modify or create, and *username* is the username that you want to add. The filename of the privileged user password file is iplusdba.pw. For unprivileged users, Oracle Corporation recommends that you use the name iplus.pw for the password file.

4. When prompted, enter the password for the username you specified.
5. If you created a new password file, see the following section for information on how to configure Oracle HTTP Server to use it.

Configuring the Oracle HTTP Server to Use a New Password File

When you create a new password file for normal users, you must configure the Oracle HTTP Server to use this password file to authenticate users that access the *i*SQL*Plus URL.

To configure the Oracle HTTP Server to use the new password file:

1. Log into the system running the Oracle HTTP Server as the Oracle software owner (oracle).
2. Change directory to the $ORACLE_HOME/sqlplus/admin directory:

   ```
   $ cd $ORACLE_HOME/sqlplus/admin
   ```

3. Open the isqlplus.conf configuration file in any text editor.
4. Locate the following section in the file:

   ```
   #
   # Enable handling of all virtual paths beginning with "/isqlplus"
   #
   <Location /isqlplus>
     SetHandler fastcgi-script
     Order deny,allow

     # Comment "Allow ..." and uncomment the four lines "AuthType ..."
     # to "Require ..." if Oracle HTTP authentication access is required
     # for the http://.../isqlplus URL
     Allow from all
     #AuthType Basic
   ```

```
  #AuthName 'iSQL*Plus'
  #AuthUserFile ORACLE_HOME/sqlplus/admin/iplus.pw
  #Require valid-user
</Location>
```

In this example, *ORACLE_HOME* is the path of the Oracle home directory

5. Modify this section as shown in the following example:

```
#
# Enable handling of all virtual paths beginning with "/isqlplus"
#
<Location /isqlplus>
 SetHandler fastcgi-script
 Order deny,allow
 AuthType Basic
 AuthName 'iSQL*Plus'
 AuthUserFile ORACLE_HOME/sqlplus/admin/filename.pw
 Require valid-user
</Location>
```

In this example, *ORACLE_HOME* is the path of the Oracle home directory and *filename*.pw is the name of the password file that you created for normal users (typically `iplus.pw`).

6. Verify the changes by parsing the Oracle HTTP Server configuration file:

```
$ $ORACLE_HOME/Apache/Apache/bin/apachectl configtest
```

If necessary, fix any errors displayed by this command.

7. Restart the Oracle HTTP Server:

```
$ $ORACLE_HOME/Apache/Apache/bin/apachectl restart
```

After the Oracle HTTP Server restarts, users accessing the *i*SQL*Plus URL are prompted for the Oracle HTTP Server authentication username and password before the browser displays the login screen.

Restricting Database Access from *i*SQL*Plus

You can restrict access to databases from *i*SQL*Plus. When restricted database access is enabled, a drop-down list of available databases is displayed in place of the Connection Identifier text field on the Login screen. This allows greater security for *i*SQL*Plus servers in hosted environments. Connection identifiers are listed in the order defined in the `iSQLPlusConnectIdList` parameter.

You can edit the `isqlplus.conf` file to enforce restricted database access by changing the following line, where *`SID1, SID2, ...`* is a comma separated list of Oracle Net connection identifiers specifying permitted databases:

```
FastCgiServer ... -initial-env "iSQLPlusConnectIdList=SID1, SID2,..."
```

For example:

```
FastCgiServer ... -initial-env "iSQLPlusConnectIdList=ABC1, PROD2, DEV3"
```

> **See Also:** "Editing the iSQL*Plus Configuration File" on page 3-7 for information on editing the `isqlplus.conf` file.

While no quotes or embedded whitespace is allowed in a connection identifier, quotes are required around the entire `iSQLPlusConnectIdList=` argument, as shown in the preceding example. Connection identifiers are not case sensitive, and each connection identifier listed in the argument should be identical to an alias defined in the `tnsnames.ora` file.

Once set, all connections made through the Login screen, all Dynamic Reports and any connections attempted with the CONNECT command are refused unless the connection is to one of the databases specified in the restricted list.

Similarly, if you use the SET INSTANCE system variable, the connection identifier defined must match an entry in the `iSQLPlusConnectIdList` parameter or the connection is refused.

If no connection identifier is given, or if the one given does not match an entry in the `iSQLPlusConnectIdList` parameter, the database connection is refused and the following error occurs:

```
SP2-0884: Connection to database database_name is not allowed
```

In the preceding example, *`database_name`* is the name of the database that you are trying to connect to.

Using Command-Line SQL*Plus

This section describes how to use command-line SQL*Plus on UNIX systems.

Using a System Editor from SQL*Plus

If you enter an ED or EDIT command at the SQL*Plus prompt, the system starts an operating system editor, such as `ed, emacs, ned`, or `vi`. The PATH variable must include the directory where the editor executable is located.

When you start the editor, the current SQL buffer is placed in the editor. When you exit the editor, the changed SQL buffer is returned to SQL*Plus.

You can specify which editor starts by defining the SQL*Plus _EDITOR variable. You can define this variable in the `glogin.sql` site profile, the `login.sql` user profile, or define it during the SQL*Plus session. For example, to set the default editor to `vi`, enter:

```
SQL> DEFINE _EDITOR=vi
```

If you do not set the _EDITOR variable, the value of either the EDITOR or the VISUAL environment variable is used. If both environment variables are set, the value of the EDITOR variable is used. When _EDITOR, EDITOR, and VISUAL are not specified, the default editor is `ed`.

If you start the editor, SQL*Plus uses the `afiedt.buf` temporary file to pass text to the editor. You can use the SET EDITFILE command to specify a different filename. For example, enter:

```
SQL> SET EDITFILE /tmp/myfile.sql
```

SQL*Plus does not delete the temporary file.

Running Operating System Commands from SQL*Plus

Using the HOST command or an exclamation mark (!) as the first character after the SQL*Plus prompt causes subsequent characters to be passed to a sub-shell. The SHELL environment variable sets the shell used to execute operating system commands. The default shell is the Bourne shell (`/bin/sh`). If the shell cannot be executed, an error message is displayed.

To return to SQL*Plus, enter the `exit` command or press Ctrl+d.

For example, to execute one command, enter:

```
SQL>! command
```

In the preceding example, *command* represents the operating system command that you want to execute.

To execute multiple operating system commands from SQL*Plus, enter the HOST or ! command then press Return. SQL*Plus returns you to the operating system prompt.

Interrupting SQL*Plus

While running SQL*Plus, you can stop the scrolling record display and terminate a SQL statement by pressing Ctrl+c.

Using the SPOOL Command

The default file extension of files generated by the SPOOL command is `.lst`. To change this extension, specify a spool file containing a period (.). For example, enter:

```
SQL> SPOOL query.txt
```

SQL*Plus Restrictions

This section describes SQL*Plus restrictions.

Resizing Windows

The default values for the SQL*Plus LINESIZE and PAGESIZE system 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.

Hiding Your Password

If you set the SYSTEM_PASS environment variable to the username and password of the SYSTEM user, the output from the `ps` command might display this information. To prevent unauthorized access, enter the SYSTEM password only when prompted by SQL*Plus.

If you want to automatically run a script, consider using an authentication method that does not require you to store a password, for example, externally authenticated logins to Oracle9*i*. If you have a low security environment, you might consider using UNIX pipes in script files to pass a password to SQL*Plus, for example:

```
$ echo SYSTEM_PASSWORD | sqlplus SYSTEM @MYSCRIPT
```

Alternatively, enter the following lines at the command prompt:

```
$ sqlplus <<EOF
SYSTEM/SYSTEM_PASSWORD
SELECT ...
EXIT
EOF
```

In the preceding examples, *SYSTEM_PASSWORD* is the password of the SYSTEM user.

4

Using Oracle Precompilers and the Oracle Call Interface

This chapter describes Oracle Precompilers and the Oracle Call Interface. It contains the following sections:

- Overview of Oracle Precompilers
- Support for 32-Bit and 64-Bit Client Applications (AIX, HP, and Solaris 64-Bit Only)
- Pro*C/C++ Precompiler
- Pro*COBOL Precompiler (AIX, HP, Solaris, and Tru64 Only)
- Pro*FORTRAN Precompiler (AIX, HP, Solaris, and Tru64 Only)
- SQL*Module for Ada (Solaris 32-Bit and AIX Only)
- Oracle Call Interface
- Custom Make Files
- Correcting Undefined Symbols (Solaris Only)
- Multi-threaded Applications
- Using Signal Handlers
- XA Functionality

See Also: "Using Demonstration Tables" on page 3-3 for information on using SQL*Plus to create the demonstration tables.

Overview of Oracle Precompilers

Oracle precompilers are application-development tools used to combine SQL statements for 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 Oracle9*i* or any other ANSI SQL database management system.

Precompiler Configuration Files

Configuration files for the Oracle precompilers are located in the `$ORACLE_HOME/precomp/admin` directory. Table 4–1 lists the names of the configuration files for each precompiler.

Table 4–1 System Configuration Files for Oracle Precompilers

Product	Configuration File
Pro*C / C++ release 9.2.0.1.0	`pcscfg.cfg`
Pro*COBOL release 9.2.0.1.0 (AIX, HP, Solaris, and Tru64 only)	`pcbcfg.cfg`
Pro*COBOL release 1.8.77.0.0 (AIX, HP, Solaris, and Tru64 only)	`pcccob.cfg`
Pro*FORTRAN release 1.8.77.0.0 (AIX, HP, Solaris, and Tru64 only)	`pccfor.cfg`
Object Type Translator release 9.2.0.1.0	`ottcfg.cfg`
Oracle SQL*Module for Ada release 9.2.0.1.0 (Solaris 32-bit and AIX only)	`pmscfg.cfg`

Relinking Precompiler Executables

Use the `$ORACLE_HOME/precomp/lib/ins_precomp.mk` make file to relink all precompiler executables. To manually relink a particular precompiler executable, enter the following command:

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

This command creates the new executable in the `$ORACLE_HOME/precomp/lib` directory, and then moves it to the `$ORACLE_HOME/bin` directory. To create the new executable without moving it to the `$ORACLE_HOME/bin` directory, enter the following command:

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

In the preceding examples, *executable* is a product executable listed in Table 4–2.

Table 4–2 Products and Their Corresponding Executable

Product	Executable
Pro*C / C++ release 9.2.0.1.0	`proc`
Pro*COBOL release 9.2.0.1.0 (AIX, HP, Solaris, and Tru64 only)	`procob` or `rtsora`
Pro*COBOL release 1.8.77.0.0 (AIX, HP, Solaris, and Tru64 only)	`procob18` or `rtsora`
Pro*FORTRAN release 1.8.77.0.0 (AIX, HP, Solaris, and Tru64 only)	`profor`
Object Type Translator release 9.2.0.1.0	`ott`
Oracle SQL*Module for Ada release 9.2.0.1.0 (Solaris 32-bit and AIX only)	`modada`

Precompiler README Files

Table 4–3 lists the location of the precompiler README files. The README files describe changes made to the precompiler since the last release.

Table 4–3 Location of Precompiler README Files

Precompiler	README File
Pro*C / C++ release 9.2.0.1.0	`$ORACLE_HOME/precomp/doc/proc2/readme.doc`
Pro*COBOL release 9.2.0.1.0	`$ORACLE_HOME/precomp/doc/procob2/readme.doc`
Pro*COBOL release 1.8.77.0.0 and Pro*FORTRAN release 1.8.77.0.0	`$ORACLE_HOME/precomp/doc/pro1x/readme.txt`

Issues Common to All Precompilers

The following issues are common to all precompilers.

Note: To run Oracle Precompiler demonstrations, Oracle9*i* must already be installed.

Uppercase to Lowercase Conversion

In languages other than C, the compiler converts an uppercase function or subprogram name to lowercase. This can cause a "No such user exists" error message. If you receive this error message, verify that the function or subprogram name in your option file matches the case used in the IAPXTB table.

Vendor Debugger Programs

Precompilers and vendor-supplied debuggers can be incompatible. Oracle Corporation does not guarantee that a program run using a debugger will perform the same way when it is run without the debugger.

Value of IRECLEN and ORECLEN

The IRECLEN and ORECLEN parameters do not have maximum values.

Static and Dynamic Linking

You can statically or dynamically link Oracle libraries with precompiler and OCI applications. 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 very large.

With dynamic linking, the executing code is partly stored in the executable program and partly stored in libraries that are linked dynamically by the application at runtime. Libraries that are linked at runtime are called dynamic or shared libraries. The benefits of dynamic linking are:

- Smaller disk requirements—More than one application or invocation of the same application can use the same dynamic library.
- Smaller main memory requirements—The same dynamic library image is loaded into main memory only once and it can be shared by more than one application.

Client Shared Library

The client shared library is located in the $ORACLE_HOME/lib directory. If you use the Oracle provided demo_*product*.mk make file to link an application, the client shared library is linked by default.

You might receive one of the following error message when starting an executable:

- On AIX systems:

```
$ sample1
exec(): 0509-036 Cannot load program ./sample1 because of the following
errors:
0509-022 Cannot load library libclntsh.a [shr.o]
0509-026 System error: A file or directory in the pathname does not exist.
```

- On HP systems:

```
$ sample1
/usr/lib/dld.sl: Can't open shared library:
/u01/app/oracle/product/9.2.0.1.0/lib/libclntsh.sl.9.0
/usr/lib/dld.sl: No such file or directory
Abort (core dumped)
```

- On Solaris and Linux systems:

```
$ sample1
ld.so.1: sample1: fatal: libclntsh.so.1.0: can't open file: errno=2
Killed
```

- On Tru64 systems:

```
$ sample1
/sbin/loader: Fatal Error: Cannot map libclntsh.so
Killed
```

If you receive one of these error messages, set one of the following environment variables, depending on your platform:

- On AIX systems:

```
$ LIBPATH=$ORACLE_HOME/lib:${LIBPATH}
$ export LIBPATH
```

- On HP systems (64-bit applications), Linux, Solaris, and Tru64 systems:

```
$ LD_LIBRARY_PATH=$ORACLE_HOME/lib:${LD_LIBRARY_PATH}
$ export LD_LIBRARY_PATH
```

- On HP systems (32-bit applications)

```
$ SHLIB_PATH=$ORACLE_HOME/lib32:${SHLIB_PATH}
$ export SHLIB_PATH
```

The client shared library is created automatically during installation. If you must recreate the client shared library:

1. Exit all client applications that use the client shared library, including all Oracle client applications such as SQL*Plus and Recovery Manager.
2. Log in as the `oracle` user and enter:

   ```
   $ genclntsh
   ```

Support for 32-Bit and 64-Bit Client Applications (AIX, HP, and Solaris 64-Bit Only)

Oracle9*i* release 2 (9.2.0.1.0) for AIX, HP, and Solaris 64-bit systems provides support for 32-bit and 64-bit client applications. By default, all demonstrations and client applications provided with release 2 (9.2.0.1.0) link and run in 64-bit mode. However, you can build 32-bit and 64-bit client applications in the same ORACLE home directory.

You can run both 32-bit and 64-bit applications on Oracle9*i* release 2 (9.2.0.1.0) for AIX, HP, and Solaris 64-bit systems. The following table lists the 32-bit and 64-bit client shared libraries:

Platform	32-Bit Client Shared Library	64-Bit Client Shared Library
AIX	`$ORACLE_HOME/lib32/libclntsh.a`	`$ORACLE_HOME/lib/libclntsh.a`
HP	`$ORACLE_HOME/lib32/libclntsh.sl`	`$ORACLE_HOME/lib/libclntsh.sl`
Solaris 64-bit	`$ORACLE_HOME/lib32/libclntsh.so`	`$ORACLE_HOME/lib/libclntsh.so`

To implement a mixed word-size installation:

1. Execute the following command to generate the 32-bit and 64-bit client shared libraries:

   ```
   $  genclntsh
   ```

2. Include the paths of the required 32-bit and 64-bit shared libraries in one of the following environment variables, depending on your platform:

Platform	Environment Variable
AIX, Solaris 64-bit	LIBPATH
HP (32-bit client applications)	SHLIB_PATH
HP (64-bit client applications)	LD_LIBRARY_PATH

Building 32-Bit Pro*C and OCI Customer Applications

Oracle9*i* release 2 (9.2.0.1.0) supports both 32-bit and 64-bit Pro*C and Oracle Call Interface (OCI) customer applications.

> **See Also:** `$ORACLE_HOME/precomp/demo/demo_proc.mk` and `$ORACLE_HOME/precomp/demo/demo_proc32.mk` files for information on building 32-bit Pro*C applications.
>
> See the `$ORACLE_HOME/rdbms/demo/demo_rdbms.mk` and `$ORACLE_HOME/rdbms/demo/demo_rdbms32.mk` files for information on building 32-bit Pro*C applications.

32-Bit Executables and Directories (AIX and HP Only)

In Oracle9*i* release 2 (9.2.0.1.0) for HP and AIX systems, the following directories contain 32-bit executables and libraries:

- `$ORACLE_HOME/lib32`
- `$ORACLE_HOME/rdbms/lib32`
- `$ORACLE_HOME/hs/lib32`
- `$ORACLE_HOME/network/lib32`
- `$ORACLE_HOME/precomp/lib32`
- `$ORACLE_HOME/bin`

Pro*C/C++ Precompiler

Before you use the Pro*C/C++ precompiler, verify that the correct version of the operating system compiler is properly installed.

> **See Also:** *Oracle9i Installation Guide Release 2 (9.2.0.1.0) for UNIX Systems* for information on the required compiler versions on each platform, and the *Pro*C/C++ Precompiler Programmer's Guide* for additional information on the Pro*C/C++ precompiler and interface features.

Pro*C/C++ Demonstration Programs

Demonstration programs are provided to show the features of the Pro*C/C++ precompiler. There are three types of demonstration programs: C, C++, and Object programs. Object programs demonstrate the new Oracle9*i* Object features. All of the demonstration programs are located in the `$ORACLE_HOME/precomp/demo/proc` directory. By default, all programs are dynamically linked with the client shared library.

The programs assume that the demonstration tables created by the `$ORACLE_HOME/sqlplus/demo/demobld.sql` script exist in the SCOTT schema with the password TIGER.

Use the `demo_proc.mk` make file, located in the `$ORACLE_HOME/precomp/demo/proc/` directory, 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
```

To create all of the C demonstration programs for Pro*C/C++, enter:

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

To create all of the C++ demonstration programs for Pro*C/C++, enter:

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

To create all of the Object demonstration programs for Pro*C/C++, enter:

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

Some demonstration programs require you to run a SQL script, located in the $ORACLE_HOME/precomp/demo/sql directory. If you do not run the script, a message displays requesting you to run it. To build a demonstration program and run the corresponding SQL script, include the make macro argument RUNSQL=run on the command line. For example, to create the calldemo demonstration program and run the required $ORACLE_HOME/precomp/demo/sql/calldemo.sql script, enter:

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

To create all of the Object demonstration programs and run all corresponding required SQL scripts, enter:

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

Pro*C/C++ User Program

You can use the $ORACLE_HOME/precomp/demo/proc/demo_proc.mk make file to create user programs. The syntax for creating a program with the demo_proc.mk make file is:

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

In the preceding example:

- *target* is the make file target that you want to use
- *objfilen* is the object file to link the program
- *exename* is the executable program

For example, to create the program myprog from the Pro*C/C++ source file myprog.pc, enter one of the following commands, depending on the source and the type of executable you want to create.

- For C source, dynamically linked with the client shared library, enter:

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

- For C source, statically linked, enter:

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

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

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

- For C++ source, statically linked, enter:

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

> **Note:** On AIX, HP, and Solaris 64-bit systems, the `demo_proc.mk` make file builds 64-bit user programs by default. You can also use the `demo_proc32.mk` make file to build 32-bit user programs. See the make file for more information on creating 32-bit user programs.

Pro*COBOL Precompiler (AIX, HP, Solaris, and Tru64 Only)

There are two versions of Pro*COBOL included with this release. Table 4–4 shows the naming conventions for each version.

Table 4–4 Pro*COBOL Naming Conventions

Item	Pro*COBOL Release 9.2.0.1.0	Pro*COBOL Release 1.8.77.0.0
Executable	`procob`	`procob18`
Demonstration Directory	`procob2`	`procob`
Make file for Merant Server Express COBOL	`demo_procob.mk`	`demo_procob18.mk`

Pro*COBOL supports statically linked, dynamically linked, or dynamically loadable programs. Dynamically linked programs use the client shared library. Dynamically loadable programs use the `rtsora` executable located in the `$ORACLE_HOME/bin` directory.

Pro*COBOL Environment Variables

This section describes the environment variables required by Pro*COBOL.

Merant Server Express COBOL Compiler

For the Merant Server Express COBOL compiler, you must set the COBDIR environment variable and the LD_LIBRARY_PATH, LIBPATH, or SHLIB_PATH environment variable, depending on your operating system.

COBDIR Set the COBDIR environment variable to the directory where the compiler is installed. For example, if the compiler is installed in the /opt/cobol directory, enter:

```
$ COBDIR=${COBDIR}:$/opt/cobol
$ export COBDIR
```

LD_LIBRARY_PATH (Solaris and HP with 64-Bit Applications Only) Set the LD_LIBRARY_PATH environment variable to the directory where the compiler is installed. For example, if the compiler is installed in the $COBDIR/coblib directory, enter:

```
$ LD_LIBRARY_PATH=${LD_LIBRARY_PATH}:$COBDIR/coblib
$ export LD_LIBRARY_PATH
```

LIBPATH (AIX only) Set the LIBPATH environment variable to the directory where the compiler is installed. For example, if the compiler is installed in the $COBDIR/coblib directory, enter:

```
$ LIBPATH=${LIBPATH}:$COBDIR/coblib
$ export LIBPATH
```

SHLIB_PATH (HP with 32-Bit Applications) Set the SHLIB_PATH environment variable to the directory where the compiler is installed. For example, if the compiler is installed in the $COBDIR/coblib directory, enter:

```
$ SHLIB_PATH=${SHLIB_PATH}:$COBDIR/coblib
$ export SHLIB_PATH
```

If the LD_LIBRARY_PATH, LIBPATH, or SHLIB_PATH environment variable setting does not include the $COBDIR/coblib directory, one of the following error messages appears when you compile a program:

- On Tru64:

  ```
  14783 rtsora: /sbin/loader: Fatal Error: cannot map libwtc9.so
  ```

- On AIX, HP, and Solaris:

  ```
  ld.so.1: rts32: fatal: libfhutil.so.2.0: can't open file: errno=2
  ```

PATH Set the PATH environment variable to include the `/opt/SUNWnsun/bin` directory, as follows:

```
$ PATH ${PATH}:/opt/SUNWnsun/bin
$ export PATH
```

LD_LIBRARY_PATH Set the LD_LIBRARY_PATH environment variable to include the `/opt/SUNWnsun/bin` directory, as follows:

```
$ LD_LIBRARY_PATH=${LD_LIBRARY_PATH}:/opt/SUNWnsun/bin
$ export LD_LIBRARY_PATH
```

If the LD_LIBRARY_PATH environment variable setting does not include the `/opt/SUNWnsun/bin` directory, the following error message appears when you compile a program:

```
ld.so.1: cobol: fatal: liblicense.so: can't open file: errno=2
```

Pro*COBOL Oracle Runtime System

Oracle provides its own complete runtime system, called `rtsora`, to run dynamically loadable Pro*COBOL programs. Use the `rtsora` runtime system in place of the `cobrun` runtime system provided by Merant to run dynamically loadable Pro*COBOL programs. If you attempt to run a Pro*COBOL program with `cobrun`, you 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
```

Pro*COBOL Demonstration Programs

Demonstration programs are provided to show the features of the Pro*COBOL precompiler. The demonstration programs are located in either the `$ORACLE_HOME/precomp/demo/procob` directory or the `$ORACLE_HOME/precomp/demo/procob2` directory, depending on the Pro*COBOL version. By default, all programs are dynamically linked with the client shared library.

The programs assume that the demonstration tables created by the `$ORACLE_HOME/sqlplus/demo/demobld.sql` script exist in the SCOTT schema with the password TIGER.

Use the appropriate make file to create the demonstration programs:

- For Pro*COBOL release 9.2.0.1.0, use:

  `$ORACLE_HOME/precomp/demo/procob/demo_procob.mk`

- For Pro*COBOL release 1.8.77.0.0, use:

  `$ORACLE_HOME/precomp/demo/procob/demo_procob18.mk`

> **Note:** The following examples assume that you are using the make file for Pro*COBOL release 9.2.0.1.0.

To precompile, compile, and link the `sample1` demonstration program for Pro*COBOL, enter:

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

To create all of the Pro*COBOL demonstration programs, enter:

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

To create and run a dynamically loadable `sample1.gnt` program to be used with the `rtsora` runtime system, enter:

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

Some demonstration programs require you to run a SQL script, located in the `$ORACLE_HOME/precomp/demo/sql` directory. If you do not run the script, a message displays requesting you to run it. To build a demonstration program and run the corresponding SQL script, include the `make` macro argument `RUNSQL=run` on the command line. For example, to create the `sample9` demonstration program and run the required `$ORACLE_HOME/precomp/demo/sql/sample9.sql` script, enter:

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

To create all of the Pro*COBOL demonstration programs and run all corresponding required SQL scripts, enter:

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

Pro*COBOL User Programs

You can use the appropriate demonstration make file to create user programs, depending on the version of Pro*COBOL you are using. The syntax for creating a program with the demonstration make file is:

- For Pro*COBOL release 9.2.0.1.0, enter:

```
$ make -f demo_procob.mk target COBS="cobfile1 cobile2 ..." EXE=exename
```

- For Pro*COBOL release 1.8.77.0.0, enter:

```
$ make -f demo_procob18.mk target COBS="cobfile1 cobfile2 ..." EXE=exename
```

In the preceding examples:

- *target* is the make file target that you want to use
- *cobfilen* is the COBOL source file for the program
- *exename* is the executable program

For example, to create the program `myprog`, enter one of the following commands, depending on the source and type of executable you want to create:

- For COBOL source, dynamically linked with the client shared library, enter:

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

- For COBOL source, statically linked, enter:

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

- For COBOL source, dynamically loadable for use with `rtsora`, enter:

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

FORMAT Precompiler Option

The `FORMAT` precompiler option specifies the format of input lines for COBOL. If you specify the default `FORMAT=ANSI`, 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 left-most column.

Pro*FORTRAN Precompiler (AIX, HP, Solaris, and Tru64 Only)

Before you use the Pro*FORTRAN precompiler, verify that the correct version of the compiler is installed.

> **See Also:** *Oracle9i Installation Guide Release 2 (9.2.0.1.0) for UNIX Systems* for information on the required compiler versions on each platform, and the *Pro*FORTRAN Precompiler Programmer's Guide* for additional information on the Pro*FORTRAN precompiler and interface features.

Pro*FORTRAN Demonstration Programs

Demonstration programs are provided to show the features of the Pro*FORTRAN precompiler. All of the demonstration programs are located in the `$ORACLE_HOME/precomp/demo/profor` directory. By default, all programs are dynamically linked with the client shared library.

The programs assume that the demonstration tables created by the `$ORACLE_HOME/sqlplus/demo/demobld.sql` script exist in the SCOTT schema with the password TIGER.

Use the `demo_profor.mk` make file, located in the `$ORACLE_HOME/precomp/demo/profor` directory, to create the demonstration programs. For example, to precompile, compile, and link the `sample1` demonstration program, enter:

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

To create all of the Pro*FORTRAN demonstration programs, enter:

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

Some demonstration programs require you to run a SQL script, located in the `$ORACLE_HOME/precomp/demo/sql` directory. If you do not run the script, a message displays requesting you to run it. To build a demonstration program and run the corresponding SQL script, include the `make` macro argument `RUNSQL=run` on the command line. For example, to create the `sample11` demonstration program and run the required `$ORACLE_HOME/precomp/demo/sql/sample11.sql` script, enter:

```
$ make -f demo_profor.mk sample11 RUNSQL=run
```

To create all of the Pro*FORTRAN demonstration programs and run all corresponding required SQL scripts, enter:

```
$ make -f demo_profor.mk samples RUNSQL=run
```

Pro*FORTRAN User Programs

You can use the $ORACLE_HOME/precomp/demo/profor/demo_profor.mk make file to create user programs. The syntax for creating a program with the demo_proc.mk make file is:

```
$ make -f demo_profor.mk target FORS="forfile1 forfile2 ..." EXE=exename
```

In the preceding example:

- *target* is the make file target that you want to use
- *forfilen* is the FORTRAN source for the program
- *exename* is the executable program

For example, to create the program myprog, from the Pro*FORTRAN source file myprog.pfo, enter one of the following commands, depending on the type of executable that you want to create:

- For an executable dynamically linked with the client shared library, enter:

  ```
  $ make -f demo_profor.mk build FORS=myprog.f EXE=myprog
  ```

- For an executable statically linked, enter:

  ```
  $ make -f demo_profor.mk build_static FORS=myprog.f EXE=myprog
  ```

SQL*Module for Ada (Solaris 32-Bit and AIX Only)

Before using SQL*Module for Ada, verify that the correct version of the compiler is installed.

> **See Also:** O*racle9i Installation Guide Release 2 (9.2.0.1.0) for UNIX Systems* for information on the required compiler versions on each platform, and the *SQL*Module for Ada Programmer's Guide* for additional information on SQL*Module for Ada.

SQL*Module for Ada Demonstration Programs

Demonstration programs are provided to show the features of SQL*Module for Ada. All of the demonstration programs are located in the `$ORACLE_HOME/precomp/demo/modada` directory. By default, all programs are dynamically linked with the client shared library.

The `ch1_drv` demonstration program assumes that the demonstration tables created by the `$ORACLE_HOME/sqlplus/demo/demobld.sql` script exist in the SCOTT schema with the password TIGER.

The `demcalsp` and `demohost` demonstration programs require that the sample college database exists in the MODTEST schema. You can use the appropriate `make` command to create the MODTEST schema and load the sample college database.

To create all of the SQL*Module for Ada demonstration programs, run the necessary SQL scripts to create the MODTEST user, and create the sample college database, enter:

```
$ make -f demo_modada.mk all RUNSQL=run
```

To create a single demonstration program (`demohost`), and run the necessary SQL scripts to create the MODTEST user, and create the sample college database, enter:

```
$ make -f demo_modada.mk makeuser loaddb demohost RUNSQL=run
```

To create all of the SQL*Module for Ada demonstration programs, without recreating the sample college database, enter:

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

To create a single demonstration program (`demohost`), without recreating the sample college database, enter:

```
$ make -f demo_modada.mk demohost
```

All programs assume that an Oracle Net connect string or alias named INST1_ALIAS is defined and is capable of connecting to the database where the appropriate tables exist.

SQL*Module for Ada User Programs

You can use the $ORACLE_HOME/precomp/demo/modada/demo_modada.mk make file to create user programs. The syntax for creating a user program with the demo_modada.mk make file is:

```
$ make -f demo_modada.mk ada OBJS="module1 module2 ..." \
EXE=exename MODARGS=SQL*Module_arguments
```

In the preceding example:

- *modulen* is a compiled Ada object
- *exename* is the executable program
- *SQL*Module_arguments* are the command-line arguments to be passed to the SQL*Module

> **See Also:** *SQL*Module for Ada Programmers Guide* for more information on SQL*Module for Ada user programs.

Oracle Call Interface

Before you use the Oracle Call Interface (OCI), verify that the correct version of Pro*C/C++ is installed.

> **See Also:** O*racle9i Installation Guide Release 2 (9.2.0.1.0) for UNIX Systems* for more information on the required version of Pro*C/C++ for your operating system, and the *Oracle Call Interface Programmer's Guide* for additional information on the Oracle Call Interface.

OCI Demonstration Programs

Demonstration programs are provided that show the features of the OCI. There are two types of demonstration programs: C and C++. All of the demonstration programs are located in the $ORACLE_HOME/rdbms/demo directory. By default, all programs are dynamically linked with the client shared library.

Many of the demonstration programs assume that the demonstration tables created by the $ORACLE_HOME/sqlplus/demo/demobld.sql script exist in the SCOTT schema with the password TIGER.

Use the demo_rdbms.mk make file, located in the $ORACLE_HOME/rdbms/demo directory, 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
```

To create all of the C demonstration programs for OCI, enter:

```
$ make -f demo_rdbms.mk demos
```

To create all of the C++ demonstration programs for OCI, enter:

```
$ make -f demo_rdbms.mk c++demos
```

OCI User Programs

You can use the $ORACLE_HOME/rdbms/demo/demo_rdbms.mk make file to create user programs. The syntax for creating a user program with the demo_rdbms.mk make file is:

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

In the preceding example:

- *target* is the make file target that you want to use
- *objfilen* is the object file to link the program
- *exename* is the executable program

For example, to create the myprog program from the C/C++ source myprog.c, enter one of the following commands, depending on the type of executable you want to create:

- For C source, dynamically linked with the client shared library, enter:

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

- For C source, statically linked, enter:

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

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

  ```
  $ make -f demo_rdbms.mk buildc++ OBJS=myprog.o EXE=myprog
  ```

- For C++ source, statically linked, enter:

  ```
  $ make -f demo_rdbms.mk buildc++_static OBJS=myprog.o EXE=myprog
  ```

Note: In the preceding examples, the file `myprog.o` is the object file generated by the compiler.

On AIX, HP, and Solaris 64-bit, the `demo_rdbms.mk` make file builds 64-bit user programs by default. You can use the `demo_rdbms32.mk` make file to build 32-bit user programs. See the make file for more information on creating 32-bit user programs.

Custom Make Files

Oracle Corporation recommends that you use the provided `demo_`*`product`*`.mk` make files to create user programs as described in the specific product sections of this chapter. If you modify the provided make file, or if you choose to use a custom-written make file, the following restrictions apply:

- Do not modify the order of the Oracle libraries. Oracle libraries are included on the link line more than once so that all of the symbols are resolved during linking.

  The order of the Oracle libraries is essential for the following reasons:

  – Oracle libraries are mutually referential. Functions in library A call functions in library B, and functions in library B call functions in library A.

  – The HP and Tru64 linkers are one-pass linkers. The AIX, Linux, and Solaris linkers are two-pass linkers.

- If you add your own library to the link line, add it to the beginning or to the end of the link line. Do not place user libraries between the Oracle libraries.
- If you choose to use a `make` utility such as `nmake` or GNU `make`, be aware of how macro and suffix processing differs from the make utility provided with the platform. Oracle make files are tested and are supported with the `make` utility for your platform.
- Oracle library names and the contents of Oracle libraries are subject to change between releases. Always use the `demo_`*`product`*`.mk` make file that ships with the current release as a guide to determine the required libraries.

Correcting Undefined Symbols (Solaris Only)

Oracle provides the `symfind` utility to assist you in locating a library or object file where a symbol is defined. When linking a program, undefined symbols are a common error that produce an error message 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
```

The error occurs when the linker cannot find a definition for a referenced symbol. If this error message occurs, 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.

The following example shows the output from the `symfind` utility, used to locate the `sqlcex` symbol:

```
$ symfind sqlcex

SymFind - Find Symbol <sqlcex> in <**>.a, .o, .so
-----------------------------------------------------
Command:         /u01/app/oracle/product/9.2.0.1.0/bin/symfind sqlcex
Local Directory: /u01/app/oracle/product/9.2.0.1.0
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
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^ ./lib/libclntsh.sl
[35]    |         0|      44|FUNC |GLOB |0    |5      |sqlcex
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^ ./lib/libsql.a
```

Multi-threaded Applications

The Oracle libraries provided with this release are thread safe, allowing support for multi-threaded applications.

Using Signal Handlers

Oracle9*i* uses signals for two-task communication. Signals are installed in a user process when you connect to the database and are removed when you disconnect.

Table 4–5 describes the signals that Oracle9*i* uses for two-task communication.

Table 4–5 Signals for Two-Task Communication

Signal	Description
SIGCLD	The pipe driver uses SIGCLD, also referred to as SIGCHLD, when an Oracle process dies. The UNIX kernel sends a SIGCLD signal to the user process. The signal handler uses the wait() routine to determine whether a server process died. The Oracle process does not catch SIGCLD; the user process catches it.
SIGCONT	The pipe two-task driver uses SIGCONT to send out-of-band breaks from the user process to the Oracle process.
SIGINT	Two-task drivers use SIGINT to detect user interrupt requests. The Oracle process does not catch SIGINT; the user process catches it.
SIGIO	Oracle Net protocols use SIGIO to indicate incoming networking events.
SIGPIPE	The pipe driver uses SIGPIPE 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. Both the Oracle process and the user process catch SIGPIPE. SIGCLD is similar to SIGPIPE, but only applies to user processes, not to Oracle processes.
SIGTERM	The pipe driver uses SIGTERM to signal interrupts from the user to the Oracle process. This occurs when the user presses the interrupt key, Ctrl+c. The user process does not catch SIGTERM; the Oracle process catches it.
SIGURG	Oracle Net TCP/IP drivers use SIGURG to send out-of-band breaks from the user process to the Oracle process.

The listed signals affect all precompiler applications. You can install one signal handler for SIGCLD (or SIGCHLD) and SIGPIPE when connected to the Oracle process. If you call the osnsui() routine to set it up, you can have more than one signal handle for SIGINT. For SIGINT, use osnsui() and osncui() to register and delete signal-catching routines.

You can also 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.

Example 4–1 shows how to set up a signal routine and a catching routine.

Example 4–1 Signal Routine and Catching Routine

```
/* 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 *handlp 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.
*/
```

Example 4–2 shows how to use the osnsui() and the osncui() routines in an application program.

Example 4–2 osnsui() and osncui() Routine Template

```
/*
** User interrupt handler template.
*/
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
                   Take appropriate action for the error. */
                ...
        }

        ...

        /* clear my interrupt handler */

        if (err = osncui(handle))
        {
                /* if the return value is non-zero, an error has occurred
                   Take appropriate action for the error. */
                ...
        }
        ...
}
```

XA Functionality

Oracle XA is the Oracle implementation of the X/Open Distributed Transaction Processing (DTP) XA interface. The XA standard specifies a bi-directional interface between resource managers (for example, Oracle) that provide access to shared resources within transactions, and between a transaction service that monitors and resolves transactions.

Oracle Call Interface has XA functionality. When building a TP-monitor XA application, ensure that the TP-monitor libraries (that define the symbols ax_reg and ax_unreg) are placed in the link line before the Oracle client shared library. This link restriction is required only when using the XA dynamic registration (Oracle XA switch `xaoswd`).

Oracle9*i* does not support Oracle7 release 7.1.6 XA calls (although it does support Oracle7 release 7.3 XA calls). Therefore, TP-monitor XA applications using Oracle7 release 7.1.6 XA calls must be relinked with the Oracle9*i* XA library. The Oracle9*i* XA calls are defined in both the `$ORACLE_HOME/lib/libclntsh.sl` shared library and the `$ORACLE_HOME/lib/libclient9.a` static library.

5

Configuring Oracle Net Services

This chapter decribes how to configure Oracle Net Services on AIX, HP, Linux, Solaris, and Tru64 systems. It contains the following sections:

- Location of Oracle Net Services Configuration Files
- Adapters Utility
- Oracle Protocol Support
- Setting Up the Listener for TCP/IP or TCP/IP with SSL
- Oracle Enterprise Manager
- Configuring Oracle Intelligent Agent for Oracle SNMP
- Oracle Advanced Security
- Calling 32-Bit External Procedures from PL/SQL (AIX, HP, and Solaris 64-Bit Only)

> **See Also:** *Oracle9i Net Services Administrator's Guide* for more information on Oracle networking.

Location of Oracle Net Services Configuration Files

Oracle Net Services configuration files are typically, but not always, located in the `$ORACLE_HOME/network/admin` directory. Depending on the file, Oracle Net uses a different search order to locate the file.

The search order for the `sqlnet.ora` and `ldap.ora` files is as follows:

1. The directory specified by the TNS_ADMIN environment variable, if set
2. The `$ORACLE_HOME/network/admin` directory

The search order for the `cman.ora`, `listener.ora`, and `tnsnames.ora` files is as follows:

1. The directory specified by the TNS_ADMIN environment variable, if set
2. One of the following directories:
   - For Solaris systems, the`/var/opt/oracle` directory
   - For AIX, HP, Linux, and Tru64, the `/etc` directory
3. The `$ORACLE_HOME/network/admin` directory

For some system-level configuration files, users may have a corresponding user-level configuration file (stored in the user's home directory). The settings in the user-level file override the settings in the system-level file. The following table lists the system-level configuration files and the corresponding user-level configuration files:

System-Level Configuration File	User-Level Configuration File
`sqlnet.ora`	`$HOME/.sqlnet.ora`
`tnsnames.ora`	`$HOME/.tnsnames.ora`

Sample Configuration Files

The `$ORACLE_HOME/network/admin/samples` directory contains samples of the `cman.ora`, `listener.ora`, `names.ora`, `sqlnet.ora`, and `tnsnames.ora` configuration files.

Adapters Utility

Use the `adapters` utility to display the transport protocols, naming methods, and Oracle Advanced Security options that Oracle9*i* supports on your system. To use the `adapters` utility, enter the following commands:

```
$ cd $ORACLE_HOME/bin
$ adapters ./oracle
```

The `adapters` utility displays output similar to the following:

```
Oracle Net transport protocols linked with ./oracle are

         IPC
         BEQ
         TCP/IP
         SSL
         RAW

Oracle Net naming methods linked with ./oracle are:

         Local Naming (tnsnames.ora)
         Oracle Directory Naming
         Oracle Host Naming
         Oracle Names Server Naming
         NIS Naming

Oracle Advanced Security options linked with ./oracle are:

         RC4 40-bit encryption
         RC4 128-bit encryption
         RC4 256-bit encryption
         DES40 40-bit encryption
         DES 56-bit encryption
         3DES 112-bit encryption
         3DES 168-bit encryption
         AES 128-bit encryption
         AES 192-bit encryption
         SHA crypto-checksumming (for FIPS)
         SHA-1 crypto-checksumming
         Kerberos v5 authentication
         CyberSAFE authentication
         RADIUS authentication
         ENTRUST authentication
```

On the client, run the `adapters` utility to display the configured Oracle transport protocols, naming methods, and security options on the system. To run the `adapters` utility on the client:

```
$ cd $ORACLE_HOME/bin
$ adapters
```

The `adapters` utility displays output similar to the following:

```
Installed Oracle Net transport protocols are:

        IPC
        BEQ
        TCP/IP
        SSL
        RAW

Installed Oracle Net naming methods are:

        Local Naming (tnsnames.ora)
        Oracle Directory Naming
        Oracle Host Naming
        Oracle Names Server Naming
        NIS Naming

Installed Oracle Advanced Security options are:

        RC4 40-bit encryption
        RC4 56-bit encryption
        RC4 128-bit encryption
        RC4 256-bit encryption
        DES40 40-bit encryption
        DES 56-bit encryption
        3DES 112-bit encryption
        3DES 168-bit encryption
        AES 128-bit encryption
        AES 192-bit encryption
        AES 256-bit encryption
        MD5 crypto-checksumming
        SHA-1 crypto-checksumming
        Kerberos v5 authentication
        CyberSAFE authentication
        RADIUS authentication
```

> **See Also:** *Oracle9i Net Services Administrator's Guide* for more information on the `adapters` utility.

Oracle Protocol Support

Oracle protocol support is a component of Oracle Net. It includes the following:

- IPC protocol support
- TCP/IP protocol support
- TCP/IP with SSL protocol support

The IPC, TCP/IP, and TCP/IP with SSL protocol supports each have an address specification that is used in Oracle Net Services configuration files and in the DISPATCHER initialization parameter in the `init`*`sid`*`.ora` file. The following sections describe the address specifications for each of the protocol supports.

> **See Also:** *Oracle9i Net Services Administrator's Guide* for more information on Oracle protocol support.

IPC Protocol Support

The IPC protocol support can be used only when the client program and the Oracle9*i* server are installed on the same system. This protocol support requires a listener for its operation. It is installed and linked to all client tools and to Oracle9*i*.

The IPC protocol support requires an address specification in the following format:

```
(ADDRESS =
     (PROTOCOL=IPC)
     (KEY=key)
)
```

The following table describes the parameters used in this address specification:

Parameter	Description
PROTOCOL	The protocol to be used. The value is IPC. It is not case sensitive.
KEY	Any name unique from any other name used for an IPC KEY on the same system.

The following shows a sample IPC protocol address:

```
(ADDRESS=
     (PROTOCOL=IPC)
     (KEY=PROD)
)
```

TCP/IP Protocol Support

TCP/IP is the standard communication protocol used for client/server communication over a network. The TCP/IP protocol support enables communication between client programs and the Oracle9*i* server, whether they are installed on the same or different systems. If the TCP/IP protocol is installed on your system, the TCP/IP protocol support is installed and linked to all client tools and to Oracle9*i*.

The TCP/IP protocol support requires an address specification in the following format:

```
(ADDRESS =
     (PROTOCOL=TCP)
     (HOST=hostname)
     (PORT=port)
)
```

The following table describes the parameters used in this address specification:

Parameter	Description
PROTOCOL	The protocol support to be used. The value is TCP. It is not case sensitive.
HOST	The host name or the host IP address.
PORT	The TCP/IP port. Specify the port as either a number or the alias name mapped to the port in the `/etc/services` file. Oracle Corporation recommends a value of 1521.

The following shows a sample TCP/IP protocol address:

```
(ADDRESS=
     (PROTOCOL=TCP)
     (HOST=MADRID)
     (PORT=1521)
)
```

TCP/IP with SSL Protocol Support

The TCP/IP with SSL protocol support enables an Oracle application on a client to communicate with remote Oracle databases through TCP/IP and SSL. To use TCP/IP with SSL, Oracle Advanced Security must be installed.

The TCP/IP with SSL protocol support requires an address specification in the following format:

```
(ADDRESS =
     (PROTOCOL=TCPS)
     (HOST=hostname)
     (PORT=port)
)
```

The following table describes the parameters used in this address specification:

Parameter	Description
PROTOCOL	The protocol to be used. The value is TCPS. It is not case sensitive.
HOST	The host name or the host IP address.
PORT	The TCP/IP with SSL port. Specify the port as either a number or the alias name mapped to the port in the `/etc/services` file. Oracle Corporation recommends a value of 2484.

The following shows a sample TCP/IP with SSL protocol address:

```
(ADDRESS=
     (PROTOCOL=TCPS)
     (HOST=MADRID)
     (PORT=2484)
)
```

Setting Up the Listener for TCP/IP or TCP/IP with SSL

Oracle Corporation recommends that you reserve a port for the listener in the `/etc/services` file of each Oracle Net Services node on the network. The default port is 1521. The entry lists the listener name and the port number, for example:

```
oraclelistener    1521/tcp
```

In this example *oraclelistener* is the name of the listener, as defined in the `listener.ora` file. Reserve more than one port if you intend to start more than one listener.

If you use intend to use SSL, you should define a port for TCP/IP with SSL in the `/etc/services` file. Oracle Corporation recommends a value of 2484. For example:

```
oraclelistenerssl      2484/tcps
```

In this example *oraclelistenerssl* is the name of the listener, as defined in the `listener.ora` file. Reserve more than one port if you intend to start more than one listener.

Oracle Enterprise Manager

Use the `oratclsh` executable to debug your Tcl scripts. Before executing `oratclsh`, set the TCL_LIBRARY environment variable to specify the `$ORACLE_HOME/network/agent/tcl` directory.

> **See Also:** *Intelligent Agent User's Guide* for more information on debugging Tcl scripts.

Configuring Oracle Intelligent Agent for Oracle SNMP

Although Oracle Intelligent Agent does not require Simple Network Management Protocol (SNMP) to work, you can configure Oracle SNMP support before starting the Intelligent Agent. Note that all of the configuration files for the following steps are located in the `$ORACLE_HOME/network/snmp/peer` directory.

Configure the Master Agent

In the `CONFIG.master` file, make the following change:

1. Search for the line beginning with MANAGER.
2. Change the value of the MANAGER parameter to the IP address or hostname of the system where you want SNMP trap messages sent.

   You can also make other changes to the `CONFIG.master` file as documented within the file.

3. On AIX systems only, add the following line to the `/etc/snmpd.conf` file, where *`ip_address`* is the IP address of the Oracle subagent:

   ```
   smux 0.0 " " ip_address
   ```

Configure the Encapsulator

To configure the encapsulator, perform the following steps:

1. Add the following line to the `snmpd.conf` file, where *`hostname_or_IP_address`* represents the local system IP address or host name:

   ```
   trap hostname_or_IP_address
   ```

2. On AIX systems only, modify the port numbers specified for the NEW_SNMPD_PORT and NEW_TRAPD_PORT variables in the `start_peer` script, if necessary.

   The port number specified for the NEW_SNMPD_PORT variable must be different to the port number specified for the NEW_TRAPD_PORT variable.

3. On all platforms except AIX, complete the following steps:

   a. If necessary, modify the port number specified in the `CONFIG.encap` file. The default port number is 1161.

   b. If you modified the port number in step a, change the value specified for the NEW_SNMPD_PORT variable in the `start_peer` script to this port number.

   c. If necessary, modify the value specified for the NEW_TRAPD_PORT variable.

      This variable specifies the PEER encapsulator port to which the `snmpd` agent sends traps. This port number must be different to the port number specified for the NEW_SNMPD_PORT variable.

Verify the Location of the SNMP Daemon in the start_peer Script

The `start_peer` script contains a line similar to the following, where *`snmpd_executable_path`* is the path of the `snmpd` executable:

```
SNMPD=snmpd_executable_path
```

Make sure that *snmpd_executable_path* is the location of the snmpd executable on your system.

Start the SNMP Components

To start the SNMP components, perform the following steps:

1. Enter the following commands to verify that the SNMP components (master_peer, encap_peer, and snmpd) are *not* running:

   ```
   $ ps -aef | grep peer
   $ ps -aef | grep snmp
   ```

   If any of the components are running, log in as the root user and use the kill command to terminate the processes before proceeding.

2. On AIX systems only, enter the following command as the root user to start the native AIX SNMP agent:

   ```
   # startsrc -s snmpd "-f /tmp/snmpd.log"
   ```

   This command starts the SNMP daemon and logs information in the /tmp/snmpd.log file.

3. On all platforms except AIX, complete the following steps:

   a. As the root user, run the start_peer script to start the PEER master agent, PEER encapsulator, and native UNIX SNMP agent:

      ```
      # cd $ORACLE_HOME/network/snmp/peer
      # ./start_peer -a
      ```

      Note: If you do not have the native UNIX SNMP agent on your system, yo must *not* use the PEER encapsulator. To start the master agent only, enter start_peer -m.

   b. Enter the following commands to verify that the SNMP components are running:

      ```
      # ps -aef | grep peer
      # ps -aef | grep snmp
      ```

Configure and Start the Database Subagent

For information on configuring and starting the database subagent (the Oracle Intelligent Agent), see the *Oracle Enterprise Manager Configuration Guide*.

Oracle Advanced Security

When you install Oracle Advanced Security, three `.bak` files are created: `naeet.o.bak`, `naect.o.bak`, and `naedhs.o.bak`. These files are located in the `$ORACLE_HOME/lib` directory. They are required for relinking during deinstallation of Oracle Advanced Security. Do not delete them.

Calling 32-Bit External Procedures from PL/SQL (AIX, HP, and Solaris 64-Bit Only)

The 64-bit Extproc executable (`extproc`) and the 32-bit Extproc executable (`extproc32`) are installed in the `$ORACLE_HOME/bin` directory. By default, the `extproc` executable is enabled to run 64-bit external procedures. To enable 32-bit external procedures:

1. Set the value of the PROGRAM parameter in the `listener.ora` file:

   ```
   (PROGRAM=extproc32)
   ```

2. Include the `$ORACLE_HOME/lib32` directory in one of the following environment variables, depending on your platform:

Platform	Environment Variable
AIX	LIBPATH
HP	SHLIB_PATH
Solaris 64-bit	LD_LIBRARY_PATH

3. Shut down and restart the listener.

> **Note:** You can configure the listener to run either 32-bit or 64-bit external procedures, but not both at the same time.

A

Tuning for Oracle9*i* on AIX

This appendix contains the following tuning information for Oracle9*i* on AIX:

- Memory and Paging
- Disk I/O Issues
- CPU Scheduling and Process Priorities
- Resilvering with Oracle9i

Memory and Paging

Memory contention occurs when processes require more memory than is available. To cope with the shortage, the system pages programs and data between memory and disks.

Controlling Buffer-Cache Paging Activity

Excessive paging activity decreases performance substantially. This can become a problem with database files created on journaled file systems (JFS and JFS2). In this situation, a large number of SGA data buffers might also have analogous journaled file system buffers containing the most frequently referenced data. The behavior of the AIX file buffer cache manager can have a significant impact on performance. It can cause an I/O bottleneck, resulting in lower overall system throughput.

On AIX, tuning buffer-cache paging activity is possible but it must be done carefully and infrequently. Use the `vmstat` command to tune the following AIX system parameters:

Parameter	Description
MINFREE	The minimum free-list size. If the free-list space in the buffer falls below this size, the system uses page stealing to replenish the free list.
MAXFREE	The maximum free-list size. If the free-list space in the buffer exceeds this size, the system stops using page stealing to replenish the free list.
MINPERM	The minimum number of permanent buffer pages for file I/O.
MAXPERM	The maximum number of permanent buffer pages for file I/O.

> **See Also:** *AIX Performance Tuning Guide, Version 3.2 and 4* or the *AIX5L Performance Management Guide* for more information on AIX system parameters.

Log on as the `root` user and use the `vmtune` command to change these limits. The AIX `vmtune` command is operating-system version specific. If you run the `vmtune` command from one release on a different AIX release, the operating system might fail.

Tuning the MINFREE and MAXFREE Parameters

You can base the size of the MINFREE parameter on the working set size of the programs that require a quick response. You must have enough pages in the free list so that the loaded program does not need to replenish the free list. You can determine the size of the working set for a program by using the `svmon -p` command. The value of the MAXFREE parameter should be greater than the value of the MINFREE parameter by the value of the MAXPGAHEAD parameter, or 8 pages, which ever is higher. For instance, if you determine that the MINFREE parameter value should be 128 pages and the MAXPGAHEAD parameter value is 16 pages, enter the following command to set the value of the MINFREE parameter to 128 pages and the value of the MAXFREE parameter to 144 pages:

```
$ /usr/samples/kernel/vmtune -f 128 -F 144
```

Tuning the AIX File Buffer Cache

The purpose of the AIX file buffer cache is to reduce disk access frequency when journaled file systems are used. If this cache is too small, disk usage increases and potentially saturates one or more disks. If the cache is too large, memory is wasted.

> **See Also:** "Controlling Buffer-Cache Paging Activity" on page A-2 for information on the implications of increasing the AIX file buffer cache.

You can configure the AIX file buffer cache by adjusting the MINPERM and MAXPERM parameters. In general, if the buffer hit ratio is low (less than 90 percent), as determined by the `sar -b` command, increasing the MINPERM parameter value might help. If maintaining a high buffer hit ratio is not critical, decreasing the MINPERM parameter value increases the physical memory available. Refer to your AIX documentation for more information on increasing the size of the AIX file buffer cache.

The performance gain cannot be quantified easily, because it depends on the degree of multiprogramming and the I/O characteristics of the workload.

Tuning the MINPERM and MAXPERM Parameters

AIX provides a mechanism for you to loosely control the ratio of page frames used for files versus those used for computational (working or program text) segments by adjusting the MINPERM and MAXPERM values according to the following guidelines:

- If the percentage of real memory occupied by file pages falls below the MINPERM value, the page-replacement algorithm steals both file and computational pages, regardless of repage rates.
- If the percentage of real memory occupied by file pages rises above the MAXPERM value, the page-replacement algorithm steals both file and computational pages.
- If the percentage of real memory occupied by file pages is between the MINPERM and MAXPERM parameter values, the virtual memory manager (VMM) normally steals only file pages, but if the repaging rate for file pages is higher then the repaging rate for computational pages, the computational pages are stolen as well.

Use the following algorithm to calculate the default values:

- MINPERM (in pages) = ((number of page frames)-1024) * 0.2
- MAXPERM (in pages) = ((number of page frames)-1024) * 0.8

Use the following command to change the value of the MINPERM parameter to 5 percent of the total number of page frames, and the value of the MAXPERM parameter to 20 percent of the total number of page frames:

```
# vmtune -p 5 -P 20
```

The default values are 20 percent and 80 percent, respectively.

If the database files are on raw devices, you can set the MINPERM and MAXPERM parameters to low values, for example 5 percent and 20 percent, respectively. This is because the AIX file buffer cache is not used for raw devices. The memory might be better used for other purposes, such as the Oracle System Global Area.

Allocating Sufficient Paging Space

Inadequate paging space usually causes the system to hang or suffer abnormally slow response times. On AIX, you can dynamically add paging space on raw disk partitions. The amount of paging space you should configure depends on the amount of physical memory present and the paging space requirements of your applications. Use the `lsps` command to monitor paging space use and the `vmstat` command to monitor system paging activities. To increase the paging space, use the `smit pgsp` command.

Prior to AIX 4.3.2, paging space needed to be large, typically two to three times the size of real memory. This is because page space was pre-allocated when a program started, whether it used the page space or not. AIX 4.3.2 and higher use deferred paging, where paging space is not allocated until needed. The system uses swap space only if it runs out of real memory. If the memory is sized correctly, there is no paging and the page space can be small. Workloads where the demand for pages does not fluctuate significantly perform well with a small paging space. Workloads likely to have peak periods of increased paging require enough paging space to handle the peak number of pages.

Controlling Paging

Constant and excessive paging indicates that the real memory is over-committed. In general, you should:

- Avoid constant paging unless the system is equipped with very fast expanded storage that makes paging between memory and expanded storage much faster than Oracle can read and write data between the SGA and disks.
- Allocate limited memory resources to where it is most beneficial to the system's performance. It is sometimes a recursive process of balancing the memory resource requirements and trade-offs.
- If memory is not adequate, build a prioritized list of memory-requiring processes and elements of the system. Assign memory to where the performance gains are the greatest. A prioritized list might look like:
  1. OS and RDBMS kernels
  2. User and application processes
  3. Redo log buffer
  4. PGAs and shared pool
  5. Database block buffer caches

For instance, if you query Oracle dynamic performance tables and views and find that both the shared pool and database buffer cache require more memory, assigning the limited spare memory to the shared pool might be more beneficial than assigning it to the database block buffer caches**.**

The following AIX commands provide paging status and statistics:

- `vmstat -s`
- `vmstat` *`interval`* [*`repeats`*]
- `sar -r` *`interval`* [*`repeats`*]

Setting the Database Block Size

You can configure the Oracle database block size for better I/O throughput. On AIX, you can set the value of the DB_BLOCK_SIZE initialization parameter to between 2 KB to 32 KB, with a default of 4 KB. If the Oracle database is installed on a journaled file system, then the block size should be a multiple of the file system blocksize (4 KB on AIX). For databases on raw partitions, the Oracle database block size is a multiple of the operating system physical block size (512 bytes on AIX).

Oracle Corporation recommends smaller Oracle database block sizes (2 KB or 4 KB) for online transaction processing (OLTP) or mixed workload environments and larger block sizes (8 KB, 16 KB, or 32 KB) for decision support system (DSS) workload environments.

Tuning the Log Archive Buffers

You can improve the speed of archiving the database by dedicating larger buffers. However, over-configuring can degrade overall system performance. You can increase the LOG_ARCHIVE_BUFFER_SIZE initialization parameter to 128 KB. The default value for the LOG_ARCHIVE_BUFFER_SIZE parameter is 4 KB.

Tune this parameter carefully so that overall performance of normal database activity does not degrade drastically. Performance improvements of 0 to 20 percent in the archiving process are possible. Some users report even larger improvements.

I/O Buffers and SQL*Loader

For high-speed data loading, such as using the SQL*Loader direct path option in addition to loading data in parallel, the CPU spends most of its time waiting for I/O to complete. By increasing the number of buffers, you can usually push the CPU usage harder, thereby increasing overall throughput.

The number of buffers (set by the SQL*Loader BUFFERS parameter) you choose depends on the amount of available memory and how hard you want to push CPU usage. See *Oracle9i Database Utilities* for information on adjusting the file processing options string for the BUFFERS parameter.

The performance gains depend on CPU usage and the degree of parallelism that you use when loading data.

> **See Also:** *Oracle9i Database Utilities* for more generic information on the SQL*Loader utility.

BUFFER Parameter for the Import Utility

The BUFFER parameter for the Import utility should be set to a large value to optimize the performance of high-speed networks when they are used. For instance, if you use the IBM RS/6000 Scalable POWERparallel Systems (SP) switch, you should use a value of at least 1 MB.

Disk I/O Issues

Disk I/O contention can result from poor memory management (with subsequent paging and swapping), or poor distribution of tablespaces and files across disks. Spread the I/O load evenly across all disks.

AIX Logical Volume Manager

The AIX Logical Volume Manager (LVM) can stripe data across multiple disks to reduce disk contention. The primary objective of striping is to achieve high performance when reading and writing large sequential files. Effective use of the striping features in the LVM allows you to spread I/O more evenly across disks, resulting in greater overall performance.

Design a Striped Logical Volume

When you define a striped logical volume, you must specify the following items:

Item	Recommended Settings
Drives	At least two physical drives. The drives should have minimal activity when performance-critical sequential I/O is executed. Sometimes you might need to stripe the logical volume between two or more adapters.
Stripe unit size	Although the stripe unit size can be any power of two from 2 KB to 128 KB, stripe sizes of 32 KB and 64 KB are good values for most workloads. For Oracle database files, the stripe size must be a multiple of the database block size.
Size	The number of physical partitions allocated to the logical volume must be a multiple of the number of disk drives used.
Attributes	Cannot be mirrored. Set the `copies` attribute to a value of `1`.

Suggested Striped Logical Volume Parameters

Use the `vmtune` command to adjust the MINPGAHEAD, MAXPGAHEAD, and MAXFREE parameters to achieve the highest sequential I/O throughput. See the *AIX Performance Tuning Guide, Version 3.2 and 4* or the *AIX5L Performance Management Guide* for the recommended settings.

Make sure that the I/O activity is evenly distributed across multiple disk drives by using AIX utilities such as `filemon`, `sar`, `iostat`, and other performance tools to identify any disks with high I/O activity.

Other Considerations

Performance gains from effective use of the LVM can vary greatly, depending on the LVM you use and the characteristics of the workload. For DSS workloads, you can see substantial improvement. For OLTP-type or mixed workloads, you can still expect significant performance gains.

Using Journaled File Systems Compared to Using Raw Partitions

Note the following considerations when you are deciding whether to use journaled file systems or raw partitions:

- File systems are continually being improved, as are various file system implementations. In some cases, file systems provide better I/O performance than raw devices.
- Different vendors implement the file system layer in different ways to exploit the strengths of different disks. This makes it difficult to compare file systems across platforms.
- The introduction of more powerful LVM interfaces substantially reduces the tasks of configuring and backing up logical disks based on raw partitions.

The degree of performance improvement largely depends on the I/O characteristics of the workload.

If you use a journaled file system, it is easier to manage and maintain database files than it is if you use raw devices. However, because JFS and JFS2 support only buffered read and write at present, every time data is transferred to or from the I/O subsystem (other than the Oracle buffer cache or SGA) extra AIX file buffer caches are created. This is the main drawback to using a journaled file system, and this penalty becomes even greater when the I/O subsystem is more heavily exercised.

Also, on heavily loaded systems, the contention for the AIX file buffer cache becomes the main bottleneck. Some side-effects include:

- Increased work for the `pagedaemon` command which flushes data from the AIX file buffer cache to disk
- More memory sacrificed to the AIX file buffer cache

Use raw devices/volumes for Oracle files only if your site has at least as many raw disk partitions as Oracle datafiles. If the raw disk partitions are already formatted, match datafile size to the partition size as closely as possible to avoid wasting space. In a heavily-used database configuration, use raw devices.

> **Note:** Oracle9*i* Real Application Clusters requires the use of raw devices instead of a journaled file system for database files.

Moving from a Journaled File System to Raw Devices

To move from a journaled file system to raw devices without the need to manually reload all of the data, enter the following command as the `root` user:

```
# dd if=old_JFS_file of=new_raw_device seek=1 bs=4k
```

Ensure that you follow these guidelines:

- Skip the first 4 KB block of the raw device because the AIX Logical Volume Manager uses it.

  Note: The first 4 KB block should not be skipped if the raw device is a Hashed Shared Disk (HSD).

- Size the raw device to prevent wasting space. The raw device should be a few blocks larger than the existing JFS or JFS2 file.
- Set the permissions on the raw partition.
- Rename the datafile.

You might experience a slight degradation in sequential read performance. If the application is performing many full table scans, you might experience a server performance degradation on these full table scans, because the AIX file buffer cache is no longer used to cache these tables.

In configurations where memory is scarce, you might need to resize the AIX file buffer cache.

In the AIX environment, Oracle Corporation encourages the use of symbolic links whenever possible, because the name of the raw device can change. For example, this change can occur when reconfiguring disk peripherals or moving the database to a new hardware configuration. Refer to the IBM documentation for more information.

See Also: O*racle9i Installation Guide Release 2 (9.2.0.1.0) for UNIX Systems* for more information on the operational issues involved in using raw devices.

Taking Advantage of Both Journaled File Systems and Raw Devices

On AIX, you can choose a journaled file system for some datafiles and raw partitions for other datafiles. Asynchronous I/O works with both raw partitions and a journaled file system. If you know the nature of I/O activity on database objects beforehand, you can place the datafiles corresponding to specific objects on either a journaled file system or raw partitions, together with an LVM, if available.

The improvement of performance varies greatly depending on the nature of your workload and the disk or file system configuration.

Using Asynchronous I/O

Oracle9*i* takes full advantage of asynchronous I/O (AIO) provided by AIX, resulting in faster database access. AIO interweaves multiple I/O to improve I/O subsystem throughput. The advantage of AIO is realized only when data is well distributed among different disks.

Using the LVM and striping enhances the effectiveness of AIO. The LVM reduces disk contention by striping data across multiple disk spindles. Using AIO with LVM significantly improves RDBMS performance.

AIX versions 4 and higher support asynchronous I/O (AIO) for database files created both on file system partitions and on raw devices. AIO on raw devices is implemented fully into the AIX kernel, and does not require server processes to service the AIO requests. When using AIO on file systems, the kernel server processes (`kproc`) control each request from the time a request is taken off the queue until it completes. The kernel server processes are also used with I/O with virtual shared disks (VSDs) and HSDs with FastPath disabled. By default, FastPath is enabled. The number of `kproc` servers determines the number of AIO requests that can be executed in the system concurrently, so it is important to tune the number of `kproc` processes when using filesystems to store Oracle9*i* datafiles.

Note: If you are using AIO with VSDs and HSDs with AIO FastPath enabled (the default), the maximum buddy buffer size must be greater than or equal to 128 KB.

Use one of the following commands to set the number of servers:

- `smit aio`
- `chdev -l aio0 -a maxservers='`*m*`' -a minservers='`*n*`'`

> **See Also:** System Management Interface Tool (SMIT) online help for information on SMIT, and the man pages for information on the `smit aio` and `chdev` commands.

Set the minimum value to the number of servers to be started at system boot. Set the maximum value to the number of servers that can be started in response to a large number of concurrent requests. These parameters apply to files only, they do not apply to raw devices.

The default value for the minimum number of servers is 1. The default value for the maximum number of servers is 10. These values are usually too low to run the Oracle server if you are not using kernelized AIO. Oracle Corporation recommends that you set the following values:

Parameter	Value
MINSERVERS	Varies depending on the asynchronous requests to the AIO servers on the system. Oracle Corporation recommends an initial value of 10.
MAXSERVERS	You can incrementally increase this value to a maximum of the value of the MAXSERVERS parameter divided by 2 or the number of disks accessed divided by 10, whichever is greater.
MAXREQS	8192

If the value of the MAXSERVERS parameter is set too low, you might see the following error messages repeated:

```
Warning: lio_listio returned EAGAIN
Performance degradation may be seen.
```

You can avoid these errors by increasing the value of the MAXSERVERS parameter to greater than the number of AIO servers running. To display the number of AIO servers running, enter the following command as the `root` user:

```
# pstat -a | grep -c aios
```

Check the number of active AIO servers periodically and change the values of the MINSERVERS and MAXSERVERS parameters if necessary. The changes take place when the system restarts.

I/O Slaves

I/O Slaves are specialized processes that perform only I/O. They can operate whether or not asynchronous I/O is available. They are allocated from shared memory buffers. I/O Slaves use a set of initialization parameters, listed in the following table, that allow a degree of control over the way they operate.

Parameter	Range of Values	Default Value
DISK_ASYNCH_IO	TRUE/FALSE	TRUE
TAPE_ASYNCH_IO	TRUE/FALSE	TRUE
BACKUP_TAPE_IO_SLAVES	TRUE/FALSE	FALSE
DBWR_IO_SLAVES	0 - 999	0
DB_WRITER_PROCESSES	1-20	1

There are times when the use of asynchronous I/O is not desirable or not possible. The first two parameters in the preceding table, DISK_ASYNCH_IO and TAPE_ASYNCH_IO, allow asynchronous I/O to be switched off respectively for disk or tape devices. Because the number of I/O slaves for each process type defaults to zero, by default no I/O Slaves are deployed.

The DBWR_IO_SLAVES parameter should be set to greater than 0 only if the DISK_ASYNCH_IO, or the TAPE_ASYNCH_IO parameter has been set to FALSE, otherwise the database writer process (DBWR) becomes a bottleneck. In this case, the optimal value on AIX for the DBWR_IO_SLAVES parameter is 4.

The DB_WRITER_PROCESSES parameter specifies the initial number of database writer processes for an instance. If you use the DBWR_IO_SLAVES parameter, only one database writer process is used, regardless of the setting of the DB_WRITER_PROCESSES parameter.

Using the DB_FILE_MULTIBLOCK_READ_COUNT Parameter

A large value for the DB_FILE_MULTIBLOCK_READ_COUNT initialization parameter usually yields better I/O throughput. On AIX, this parameter ranges from 1 to 512, but using a value higher than 16 usually does not provide additional performance gain.

Set this parameter so that its value when multiplied by the value of the DB_BLOCK_SIZE parameter produces a number that is larger than the LVM stripe size. Such a setting causes more disks to be used.

Using RAID Capabilities

RAID 5 enhances sequential read performance, but decreases overall write performance. Oracle Corporation recommends using RAID 5 only for workloads that are not write-intensive. Intensive writes on RAID 5 might result in a performance degradation compared to a non-RAID environment.

RAID 0 and 1 generally result in better performance, as they introduce striping and mirroring at the hardware level, which is more efficient than at the AIX or Oracle level. RAID 7 is capable of providing better small and large read and write performance than RAID 0 to 6.

Using Write Behind

The write behind feature enables the operating system to group write I/Os together up to the size of a partition. Doing this increases performance because the number of I/O operations is reduced. The file system divides each file into 16 KB partitions to increase write performance, limit the number of dirty pages in memory, and minimize disk fragmentation. The pages of a particular partition are not written to disk until the program writes the first byte of the next 16 KB partition. To set the size of the buffer for write behind to eight 16 KB partitions, enter the following command:

```
# vmtune -c 8
```

To disable write behind, enter the following command:

```
# vmtune -c 0
```

Tuning Sequential Read Ahead

The Virtual Memory Manager (VMM) anticipates the need for pages of a sequential file. It observes the pattern in which a process accesses a file. When the process accesses two successive pages of the file, the VMM assumes that the program will continue to access the file sequentially, and schedules additional sequential reads of the file. These reads overlap the program processing and make data available to the program sooner. Two VMM thresholds, implemented as kernel parameters, determine the number of pages it reads ahead:

- MINPGAHEAD

  The number of pages read ahead when the VMM first detects the sequential access pattern

- MAXPGAHEAD

  The maximum number of pages that VMM reads ahead in a sequential file

Set the MINPGAHEAD and MAXPGAHEAD parameters to appropriate values for your application. The default values are 2 and 8 respectively. Use the `vmtune` command to change these values. You can use higher values for the MAXPGAHEAD parameter in systems where the sequential performance of striped logical volumes is of paramount importance. To set the MINPGAHEAD parameter to 32 pages and the MAXPGAHEAD parameter to 64 pages, enter the following command:

```
# vmtune -r 32 -R 64
```

Set both the MINPGAHEAD and MAXPGAHEAD parameters to a power of two. For example, 2, 4, 8,...512, 1042... and so on.

Tuning Disk I/O Pacing

Disk I/O pacing is an AIX mechanism that allows the system administrator to limit the number of pending I/O requests to a file. This prevents disk I/O intensive processes from saturating the CPU. Therefore, the response time of interactive and CPU-intensive processes does not deteriorate.

You can achieve disk I/O pacing by adjusting two system parameters: the high-water mark and the low-water mark. When a process writes to a file that already has a pending high-water mark I/O request, the process is put to sleep. The process wakes up when the number of outstanding I/O requests falls below or equals the low-water mark.

You can use the `smit` command to change the high and low-water marks. Determine the water marks through trial-and-error. Use caution when setting the water marks because they affect performance. Tuning the high and low-water marks has less effect on disk I/O larger than 4 KB.

Disk Geometry Considerations

On AIX, you can, to some extent, control the placement of a logical volume on a disk. Placing logical volumes with high disk activity close to each other can reduce disk seek time, resulting in better overall performance.

Minimizing Remote I/O Operations

Oracle9*i* Real Application Clusters running on the SP architecture uses VSDs or HSDs as the common storage that is accessible from all instances on different nodes. If an I/O request is to a VSD where the logical volume is local to the node, local I/O is performed. The I/O traffic to VSDs that are not local goes through network communication layers.

For better performance, it is important to minimize remote I/O as much as possible. Redo logs of each instance should be placed on the VSDs that are on local logical volumes. Each instance should have its own private rollback segments that are on VSDs mapped to local logical volumes if updates and insertions are intensive.

In each session, each user is allowed only one temporary tablespace. The temporary tablespaces should each contain at least one datafile local to each of the nodes.

Carefully design applications and databases (by partitioning applications and databases, for instance) to minimize remote I/O.

VSD Cache Buffers

Do not use VSD cache buffers (nocache) under normal situations for the following reasons:

- VSD LRU cache buffers use pinned kernel memory, which can be put to more effective use.
- When the cache buffer is enabled, every physical read incurs the overhead of searching the cache blocks for overlapping pages and copying data in and out of the cache buffers.

Use the `statvsd` command to check the performance of the VSD. If the `statvsd` command consistently shows requests queued waiting for buddy buffers, do not add more buddy buffers. Instead, increase the size of the switch send pool:

```
# /usr/lpp/ssp/css/chgcss -l css0 -a spoolsize=new_size_in_bytes
```

If the send pool size increases, you should also increase the mbuf parameter top ceiling mark:

```
# /etc/no -o thewall=new_size_in_kbytes
```

Note: The maximum value that you can specify is 64 MB.

The mbuf parameter top ceiling mark specifies the maximum amount of memory that can be used for network buffers. To check the current sizes of the send and receive pools, enter the following command:

```
$ /usr/sbin/lsattr -El css0
```

See Also: *Oracle9i Release Notes Release 2 (9.2.0.1.0) for AIX-Based Systems* for information on IBM Web addresses.

CPU Scheduling and Process Priorities

The CPU is another system component for which processes might contend. Although the AIX kernel allocates CPU effectively most of the time, many processes compete for CPU cycles. If your system has more than one CPU (SMP), there might be different levels of contention on each CPU.

Changing Process Running Time Slice

The default value for the runtime slice of the AIX RR dispatcher is ten milliseconds. Use the `schedtune` command to change the time slice. However, be careful when using this command. A longer time slice causes a lower context switch rate if the applications' average voluntary switch rate is lower. As a result, fewer CPU cycles are spent on context-switching for a process and the system throughput should improve.

However, a longer runtime slice can deteriorate response time, especially on a uniprocessor system. The default runtime slice is usually acceptable for most applications. When the run queue is high and most of the applications and Oracle shadow processes are capable of running a much longer duration, you might want to increase the time slice by entering the following command:

```
# /usr/samples/kernel/schedtune -t n
```

In the previous example, choosing a value for *n* of 0 results in a slice of 10 milliseconds (ms), choosing a value of 1 results in a slice of 20 ms, choosing a value of 2 results in a slice of 30 ms, and so on.

Using Processor Binding on SMP Systems

Binding certain processes to a processor can improve performance substantially on an SMP system. Processor binding is available and fully functional with AIX version 4 and higher.

Processor binding offers the following benefits:

- Provides higher-priority applications with a relatively larger share of CPU time
- Maintains the process context for a longer period

Processor binding on AIX is not automatic. On a multiprocessor system, you must explicitly bind a process to a processor by using the `bindprocessor` command. Only the `root` user or the Oracle software owner can bind an Oracle process to a processor. The child processes inherit the processor binding.

Oracle Corporation recommends binding the various Oracle background processes (except the database writer process) to different processors and leaving one processor free to service the database writer process. This guarantees the database writer a processor on which to execute and at the same time allows the database writer process to migrate freely to the other processors if it becomes CPU bound.

Note: Processor binding is a complicated issue and it should be handled with care. Processes bound to a processor cannot migrate to different processors even if these processors are free. This might degrade application performance. An environment of homogenous applications with a balanced load is more suitable for processor binding.

The binding of a process to a processor is not exclusive. The processor is free to execute other processes.

Processor Binding in a Networked Client and Server Environment

When an Oracle client process connects to an Oracle server process using an Oracle Net Services listener, the server process can be easily bound to a processor by binding the listener process. All Oracle server processes that the listener subsequently spawns are bound to the same processor.

One way to do this is to start multiple listeners, each listening on its own port. You must customize the `$ORACLE_HOME/network/admin/listener.ora` file to have one set of lines for each listener. You launch multiple listeners on the server side. On the client side, you might want to customize the `tnsnames.ora` file so that clients or even applications connect to different ports that are listened on by different listeners. For example, you can modify the `listener.ora` file and have two listeners, L1 and L2, that listen on ports 1521 and 1522, respectively, as follows:

1. Modify the `listener.ora` file as follows:

```
L1 =
  (ADDRESS_LIST =
     (ADDRESS= (PROTOCOL= TCP)(Host= nowhere)(Port= 1521))
  )
SID_LIST_L1 =
  (SID_LIST =
    (SID_DESC =
      (ORACLE_HOME= /oracle)
      (SID_NAME = ordb)
    )
  )
L2 =
  (ADDRESS_LIST =
      (ADDRESS= (PROTOCOL= TCP)(Host= nowhere)(Port= 1522))
  )
SID_LIST_L2 =
  (SID_LIST =
    (SID_DESC =
      (ORACLE_HOME= /oracle)
      (SID_NAME = ordb)
    )
  )
```

2. Start these two listeners:

```
$ lsnrctl start L1
$ lsnrctl start L2
```

3. Determine the process ids for the two listeners:

```
$ ps -ef | grep tnslsnr
```

4. Bind the listener processes to particular processors:

```
$ bindprocessor process_id_for_L1 id_for_process1
$ bindprocessor process_id_for_L2 id_for_process2
```

   In the preceding example, *id_for_process1* is 0, *id_for_process2* is 1, and so on.

Using this method, all Oracle server processes that handle communications with a particular client run on the same processor.

Processor Binding in a Local Environment

Processor binding is more difficult when the clients and the Oracle servers run on the same computer using the two-task pipe driver. You must determine the process ID for each server process and manually bind it to a processor. The administrative overhead is excessive and probably not worth the effort unless the servers have long process lives.

Processor binding of Oracle processes can have negative effects on the performance of other applications running on the same system. Careful tuning and monitoring is strongly recommended.

UDP Tuning

Oracle9*i* Real Application Clusters uses User Datagram Protocol (UDP) for interprocess communications. You can tune UDP kernel settings to improve Oracle performance. You can modify kernel UDP buffering on AIX by changing the udp_sendspace and udp_recvspace parameters.

- Set the value of the udp_sendspace parameter to a maximum of 65536.
- Set the value of the udp_recvspace parameter to less than the value of the sb_max parameter.

The value of the udp_recvspace parameter should be at least ten times the value of the udp_sendspace parameter because UDP might not be able to send a packet to an application before another packet arrives.

To determine the suitability of the udp_recvspace parameter settings, enter the following command:

```
$ netstat -s | grep "socket buffer overflows"
```

If the number of overflows is not zero, increase the value of the udp_recvspace parameter.

> **See Also:** *AIX Performance Tuning Guide, Version 3.2 and 4* or the *AIX5L Performance Management Guide* for more information on AIX tuning parameters.

Backing Up Raw Devices

Use the dd command to back up raw devices. Do not overwrite the first 4 KB block of a raw device, which is used by the AIX Logical Volume Manager. It is faster to back up the device using a blocksize of larger than 4K. A good blocksize for backing up to tape is 256K.

To back up the raw device to tape, enter a command similar to the following:

```
$ dd if=/dev/raw_device of=/dev/rmt0.1 bs=256k
```

To restore the raw device from tape, enter commands similar to the following:

```
$ dd if=/dev/rmt0.1 of=/dev/raw_device count=63 seek=1 skip=1 bs=4k
$ mt -f /dev/rmt0.1 bsf 1
$ dd if=/dev/rmt0.1 of=/dev/raw_device seek=1 skip=1 bs=256k
```

Note: For HSDs, do not skip the first 4 KB. Use the following command to restore the raw device, instead of the three preceding commands:

```
$ dd if=/dev/remt0.1 of=/dev/raw_device bs=256K
```

Resilvering with Oracle9*i*

If you disable mirror write consistency (MWC) for an Oracle datafile allocated on a raw logical volume (LV), the Oracle9*i* crash recovery process uses resilvering to recover after a system crash. This resilvering process prevents database inconsistencies or corruption.

During crash recovery, if a datafile is allocated on a logical volume with more than one copy, the resilvering process performs a checksum on the data blocks of all of the copies. It then performs one of the following:

- If the data blocks in a copy have valid checksums, the resilvering process uses that copy to update the copies that have invalid checksums.
- If all copies have blocks with invalid checksums, the resilvering process rebuilds the blocks using information from the redo log file. It then writes the datafile to the logical volume and updates all of the copies.

On AIX, the resilvering process works only for datafiles allocated on raw logical volumes for which MWC is disabled. Resilvering is not required for datafiles on mirrored logical volumes with MWC enabled, because MWC ensures that all copies are synchronized.

If the system crashes while you are upgrading a previous release of Oracle9*i* that used datafiles on logical volumes for which MWC was disabled, enter the `syncvg` command to synchronize the mirrored LV before starting the Oracle server. If you do not synchronize the mirrored LV before starting the server, Oracle might read incorrect data from an LV copy.

Note: If a disk drive fails, resilvering does not occur. You must enter the `syncvg` command before you can reactivate the LV.

Caution: Oracle Corporation supports resilvering for data files only. Do not disable MWC for redo log files.

B

Tuning for Oracle9*i* on HP

This appendix contains the following tuning information for Oracle9*i* on HP:

- HP-UX Shared Memory Segments for a 64-Bit Oracle Instance
- HP SCHED_NOAGE Scheduling Policy
- Lightweight Timer Implementation
- Asynchronous I/O

HP-UX Shared Memory Segments for a 64-Bit Oracle Instance

When a 64-bit Oracle server creates a database instance, the server creates memory segments by dividing the available shared memory by the value of the HP-UX SHMMAX kernel parameter. For example, if 64 GB of shared memory is available for a single Oracle instance and the value of the SHMMAX parameter is 1 GB, the Oracle server creates 64 shared memory segments for that instance.

In laboratory tests, performance degradation occurs when an Oracle instance creates more than six shared memory segments. This is because each shared memory segment receives a unique protection key when the Oracle server creates the instance. There are six protection keys available for shared memory segments on the PA-RISC processor. If your system has more than six shared memory segments, the HP-UX operating system displays protection key faults.

Oracle Corporation recommends that you set the SHMMAX parameter value to the amount of available physical memory on the system. Doing this ensures that the entire shared memory for a single Oracle instance is assigned to one shared memory segment and your instance needs only one protection key.

To display the list of active shared memory segments on the system, enter the following command:

```
$ ipcs -m
```

If the Oracle server creates more than six segments for the instance, increase the value of the SHMMAX kernel parameter.

> **See Also:** *Oracle9i Installation Guide Release 2 (9.2.0.1.0) for UNIX Systems* for more information on recommended kernel parameter settings.

HP SCHED_NOAGE Scheduling Policy

On HP, most processes use a time sharing scheduling policy. Time sharing can have detrimental effects on Oracle performance by descheduling an Oracle process during critical operations, for example, holding a latch. HP has a modified scheduling policy, referred to as SCHED_NOAGE, that specifically addresses this issue. Unlike the normal time sharing policy, a process scheduled using SCHED_NOAGE does not increase or decrease in priority, nor is it preempted.

This feature is suited to online transaction processing (OLTP) environments because OLTP environments can cause competition for critical resources. In laboratory tests, the use of the SCHED_NOAGE policy with Oracle9*i* increased performance by up to 10 percent in OLTP environments.

The SCHED_NOAGE policy creates little or no gains in decision support (DSS) environments because there is little resource competition in these environments. Because each application and server environment is different, you should test and verify whether your environment benefits from the SCHED_NOAGE policy.

Enabling SCHED_NOAGE for Oracle9*i*

To allow Oracle9*i* to use the SCHED_NOAGE scheduling policy, the group that the Oracle software owner belongs to (`dba`), must have the RTSCHED and RTPRIO privileges to change the scheduling policy and set the priority level for Oracle processes. To give the `dba` group these privileges, as the `root` user enter the following command:

```
# setprivgrp dba RTSCHED RTPRIO
```

To retain these privileges after rebooting, create the `/etc/privgroup` file, if it does not exist on your system, and add the following line to it:

```
dba RTSCHED RTPRIO
```

Add the HPUX_SCHED_NOAGE parameter to the initialization file for each instance, setting the parameter to an integer value to specify process priority levels. On HP-UX 11.0, the range is 153 to 255 and on HP-UX 11i, the range is 178 to 255. If the parameter setting is out of range, Oracle9*i* sets the parameter to a permissible value and continues with the SCHED_NOAGE policy with the new value. It also generates a message in the `alert_`*`sid`*`.log` file about the new setting. Oracle Corporation recommends that you set the parameter to the required priority level for Oracle processes.

> **See Also:** The HP documentation, the `rtsched(1)` man page, and the `rtsched(2)` man page for more information on priority policies and priority ranges.

Lightweight Timer Implementation

Unlike Oracle9*i* release 1 (9.0.1), Oracle9*i* release 2 (9.2.0.1.0) on HP-UX systems uses the `gethrtime()` system library call whether the TIMED_STATISTICS initialization parameter is set to TRUE or FALSE. This enables you to collect run-time statistics at any time while running an Oracle instance.

In laboratory tests, this library call provides a performance improvement of up to 10 percent over an Oracle system not using the `gethrtime()` system library call when the TIMED_STATISTICS initialization parameter is set to TRUE. In addition, there is no negative impact on Oracle9*i* release 2 (9.2.0.1.0) OLTP performance while using the `gethrtime()` system library call when the TIMED_STATISTICS initialization parameter is set to FALSE.

> **See Also:** *Oracle9i Installation Guide Release 2 (9.2.0.1.0) for UNIX Systems* for information on HP patches required for Oracle9*i*.

Asynchronous I/O

The asynchronous I/O pseudo-driver on HP allows the Oracle server to perform I/O to raw disk partitions using an asynchronous method, resulting in less I/O overhead and higher throughput. You can use the asynchronous I/O pseudo-driver on both HP-UX 9000 servers and workstations.

MLOCK Privilege

To allow Oracle9*i* to execute asynchronous I/O operations, the group that the Oracle software owner belongs to (`dba`) must have the MLOCK privilege. To give the `dba` group the MLOCK privilege:

1. As the `root` user, enter the following command:

   ```
   # setprivgrp dba MLOCK
   ```

2. To retain the MLOCK privilege after rebooting, create the `/etc/privgroup` file, if it does not exist on your system, and add the following line to it:

   ```
   dba MLOCK
   ```

Note: If the Oracle software owner does not have the MLOCK privilege, Oracle9*i* on HP-UX generates trace files that include the following error:

```
Ioctl ASYNC_CONFIG error, errno = 1
```

Implementing Asynchronous I/O

If you want to use asynchronous I/O on HP, you must use raw partitions for database files. Use the System Administrator Management (SAM) utility to configure the asynchronous disk driver into the HP kernel.

To add the asynchronous disk driver and configure the kernel using the SAM utility:

1. Enter the following command as the `root` user:

   ```
   # sam
   ```

2. Choose the Kernel Configuration area.
3. Choose the Drivers area.
4. Choose the asynchronous disk driver (`asyncdsk`).
5. Select Actions>Add Driver to Kernel.
6. Select List>Configurable Parameters.
7. Choose the MAX_ASYNC_PORTS parameter.
8. Select Action>Modify Configurable Parameter.
9. Specify a new value for the parameter, then choose OK.

   The MAX_ASYNC_PORTS parameter is a configurable HP kernel parameter that controls the maximum number of processes that can open the `/dev/async` file simultaneously.

   The system displays an error when a process tries to open the `/dev/async` file after the maximum number of processes have opened the file. This error can reduce performance on systems with a large number of shadow processes or many parallel query slaves performing asynchronous I/O. This error is not recorded. To avoid this error, estimate the highest likely number of processes that can access the `/dev/async` file and set the MAX_ASYNC_PORTS parameter to this value.

10. Choose Actions>Process a New Kernel.
11. Select one of the following options, then choose OK:
    - Move Kernel Into Place and Shutdown System / Reboot Now
    - Do Not Move Kernel Into Place: Do Not Shutdown / Reboot Now

    If you choose the second option, the new kernel, `vmunix_test`, and the `system.SAM` configuration file used to create it, are both created in the `/stand/build` directory.

To use the new kernel:

1. Enter the following command to move the new kernel into place:

   ```
   # /usr/sbin/kmupdate
   ```

2. Enter the following command to reboot the system:

   ```
   # shutdown -r
   ```

To enable asynchronous I/O operations using the HP asynchronous device driver:

1. Log in as the `root` user.
2. Enter the following command to create a new device file:

   ```
   # /sbin/mknod /dev/async c 101 0x0
   ```

3. Enter the following command to verify that the `/dev/async` device file exists and has the major number 101:

   ```
   # ls -l /dev/async
   ```

   The output of this command should look similar to the following:

   ```
   crw------- 1 orcle   oracle     101 0x000000 Oct 28 10:32  /dev/async
   ```

4. If necessary, give the device file the UNIX owner and permissions consistent with those of the Oracle software owner.

   If the Oracle software owner is `oracle`, enter the following commands:

   ```
   # /usr/bin/chown oracle:dba /dev/async
   # /usr/bin/chmod 660 /dev/async
   ```

Verifying Asynchronous I/O

To verify that asynchronous I/O is working:

1. Set the DISK_ASYNCH_IO initialization parameter to TRUE.
2. Start the Oracle database.
3. Enter the following command to start the GlancePlus/UX utility:

   ```
   $ gpm
   ```

4. In the main window, choose Reports>Process List.
5. In the Process List window, select the database writer process and choose Reports>Process Open Files.

   The list of files currently opened by the database writer process appears.

6. Look for the `/dev/async` file or the mode 101 0x000000 in the list of open files.

   If either the `/dev/async` file or the mode 101 0x000000 is in the list, the `/dev/async` file has been opened by the database writer process. This means that the database writer process is executing I/O through the HP asynchronous device driver and is working properly.

Asynchronous Flag in SGA

Oracle9*i* on HP uses a non-blocking polling facility provided by the HP asynchronous driver to check the status of I/O operations. This polling is performed by checking a flag that is updated by the asynchronous driver based on the status of the I/O operations submitted. HP requires that this flag be in shared memory.

Oracle9*i* configures an asynchronous flag in the SGA for each Oracle process. Oracle9*i* on HP has a true asynchronous I/O mechanism where I/O requests can be issued even though some previously issued I/O operations are not complete. This helps to enhance performance and ensures good scalability of parallel I/O processes.

Before Oracle8*i* release 8.1.7, the Oracle server was able to execute I/O operations only from shared memory using the HP asynchronous driver. Oracle8*i* release 8.1.7 executes I/O operations from both shared memory and process-private regions using the new HP asynchronous driver. However, I/O operations through the asynchronous driver are not asynchronous in nature. This is because Oracle8*i* must perform a blocking wait to check the status of I/O operations submitted to the asynchronous driver. Doing this causes some Oracle processes, for example the database writer process, to essentially execute synchronous I/O.

C

Tuning for Oracle9*i* on Linux

This appendix contains the following tuning information for Oracle9*i* on Linux:

- Extended Buffer Cache Support
- Asynchronous I/O Support

Extended Buffer Cache Support

Oracle9*i* can allocate and use more than 4 GB of memory for the database buffer cache. This section describes the limitations and requirements of the extended buffer cache support on Linux.

> **See Also:** *Oracle9i Database Concepts* for more information on the extend cache feature.

In-Memory File System

To use the extended buffer cache support on Linux, create an in-memory file system on the `/dev/shm` mount point equal in size or larger than the amount of memory that you intend to use for the database buffer cache.

For example, for Linux to create an 8 GB `shmfs` file system on the `/dev/shm` mount point, enter the following as the `root` user:

```
$ mount -t shm shmfs -o size=8g /dev/shm
```

When Oracle9*i* starts with the extended buffer cache feature enabled, it creates a file in the `/dev/shm` directory that corresponds to the Oracle buffer cache.

> **Note:** If an in-memory file system is already mounted on the `/dev/shm` mount point, ensure that it is equal to or larger than the amount of memory that is used for the database buffer cache.

USE_INDIRECT_DATA_BUFFERS Parameter

To enable the extended buffer cache feature, set the USE_INDIRECT_DATA_BUFFERS parameter to `true` in the `init`*`sid`*`.ora` file. Doing this allows Oracle9*i* to specify a larger buffer cache.

Dynamic Cache Parameters

Do not use the following dynamic cache parameters while the extended cache feature is enabled:

- DB_CACHE_SIZE
- DB_2K_CACHE_SIZE
- DB_4K_CACHE_SIZE
- DB_8K_CACHE_SIZE
- DB_16K_CACHE_SIZE
- DB_32K_CACHE_SIZE

If the extended cache feature is enabled, use the DB_BLOCK_BUFFERS parameter to specify the database cache size.

Limitations

The following limitations apply to the extended buffer cache feature on Linux:

- You cannot change the size of the buffer cache while the instance is running.
- You cannot create or use tablespaces with non-standard block sizes.

> **See Also:** *Oracle9i SQL Reference* for information on the standard block size used by the CREATE TABLE SPACE command.

Asynchronous I/O Support

Oracle9*i* supports kernel asynchronous I/O. This feature is disabled by default. If you are running Oracle9*i* on a system that supports kernel asynchronous I/O and that is certified by Oracle Corporation to use asynchronous I/O, perform the following steps to enable asynchronous I/O support:

1. As the `root` user, change directory to the `$ORACLE_HOME/rdbms/lib` directory.
2. Enter the following commands:

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

Note: If you receive the "`/usr/bin/ld: cannot find -laio`" error, then the system does not support kernel asynchronous I/O and you must enter the following command to restore the Oracle instance to a usable state:

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

By default, the DISK_ASYNCH_IO parameter in the `init`*`sid`*`.ora` file is set to `true` to enable asynchronous I/O on raw devices. To enable asynchronous I/O on filesystem files:

1. Ensure that all Oracle data files are located on file systems that support asynchronous I/O.
2. Set the FILESYTEMIO_OPTIONS parameter in the `init`*`sid`*`.ora` file to `asynch`.

D

Tuning for Oracle9*i* on Tru64

This appendix contains the following tuning information for Oracle9*i* on Tru64:

- Supporting Mixed CPU Systems
- Gathering Database Statistics on Tru64
- Oracle9i Real Application Clusters on Tru64
- Tuning Asynchronous I/O
- Direct I/O Support and Concurrent Direct I/O Support
- Enabling Access to the Real Time Clock
- Setting Up Raw Devices
- Spike Optimization Tool

Enabling Oracle9*i* Directed Placement Optimizations

Compaq GS80, GS160, and GS320 systems consist of smaller building blocks called Resource Affinity Domains (RADs). A RAD is a collection of tightly coupled CPUs, memory modules, and an I/O controller coupled through a fast interconnect. A second-level interconnect connects each of the RADs together to form a larger configuration.

Unlike previous generation servers which have only one common shared interconnect between CPUs, memory, and I/O controller, GS80, GS160, and GS320 servers can offer superior performance and memory access times when a particular CPU accesses memory within its own RAD or uses its local I/O controller. Because of the switched interconnect, all I/O activity and memory accesses within one RAD do not interfere with those within another RAD. However, because memory accesses between a CPU and memory module located across RAD boundaries must traverse two levels of interconnect hierarchy, these memory references take longer relative to memory references that are within a RAD.

Directed memory and process placement support (available on Tru64 UNIX V5.1 and higher) allows sophisticated applications to communicate their specific needs for process and memory layout to the operating system. This communication results in greater performance through increased localization of memory references within a RAD.

Oracle9*i* includes enhanced support for the special capabilities of high performance servers such as the GS80, GS160, and GS320. Directed placement optimizations specifically take advantage of hierarchical interconnects available in GS80, GS160, and GS320 class servers. All previous generation servers have a single shared interconnect, so these servers neither directly benefit from directed placement optimizations nor is there any loss of performance on these servers. Therefore, by default, these optimizations are disabled in Oracle9*i*.

Requirements to Run the Directed Placement Optimizations

The system must meet the following requirements for Oracle9*i* directed placement optimizations to work:

- The system must be a Compaq GS80, GS160, or GS320 AlphaServer or similar locality sensitive Compaq system. The Oracle9*i* optimizations only affect systems that are locality sensitive.
- The operating system must be Compaq Tru64 UNIX V5.1 or higher. Previous operating system versions do not include the required operating system support for Oracle9*i* to perform directed process and memory placement.

Enabling Oracle Directed Placement Optimizations

To enable Oracle directed placement optimizations, follow these steps:

1. Shut down the Oracle instance.
2. Relink the Oracle server by entering the following commands:

```
$ cd $ORACLE_HOME/rdbms/lib
$ make -f ins_rdbms.mk numa_on
$ make -f ins_rdbms.mk ioracle
```

If you are not using a compatible version of Tru64 UNIX, the following message is displayed:

```
Operating System Version Does not Support NUMA.
Disabling NUMA!
```

If you enable Oracle directed placement optimizations, and later change Tru64 to an incompatible version, disable Oracle directed placement optimizations as described in the following section.

Disabling Oracle Directed Placement Optimizations

To disable Oracle directed placement optimizations, follow these steps:

1. Shut down the Oracle instance.
2. Relink the Oracle server using the `numa_off` option:

```
$ cd $ORACLE_HOME/rdbms/lib
$ make -f ins_rdbms.mk numa_off
$ make -f ins_rdbms.mk ioracle
```

Using Oracle Directed Placement Optimizations

The Oracle directed placement optimizations assume an equi-partitioned configuration. This means that all RADs are configured with the same number of CPUs and the same amount of memory. The Oracle server is assumed to run across all RADs on the system.

Oracle Initialization Parameters

To make the most efficient use of the local environment, Oracle9*i* adjusts some initialization parameters automatically depending on the server configuration as reported by the operating system. This practice eliminates common errors in correctly computing subtle dependencies in these parameters.

Tru64 UNIX System Parameters

You must set the system parameters in the following table to realize the full benefits of a NUMA system:

Subsystem	Parameters	Setting
ipc	ssm_threshold	0
ipc	shm_allocate_striped	1 (default)
vm	rad_gh_regions[0] rad_gh_regions[1]... and so on	Size of the Shared Global Area in MBs divided by the number of RADs on the system

There are 63 rad_gh_regions parameters in the vm subsystem in Tru64 V5.1. Set only the parameters for the total number of RADs on the system. For example, if there are 4 RADs on the system (a GS160) and the SGA size is 10 GB, then set rad_gh_regions[0], rad_gh_regions[1], rad_gh_regions[2], and rad_gh_regions[3] to 2500. Note that you might have to raise this value slightly to 2501 or 2502 to successfully start the instance.

If CPUs and memory are taken off-line, Oracle9*i* continues to function, but loses performance. If you anticipate frequent off-lining of RADs or equi-partitioning is not feasible, Oracle Corporation recommends running Oracle9*i* Real Application Clusters, using one instance per RAD. Using Oracle9*i* Real Application Clusters, you can configure individual instances with different sets of initialization parameters to match the actual RAD configuration. You can also start up or shut down specific instances without affecting overall application availability.

Process Affinity to RADs

You can improve performance by directing the operating system to run the processes on specific RADs. If connections to the database are made through the Oracle Listener process, and there is a corresponding network interconnect adapter on the RAD, you can run a listener on each RAD. To run the listener on a particular RAD, enter the following command:

```
$ runon -r lsnrctl start [listener_name]
```

All Oracle shadow processes are automatically created on the same RAD as the Oracle listener.

Supporting Mixed CPU Systems

Compaq systems using Tru64 V5.1A or higher can have mixed CPU speeds and types. All CPUs in a single RAD must have the same speed and cache size. Another RAD can have a set of CPUs with a different speed and cache size.

The performance of a mixed CPU system depends on the proportion of slower CPUs to faster CPUs. Also, performance is affected by the placement of Oracle processes on the system. In a high transaction Online Transaction Processing (OLTP) environment, placing the database writer and log writer processes on the slower CPUs can adversely affect performance. In a data warehousing or decision support environment, placing the database writer and log writer processes on the slower CPUs might not be noticeable at all.

The ability to mix CPU systems enables you to protect your hardware investment. You can add faster and more powerful CPUs to a system without needing to replace older CPUs. Compaq and Oracle Corporation have tested and support mixed CPU systems.

Note: You should not expect the mixed CPU system to perform as well as a system made up entirely of the fastest CPUs of the mixed CPU system. However, a mixed CPU system should perform better than a system made up entirely of the slowest CPUs of the mixed CPU system. Contact Compaq for a complete list of rules and restrictions for mixed CPU systems.

Gathering Database Statistics on Tru64

Oracle9*i* release 2 (9.2.0.1.0) runs only on Tru64 UNIX V5.1 or higher. This is because Compaq changed the size of the long double data type from 64 bits on Tru64 UNIX V4.0x to 128 bits on Tru64 UNIX V5.*x*. This change causes certain Oracle operations to perform with increased precision. One of these operations stores statistics in the data dictionary after a table or index is analyzed.

The query optimizer within the Oracle server uses the statistics stored in the data dictionary to determine how best to execute a query. If a stored statistic does not match a statistic calculated by the query optimizer while it searches for the best plan, the query optimizer might use the wrong plan to execute the query. This can cause the query to perform poorly or fail.

For this reason, after upgrading from Oracle8*i* release 8.1.7 or lower to Oracle9*i* release 2 (9.2.0.1.0) you should analyze all object statistics for each schema. There is no need to reanalyze any schemas after upgrading from Oracle9*i* release 1 (9.0.1) to Oracle9*i* release 2. You can use the DBMS_STATS.GATHER_SCHEMA_STATS procedure to perform the analysis to gather statistics for each schema. The DBMS_STATS package saves the current table or index statistics in a table in case the new statistics cause problems.

> **See Also:** *Oracle9i Supplied PL/SQL Packages and Types Reference* for more information on gathering database statistics.

Oracle9*i* Real Application Clusters on Tru64

This section describes Oracle9*i* Real Application Clusters on Tru64.

Reliable Data Gram

Reliable Data Gram (RDG) is an IPC infrastructure for the Tru64 TruCluster platform. It is the default IPC method for Oracle9*i* on Tru64 and is optimized for Oracle9*i* Real Application Clusters environments.

Requirements

RDG requires that the node be a member of the cluster and connected through the memory channel. Oracle Corporation recommends that you set the node-wide operating system parameters listed in Table D–1 when using RDG.

Table D–1 RDG Subsystem Operating System Parameter Settings

Parameter	Setting
max_objs	At least 5 times the number of Oracle processes per node and up to the larger of 10240 or the number of Oracle processes multiplied by 70.
msg_size	Equal to or greater than the maximum value of the DB_BLOCK_SIZE parameter for the database. Oracle Corporation recommends a value of 32768 because Oracle9*i* supports different block sizes for each tablespace.
max_async_req	At least 100 or the operating system default, whichever is larger. **Note**: A value of 1000 or greater might provide better performance.
max_sessions	At least the number of Oracle processes plus 2.
rdg_max_auto_msg_wires	Must be set to 0.

Enabling UDP IPC

With Oracle9*i*, RDG is the default IPC method on Tru64. When the Oracle9*i* Real Application Clusters option is enabled, the Global Cache Service (GCS), Global Enqueue Service (GES), Interprocessor Parallel Query (IPQ), and Cache Fusion use RDG. The User Datagram Protocol (UDP) IPC implementation is still available but you must enable it explicitly.

You must enable the Oracle9*i* Real Application Clusters option before enabling UDP IPC. To enable the Oracle9*i* Real Application Clusters option, use the Oracle Universal Installer or enter the following commands:

```
$ cd $ORACLE_HOME/rdbms/lib
$ make -f ins_rdbms.mk rac_on
$ make -f ins_rdbms.mk ioracle
```

To make the Oracle IPC routines use the UDP protocol, you must relink the `oracle` executable. Before performing the following steps, shut down all instances in the cluster.

To enable UDP IPC, enter the following commands:

```
$ cd $ORACLE_HOME/rdbms/lib
$ make -f ins_rdbms.mk ipc_udp
$ make -f ins_rdbms.mk ioracle
```

To disable UDP IPC and revert to the default implementation for Oracle9*i* Real Application Clusters, enter the following commands:

```
$ cd $ORACLE_HOME/rdbms/lib
$ make -f ins_rdbms.mk rac_on
$ make -f ins_rdbms.mk ioracle
```

CLUSTER_INTERCONNECTS Initialization Parameter (Formerly TRU64_IPC_NET)

In Oracle9*i* release 2 (9.2.0.1.0) on Tru64 UNIX, the TRU64_IPC_NET parameter is replaced by the CLUSTER_INTERCONNECTS parameter. This parameter requires the IP address of the interconnect instead of the device name. The CLUSTER_INTERCONNECTS parameter allows your system to specify multiple IP addresses. Oracle9*i* Real Application Clusters traffic is distributed between all of the specified IP addresses.

The CLUSTER_INTERCONNECTS parameter is useful only if Oracle9*i* Real Application Clusters and UDP IPC are enabled. They enable users to specify an interconnect for all IPC traffic that includes Oracle GCS, GES, and IPQ.

Use the CLUSTER_INTERCONNECTS parameter when the Memory Channel interconnect is overloaded. Overall cluster stability and performance might improve when you force Oracle GCS, GES, and IPQ over a different interconnect by setting the CLUSTER_INTERCONNECTS parameter. For example, to use the first fiber distributed data interface (FDDI) network controller whose IP address is 129.34.137.212 for all GCS, GES, and IPQ IPC traffic, set the CLUSTER_INTERCONNECTS parameter as follows:

```
CLUSTER_INTERCONNECTS=129.34.137.212
```

Use the `/usr/sbin/ifconfig -a` command to display the IP address of a device. This command provides a map between device names and IP addresses. To determine the IP address of a device, enter the following command:

```
$ /usr/sbin/ifconfig -a
fta0: flags=c63<UP,BROADCAST,NOTRAILERS,RUNNING,MULTICAST,SIMPLEX>
      inet 129.34.137.212 netmask fffffc00 broadcast 129.34.139.255 ipmtu 1500

lo0:  flags=100c89<UP,LOOPBACK,NOARP,MULTICAST,SIMPLEX,NOCHECKSUM>
      inet 127.0.0.1 netmask ff000000 ipmtu 4096

mc0:  flags=1100063<UP,BROADCAST,NOTRAILERS,RUNNING,NOCHECKSUM,CLUIF>
      inet 10.0.0.1 netmask ffffff00 broadcast 10.0.0.255 ipmtu 7000
```

```
sl0:  flags=10<POINTOPOINT>

tun0: flags=80<NOARP>
```

In the preceding example, device fta0: has an IP address of 129.34.137.212 and device mc0: has an IP address of 10.0.0.1.

Remember the following important points when using the CLUSTER_INTERCONNECTS initialization parameter:

- The CLUSTER_INTERCONNECTS parameter is used only when UDP is enabled as the IPC implementation.
- The IP addresses specified for the different instances of the same database on different nodes should belong to network adaptors that connect to the same network. If you do not follow this rule, internode traffic may pass through bridges and routers or there may not be a path between the two nodes at all.
- Specify the CLUSTER_INTERCONNECTS parameter in the instance-specific initialization parameter file. Do not specify the CLUSTER_INTERCONNECTS parameter in the common initialization parameter file because the devices on different nodes connected to the same network have different IP addresses.
- If you specify multiple IP addresses for this parameter, list them in the same order for all instances of the same database. For example, if the parameter for instance 1 on node 1 lists the IP addresses of the alt0, fta0 and mc0 devices in that order, the parameter for instance 2 on node 2 should list the IP addresses of the equivalent network adaptors in the same order.
- If the interconnect device or IP address specified is incorrect or does not exist on the system, Oracle9*i* uses the default cluster interconnect device. On Tru64 UNIX V5.1, the default device is `mc0`. On Tru64 UNIX V5.1A and above, the default device is `ics0`.

  Oracle9*i* does not confirm which device is being used. To determine the IP address of the cluster interconnect device being used, perform the following steps:

  1. Enter the following command:

     ```
     $ /usr/sbin/clu_get_info
     ```

2. In the output for this command, identify the cluster interconnect IP name, stored in the `Hostname` parameter, and the cluster interconnect address, stored in the `Cluster interconnect IP address` parameter. In the following example, the cluster interconnect IP name is `server1` and the address is `10.0.0.1`:

```
Information on each cluster member

Cluster memberid = 1
Hostname = server1.employee.records
Cluster interconnect IP name = server1-mc0
Cluster interconnect IP address = 10.0.0.1
Member state = UP
Member base O/S version = Compaq Tru64 UNIX V5.1 (Rev. 732)
Member cluster version = TruCluster Server V5.1 (Rev. 389)
Member running version = INSTALLED
Member name = server1
Member votes = 1
csid = 0x20002
```

Tuning Asynchronous I/O

Oracle9*i* for Tru64 systems can perform either synchronous or asynchronous I/O. To improve performance, Oracle Corporation recommends that you use asynchronous I/O. Set the DISK_ASYNCH_IO parameter to TRUE to enable asynchronous I/O.

Oracle9*i* can use asynchronous I/O on any datafiles that are stored on AdvFS file systems, clustered file systems (CFS), or raw devices. You must tune some operating system parameters for optimal asynchronous I/O performance.

aio_task_max_num Parameter

Set the aio_task_max_num operating system parameter for a single instance to the higher of the following:

- Greater than the maximum of the DBWR I/O operations
- The value of the DB_FILE_MULTIBLOCK_READ_COUNT initialization parameter

The maximum number of DBWR I/O operations defaults to 8192.

You should adjust the setting of the aio_task_max_num parameter to accommodate any other applications that use asynchronous I/O, including multiple Oracle9*i* instances on a single node. Set the value of the parameter to the maximum number of I/O requests that any application can issue. For example, if three applications are running on a system and application one can issue a maximum of 10 simultaneous asynchronous I/O requests, application two can issue 100 simultaneous asynchronous I/O requests, and application three can issue 1000 simultaneous asynchronous I/O requests, you should set the aio_task_max_num parameter to at least 1000.

If you do not set the aio_task_max_num operating system parameter as described in this section, the performance of Oracle9*i* is reduced and spurious I/O errors might occur. These errors are stored in the alert log and trace files.

Direct I/O Support and Concurrent Direct I/O Support

This section describes support for direct and concurrent I/0.

Single Instance Requirements

Oracle9*i* has the following requirements for single instance installations:

- Tru64 UNIX V5.1 or later with the appropriate patchkits.

  See Also: *Oracle9i Installation Guide Release 2 (9.2.0.1.0) for UNIX Systems* for information on Tru64 patchkits.

- Oracle datafiles stored on a Tru64 UNIX AdvFS file system.
- The disks that use the AdvFS file system must be physically connected to the computer running the Oracle9*i* instance. This includes disks attached by fiber channel. This specifically excludes cases where I/O must be served by another node because of a lack of physical connectivity.

On Tru64 UNIX V5.1 systems and higher in a non-clustered system environment, the AdvFS file system and direct I/O give almost all of the performance of raw devices because the file system cache is not used. In addition to this, the file system allows you to more easily manage the database files.

Clustered Systems

On V5.1 systems and higher, Tru64 supports Clustered File Systems (CFS). CFS provides a single namespace file system for all nodes in a cluster. All file systems mounted in a cluster are automatically seen by all nodes in the cluster. Because it is layered on top of the AdvFS file system, the CFS file system inherits much of the characteristics of non-clustered systems.

Tru64 UNIX V5.1 Clustered Systems

Oracle Corporation supports CFS only on Tru64 UNIX V5.1 or later because this file system now supports a concurrent direct I/O model. Any node that has physical connectivity to a drive can issue data I/O to its file systems without consulting with the owning node.

All metadata changes to a file, for example extending, closing, changing the access or modification date, are still served by the owner node and can still cause cluster interconnect saturation. Therefore, it is possible for the CREATE TABLESPACE, ALTER TABLESPACE, ADD DATAFILE, ALTER DATABASE DATAFILE, or RESIZE commands to perform poorly on a CFS file system when compared to raw devices.

Multiple Instance Requirements (Oracle9*i* Real Application Clusters)

Oracle9*i* Real Application Clusters requires that you store Oracle datafiles on the Tru64 AdvFS file system. The disks that use the AdvFS file system must be physically connected to all computers running the Oracle instances. This includes disks attached by fiber channel. It excludes cases where I/O must be served by another node because of physical connectivity.

If the database is running in archive mode and the archive logs are being written to disk, the destination AdvFS domain should be served by the node of the instance that is archiving the redo log. For example, if you have a three-node cluster with one instance on each node (nodea, nodeb, and nodec), you must also have three archive destination AdvFS domains (arcnodea, arcnodeb, and arcnodec). The domains should be served by nodea, nodeb, and nodec respectively and the LOG_ARCHIVE_DEST initialization parameter for each instance should specify their respective locations.

Disabling Direct I/O Support

An Oracle9*i* database running on an AdvFS file system with direct I/O support enabled should perform as well as an Oracle9*i* database running on raw devices. In most cases, an Oracle9*i* database that is stored on AdvFS volumes with direct I/O support enabled should perform the same as or better than the same database with direct I/O support disabled. However, the following workload attributes can reduce performance when direct I/O support is enabled:

- A high read to write ratio
- Oracle data blocks not cached in the SGA because the query utilizes parallel query slaves
- A Unix Buffer Cache (UBC) several megabytes or larger
- Full table scan queries where the same set of tables are scanned repeatedly
- Tables being scanned can fit in the UBC

When direct I/O support is disabled, workloads that have most of the attributes in the preceding list rely heavily on the UBC. Because most if not all of the tables being scanned are cached in the UBC, the I/O requests issued by the parallel query are met by the UBC. As a result, the query completes much faster than if all of the data had to be read from disk, as it would with direct I/O enabled.

When direct I/O support is enabled, Oracle data blocks are not cached in the UBC. They are read into process-private memory instead. This means that any query that reads a previously-scanned table must perform I/O requests to disk to retrieve the data. Disk I/O latencies are several orders of magnitude slower than memory latencies. Therefore, the query runs slower and performance suffers.

If your workload has most of the attributes described in the preceding list, disabling direct I/O support will probably improve performance. However, often there are many different types of queries running on the system at the same time. Some queries only read data while others insert, modify, or delete data and the ratio of the various types of queries differ from site to site. Generally, if your site has more of an OLTP workload, disabling direct I/O support does not improve performance.

Direct I/O support is enabled by default in Oracle9*i* release 2 (9.2.0.1.0). The undocumented _TRU64_DIRECTIO_DISABLED initialization parameter that is used to disable direct I/O support in Oracle9*i* release 1 (9.0.1) is removed in Oracle9*i* release 2 (9.2.0.1.0). The generic FILESYSTEMIO_OPTIONS initialization parameter is used instead. The following table describes the valid values for the FILESYSTEMIO_OPTIONS parameter as interpreted on Tru64:

Value	Description
directio	Implies that direct I/O support is enabled but asynchronous I/O support is not enabled for I/O to files on an AdvFS files system.
asynch	Equivalent to none because asynchronous I/O support is enabled for AdvFS files only if direct I/O support is also enabled.
setall	Implies that both direct I/O and asynchronous I/O support are enabled for AdvFS files. This is the default option.
none	Disables both direct I/O support and asynchronous I/O support on AdvFS files.

See Also: *Oracle9i Reference Guide* for more information on the FILESYSTEMIO_OPTIONS initialization parameter.

The DISK_ASYNCH_IO initialization parameter controls the asynchronous I/O state for all database files, whether they are on file systems or raw devices. Therefore, if the DISK_ASYNCH_IO parameter is set to FALSE, all I/O requests to file system files are synchronous regardless of the value of the FILESYSTEMIO_OPTIONS parameter. The DISK_ASYNCH_IO parameter defaults to TRUE.

Preventing File Fragmentation

Because of an interaction between direct I/O and file allocation, files created with direct I/O support enabled can become severely fragmented. Severely fragmented files cause performance degradation and can lead to I/O errors, especially during backups and recovery. Oracle9*i* release 2 (9.2.0.1.0) solves this problem by temporarily disabling direct I/O support during file creation and file extension. If direct I/O support is enabled, the new file or extended file is reopened with direct I/O support enabled after the file create or resize operation is complete. On a file resize operation, direct I/O support is not temporarily disabled if the new file size is smaller than the current file size. This does not cause fragmentation because the file or extent already exists.

Enabling Access to the Real Time Clock

Many Oracle processes are timed, especially if the TIMED_STATISTICS initialization parameter is set to TRUE. These timing functions call the Tru64 kernel and can affect Oracle9*i* performance. On Tru64, you can improve performance on heavily loaded systems by enabling processes to directly access the real time clock.

To enable access to the real time clock:

1. Log in as `root`.
2. Enter the following commands:

   ```
   # mknod /dev/timedev c 15 0
   # chmod +r /dev/timedev
   ```

   If your system is a cluster running Tru64 UNIX V5.1 or higher, enter these commands on each cluster. If your system is a cluster running an earlier version of Tru64, enter the commands on each node.

   Note: The special file `/dev/timedev` remains on the system after rebooting.

3. Restart the Oracle9*i* instance.

   The system checks for the existence of the `/dev/timedev` file only on instance startup.

Oracle Corporation recommends that you enable this feature on all instances in a cluster, and therefore on all nodes.

Setting Up Raw Devices

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

To set up raw devices / volumes on Tru64 systems:

1. If you are using Oracle9*i* Real Application Clusters, make sure that the partitions you are adding are on a shared disk. However, if your platform supports a cluster file system certified by Oracle Corporation, you can store the files that Oracle9*i* Real Application Clusters requires directly on the cluster file system.
2. Determine the names of the free disk partitions.

   A free disk partition is one that is not used for a Tru64 file system that complies with the following restrictions:

   - It is not listed when you execute the `/usr/sbin/mount` command.
   - It is not in use as a swap device.
   - It does not overlap a swap partition.
   - It is not in use by other Tru64 applications (for example, other instances of the Oracle9*i* server).
   - It does not overlap the Tru64 file system.
   - It does not use a space already used by the file system.

   To determine 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. Some partitions may contain file systems that are currently not mounted and are not listed in the `/usr/sbin/mount` output.

   Note: Make sure that the partition does *not* start at cylinder 0.

3. Set up the raw device for use by the Oracle9*i* Server.

   Begin by verifying that the disk is partitioned. If it is not, use the `disklabel` command to partition it.

4. Enter the `ls` command to view the owner and permissions of the device file. For example:

   ```
   $ ls -1a
   ```

5. Make sure that the partition is owned by the Oracle software owner. If necessary, use the chown command to change the ownership on the block and character files for the device. For example:

```
# chown oracle /dev/rdisk/dsk10c
```

6. Make sure that the partition has the correct permissions. If necessary, use the chmod command to make the partition accessible to only the Oracle software owner. For example:

```
# chmod 600 /dev/rdisk/dsk10c
```

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

```
$ ln -s /dev/rdisk/dsk10c /oracle_data/datafile.dbf
```

   To verify that you have created the symbolic link, use the character special device (not the block special device) and enter the following command:

```
$ ls -Ll datafile
```

   The following message should appear:

```
crwxrwxrwx oracle dba datafile
```

> **Caution:** This symbolic link must be set up on each node of the cluster. Check that no two symbolic links specify the same raw device.

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

   To create a new partition, from SQL*Plus enter the following SQL command:

> **Note:** The size of an Oracle datafile created in a raw partition must be at least 64 KB plus one Oracle block size smaller than the size of the raw partition.

```
SQL> CREATE DATABASE sid
  2  LOGFILE '/oracle_data/log1.dbf' SIZE 100K
  3 '/oracle_data/log2.dbf' SIZE 100K
  3  DATAFILE '/oracle_data/datafile.dbf' SIZE 10000K REUSE;
```

To add a partition to a tablespace in an existing Oracle database, enter:

```
SQL> ALTER TABLESPACE tablespace_name
  2  ADD DATAFILE '/dev/rdisk/dsk10c' SIZE 10000K REUSE;
```

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

Spike Optimization Tool

The Spike optimization tool (Spike) is a performance optimization tool that increases the performance of a Tru64 binary. In a testing environment, Spike, with feedback, increased the performance of the Oracle9*i* server by up to 23 percent on an OLTP workload.

For information on Spike, see the Tru64 documentation or enter one of the following commands:

- `man spike`
- `spike`

Oracle9*i* requires Spike version V5.1 (1.2.2.31.2.4 ADK) Feb 22 2001 or later.

Note: If you have a version of Spike earlier than V5.1 (1.2.2.31.2.4 ADK) Feb 22 2001, contact Compaq for a patchkit.

Enter the following command to check the version of Spike:

```
$ spike -V
```

You can download the latest version of Spike from the Compaq Web site.

Note: Oracle Corporation does not support versions of the Oracle executable optimized using the `spike` command. If you encounter a problem in an Oracle9*i* binary that has been optimized using Spike, reproduce the problem with the original un-optimized binary. If the problem persists, see the "Preface" for information on Oracle services and support.

Using Spike

This section describes the system resources required by Spike, how and why to use Spike optimization flags, and the various ways to run Spike.

Setting System Resources

Table D–2 lists the system resources required to run Spike.

Table D–2 System Resource Requirements for Spike

Resource	Minimum Value
Physical memory	1024 MB
max-per-proc-address-space parameter in the `sysconfigtab` file	1024 MB
max-per-proc-data-space parameter in the `sysconfigtab` file	1024 MB
vm-maxvas parameter in the `sysconfigtab` file	1024 MB

To set the value of these parameters in the `/etc/sysconfigtab` file, edit the following lines:

```
proc:
    max-per-proc-address-space = 0x40000000
    max-per-proc-data-size = 0x40000000
vm:
    vm-maxvas = 0x40000000
```

Set the limits in your shell environment to the highest values. For the C shell, enter:

```
% limit datasize unlimited
% limit memoryuse unlimited
% limit vmemoryuse unlimited
```

Spike can run out of virtual memory if the stacksize limit is set too high. To avoid this problem, enter the following C shell command:

```
% limit stacksize 8192
```

Checking Optimization Flags

Spike provides a large number of optimization flags. However, you cannot use all `spike` command optimizations with Oracle9*i*. The following Spike optimization flags are certified to run with Oracle9*i*:

```
-arch, -controlOpt, -fb, -feedback, -map, -nosplit, -nochain, -noporder,
-noaggressiveAlign, -o, optThresh, -splitThresh, -symbols_live, -tune, -v, -V
```

When you run Spike, it places a copy of the optimization flags in the image header comment section of the binary that you are optimizing. Oracle9*i* checks Spike optimizations used on itself at the beginning of instance startup. If Oracle9*i* detects an optimization not known to work for the Oracle9*i* binary, or if the binary had been previously optimized with OM (the predecessor to Spike from Compaq), the instance startup fails with an ORA-4940 error message. If the instance startup fails, check the alert log file for more information.

Note: Oracle9*i* release 2 (9.2.0.1.0) requires that you use the Spike `-symbols_live` optimization flag.

Running Spike

Use one of the following methods to optimize an executable using Spike:

- Static spiking
- Running Spike with feedback

Static spiking requires only a few set-up steps and yields approximately half the performance benefit possible compared to running Spike with feedback.

Running Spike with feedback includes all of the optimizations of static spiking plus additional optimizations that are workload-related. Running spike with feedback provides the best possible performance benefit, however, it requires considerably more effort than static spiking.

For both running Spike with feedback and static spiking, Oracle Corporation recommends running the spiked Oracle binary in a test environment before moving it to a production environment.

Static Spiking

Static spiking performs optimizations that are not specific to your workload, such as manipulating the global pointer (gp) register and taking advantage of the CPU architecture. In a test environment, roughly half of the performance optimization

gain possible from Spike was through static spiking. Furthermore, static spiking is relatively straight-forward and simple. The combination of simplicity and performance gain makes static spiking worth the effort.

Perform the following steps to use static spiking:

1. Shut down the database.
2. Spike the `oracle` image by entering the following command:

   ```
   $ spike oracle -o oracle.spike -symbols_live
   ```

3. Save the original image and create a symbolic link to the spiked image by entering the following commands:

   ```
   $ mv oracle oracle.orig
   $ ln -s oracle.spike oracle
   ```

4. Start up the database.

> **Note:** Before contacting Oracle for support, you must use the original image to reproduce any problems.

Running Spike with Feedback

Running Spike with feedback performs all of the same optimizations as static spiking plus optimizations that are workload-related such as hot and cold basic block movement. In a test environment, approximately half of the performance optimizations gained from Spike was due to the optimizations that depend on feedback information. Running Spike with feedback requires multiple steps and considerably more effort than static spiking. However, performance sensitive customers may find the extra effort worthwhile.

Perform the followings steps to run Spike with feeback:

1. Instrument the Oracle binary by entering the following command:

   ```
   $ pixie -output oracle.pixie -dirname dir -pids oracle_image
   ```

   In the preceding example, *`oracle_image`* is your original image and *`dir`* is the name of the directory into which the instrumented executable writes the profiling data files.

> **Note:** The `-dirname` option saves the `oracle.Counts.`*`pid`* files in the *`dir`* directory. Because these files are large and may be numerous depending on the workload, make sure that the directory has enough disk space.

This step also creates an `oracle.Addrs` file that is required later.

The output of the `pixie` command might contain errors. You can safely ignore these errors.

2. Shut down the database.
3. Save the original image and create a symbolic link to the pixie image by entering the following commands:

```
$ mv oracle oracle.orig
$ ln -s oracle.pixie oracle
```

4. Start up the database and run your workload.

   You cannot run as many users as you can with the standard executable because the `pixie` executable is larger and slower. As you use the Oracle9*i* server, several `oracle.Counts.`*`pid`* files are created, where *`pid`* is the process ID for the corresponding Oracle process. Keep track of the process id of each Oracle process for which the optimization is aimed. These could be the shadow Oracle processes of the clients.

5. Shut down the database.
6. Create a symbolic link to replace the original executable by entering the following command:

```
$ ln -s oracle.orig oracle
```

7. If you can identify one `oracle.Counts.`*`pid`* file as representative of your workload, perform step a. If you must merge several counts files together to better represent your workload, perform step b.

   a. Make sure that the `oracle.Addrs` file created by the `pixie` command, the `oracle.Counts.`*`pid`* files, and the original Oracle executable are available.

      Use the process id (pid) to pick a representative `oracle.Counts.`*`pid`* file and then copy it by entering the following command:

```
$ cp oracle.Counts.pid oracle.Counts
```

b. Use the `prof` utility to merge several `oracle.Counts.`*`pid`* files. See the `prof` man pages for more information on this utility.

If you are using the parallel query option, merge the `oracle.Counts.`*`pid`* files generated by the query slaves and the query coordinator, which is the shadow Oracle process of the query-initiating client.

If you are not using the parallel query option, merge the `oracle.Counts.`*`pid`* files from the Oracle foreground processes that use the most memory.

To merge the `oracle.Counts.`*`pid`* files, enter the following command:

```
$ prof -pixie -merge oracle.Counts $ORACLE_HOME/bin/oracle \
oracle.Addrs oracle.Counts.pid1 oracle.Counts.pid2
```

8. Make sure that the `oracle.Addrs` and `oracle.Counts` files are available in the current directory, then run Spike using the feedback information by entering the following command:

```
$ spike oracle -fb oracle -o oracle.spike_fb -symbols_live
```

The output of the `spike` command might contain errors. You can safely ignore these errors.

9. Create a symbolic link to the new `oracle` image by entering the following command:

```
$ ln -s oracle.spike_fb oracle
```

10. Start up the database.

E

Running Oracle *inter*Media, Oracle Text, and Oracle Spatial Demonstrations

This chapter contains information on running the Oracle9*i* *inter*Media, Oracle Text, and Oracle Spatial demonstrations. It contains the following sections:

Oracle *inter*Media

Oracle *inter*Media is an integrated suite of services that extends Oracle9*i* by offering services to store, manage, and retrieve image, audio, and video data. It also provides location services, and annotation services for multimedia data. Oracle *inter*Media includes the following components:

- Oracle interMedia Audio, Image, and Video Services
- Oracle interMedia Annotator
- Locator
- Clipboard

Oracle *inter*Media Audio, Image, and Video Services

Oracle *inter*Media includes a number of scripts and sample programs. See the README files in the directories listed in Table E–1 for more information on each script or sample program.

Table E–1 Oracle interMedia Demonstration Programs

Oracle *inter*media Demonstration	Directory
ORDAudio SQL	`$ORACLE_HOME/ord/aud/demo/`
ORDAudio Java	`$ORACLE_HOME/ord/aud/demo/java/`
ORDDoc SQL	`$ORACLE_HOME/ord/doc/demo/`
ORDDoc Java	`$ORACLE_HOME/ord/doc/demo/java/`
ORDImage OCI	`$ORACLE_HOME/ord/img/demo/`
ORDImage Java	`$ORACLE_HOME/ord/img/demo/java/`
ORDVideo SQL	`$ORACLE_HOME/ord/vid/demo/`
ORDVideo Java	`$ORACLE_HOME/ord/vid/demo/java/`
Code Wizard for PL/SQL Gateway	`$ORACLE_HOME/ord/http/demo/plsgwycw`
PL/SQL Web Toolkit Demonstration	`$ORACLE_HOME/ord/http/demo/plsqlwtk`
Java Servlet Photo Album	`$ORACLE_HOME/ord/http/demo/servlet`
Java Server Pages Photo Album	`$ORACLE_HOME/ord/http/demo/jsp`
Relational Interface	`$ORACLE_HOME/ord/im/demo/relintfc`

Oracle *inter*Media Annotator

Oracle *inter*Media Annotator includes demonstrations that you can modify and run. For information on the location of these demonstrations, see the `$ORACLE_HOME/ord/Annotator/README.txt` file.

Locator

Oracle9*i* Locator includes demonstrations that you can modify and run. These demonstrations are located in the `$ORACLE_HOME/md/demo` directory.

Clipboard

Oracle Clipboard, a component of Oracle *inter*Media, is not available on the Oracle9*i* CD. To download the Oracle Clipboard from the Oracle Technology Network (OTN) Web site:

> **Note:** You must have an OTN account to download software from this Web site. Click the Membership button on the OTN index page to register for an OTN account.

1. Enter the following URL in a browser:

   `http://otn.oracle.com/products/intermedia/content.html`

2. Click the Software link under the *inter*Media heading, in the column on the left.

   The login dialog box appears.

3. Enter your OTN username and password and click the OK button.

> **See Also:** README files and the *Using Oracle interMedia on the Web* guide, included in the download, for information on using Clipboard.

Oracle Text

See the `$ORACLE_HOME/ctx/sample/api/index.html` file for information on the Oracle Text code samples.

See Also: *Oracle Text Reference* for more information on Oracle Text.

Oracle Spatial

See the *Oracle Spatial User's Guide and Reference* for information on Oracle Spatial.

F

Oracle Cluster Management Software for Linux

Oracle Cluster Management Software (OCMS) is available with Oracle9*i* on Linux systems. This appendix contains the following sections:

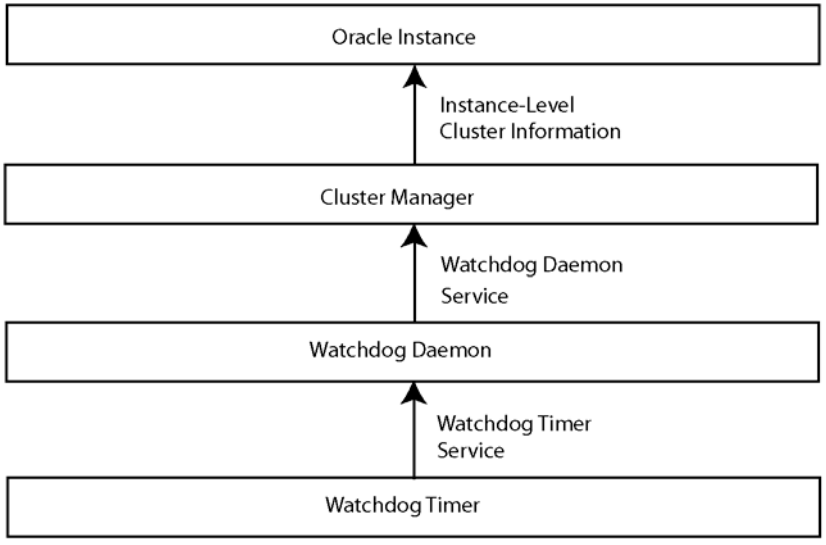
Overview

OCMS is included with the Oracle9*i* Enterprise Edition for Linux. It provides cluster membership services, a global view of clusters, node monitoring, and cluster reconfiguration. It is a component of Oracle9*i* Real Application Clusters on Linux and is installed automatically when you choose Oracle9*i* Real Application Clusters. OCMS consists of the following components:

- Watchdog Daemon
- Cluster Manager

Figure F–1 shows how the Watchdog daemon provides services to the Cluster Manager.

Figure F–1 Oracle Instance and Components of OCMS



Watchdog Daemon

The Watchdog daemon (`watchdogd`) uses a software-implemented Watchdog timer to monitor selected system resources to prevent database corruption.

Watchdog timer is a feature of the Linux kernel. The Watchdog daemon is part of Oracle9*i* Real Application Clusters.

The Watchdog daemon monitors the Cluster Manager and passes notifications to the Watchdog timer at defined intervals. The behavior of the Watchdog timer is partially controlled by the CONFIG_WATCHDOG_NOWAYOUT configuration parameter of the Linux kernel.

If you use Oracle9*i* Real Application Clusters, you must set the value of the CONFIG_WATCHDOG_NOWAYOUT configuration parameter to Y. If the Watchdog Timer detects an Oracle instance or Cluster Manager failure, it resets the instance to avoid possible database corruption.

For information on how to set the CONFIG_WATCHDOG_NOWAYOUT parameter, see the `/usr/src/linux/Documentation/configure.help` file in the Linux kernel source code. For more information on Watchdog devices, see the `/usr/src/linux/Documentation/watchdog.txt` file in the Linux kernel source code.

Cluster Manager

The Cluster Manager maintains the status of the nodes and the Oracle instances across the cluster. The Cluster Manager process runs on each node of the Real Applications Cluster. Each node has one Cluster Manager. The number of Oracle instances for each node is not limited by Oracle9*i* Real Application Clusters. The Cluster Manager uses the following communication channels between nodes:

- Private network
- Quorum partition on the shared disk

During normal cluster operations, the Cluster Managers on each node of the cluster communicate with each other through heartbeat messages sent over the private network. The quorum partition is used as an emergency communication channel if a heartbeat message fails. A heartbeat message can fail for the following reasons:

- The Cluster Manager terminates on a node
- The private network fails
- There is an abnormally heavy load on the node

The Cluster Manager uses the quorum partition to determine the reason for the failure. From each node, the Cluster Manager periodically updates the designated block on the quorum partition. Other nodes check the timestamp for each block. If the message from one of the nodes does not arrive, but the corresponding partition on the quorum has a current timestamp, the network path between this node and other nodes has failed.

Each Oracle instance registers with the local Cluster Manager. The Cluster Manager monitors the status of local Oracle instances and propagates this information to Cluster Managers on other nodes. If the Oracle instance fails on one of the nodes, the following events occur:

1. The Cluster Manager on the node with the failed Oracle instance informs the Watchdog daemon about the failure.
2. The Watchdog daemon requests the Watchdog timer to reset the failed node.
3. The Watchdog timer resets the node.
4. The Cluster Managers on the surviving nodes inform their local Oracle instances that the failed node is removed from the cluster.
5. Oracle instances in the surviving nodes start the Oracle9*i* Real Application Clusters reconfiguration procedure.

The nodes must reset if an Oracle instance fails. This ensures that:

- No physical I/O requests to the shared disks from the failed node occur after the Oracle instance fails.
- Surviving nodes can start the cluster reconfiguration procedure without corrupting the data on the shared disk.

> **See Also:** "Configuring Timing for Cluster Reconfiguration" on page F-9 and "Watchdog Daemon and Cluster Manager Starting Options" on page F-11 for more information on the Cluster Manager.

Starting OCMS

The following sections describe how to start OCMS:

- Starting the Watchdog Daemon
- Configuring the Cluster Manager
- Starting the Cluster Manager
- Configuring Timing for Cluster Reconfiguration

Note: Oracle Corporation supplies the `$ORACLE_HOME/oracm/bin/ocmstart.sh` sample startup script. Run this script as the `root` user. Make sure that the ORACLE_HOME and PATH environment variables are set as described in the O*racle9i Installation Guide Release 2 (9.2.0.1.0) for UNIX Systems*. After you are familiar with starting the Watchdog daemon and the Cluster Manager, you can use the script to automate the start-up process.

Starting the Watchdog Daemon

To start the Watchdog daemon, enter the following:

```
$ su root
# cd $ORALE_HOME/oracm/bin
# watchdogd
```

Note: Always start the Watchdog daemon as the `root` user.

The default Watchdog daemon log file is `$ORACLE_HOME/oracm/log/wdd.log`.

The Watchdog daemon does not have configuration files. Table F–1 describes the arguments that you can use when starting the Watchdog daemon.

Table F–1 Watchdogd Daemon Arguments

Argument	Valid Values	Default Value	Description
`-l` *number*	0 or 1	1	If the value is 0, no resources are registered for monitoring. This argument is used for debugging system configuration problems. If the value is 1, the Cluster Managers are registered for monitoring. Oracle Corporation recommends using this option for normal operations.
`-m` *number*	5000 to 180000 ms	5000	The Watchdog daemon expects to receive heartbeat messages from all clients (`oracm` threads) within the time specified by this value. If a client fails to send a heartbeat message within this time, the Watchdog daemon stops sending heartbeat message to the kernel Watchdog timer, causing the system to reset.
`-d` *string*		`/dev/watchdog`	Path of the device file for the Watchdog timer.
`-e` *string*		`$ORACLE_HOME/oracm /log/wdd.log`	Filename of the Watchdog daemon log file.

Configuring the Cluster Manager

You must create the `$ORACLE_HOME/oracm/admin/cmcfg.ora` Cluster Manager configuration file on each node of the cluster before starting OCMS. Include the following parameters in this file:

- PublicNodeNames
- PrivateNodeNames
- CmDiskFile
- WatchdogTimerMargin
- HostName

Before creating the `cmcfg.ora` file, verify that the `/etc/hosts` file on each node of the cluster has an entry for the public network (public name) and an entry for the private network (private name for each node). The private network is used by the Oracle9*i* Real Application Clusters internode communication. The CmDiskFile parameter defines the location of the Cluster Manager quorum partition. The CmDiskFile parameter on each node in a cluster must specify the same quorum partition.

The following example shows a `cmcfg.ora` file on the first node of a four node cluster:

```
PublicNodeNames=pubnode1 pubnode2 pubnode3 pubnode4
PrivateNodeNames=prinode1 prinode2 prinode3 prinode4
CmDiskFile=/dev/raw1
WatchdogTimerMargin=1000
HostName=prinode1
```

Table F–2 lists all of the configurable Cluster Manager parameters in the `cmcfg.ora` file.

Table F–2 Cluster Manager Parameters of the cmcfg.oraFile

Parameter	Valid Values	Default Value	Description
CmDiskFile	Directory path, up to 256 characters in length	No default value. You must set the value explicitly.	Specifies the pathname of the quorum partition.
MissCount	2 to 1000	5	Specifies the time that the Cluster Manager waits for a heartbeat from the remote node before declaring that node inactive. The time in seconds is determined by multiplying the value of the MissCount parameter by 3.
PublicNodeNames	List of host names, up to 4096 characters in length	No default value.	Specifies the list of all host names for the public network, separated by spaces. List host names in the same order on each node.
PrivateNodeNames	List of host names, up to 4096 characters in length	No default value.	Specifies the list of all host names for the private network, separated by spaces. List host names in the same order on each node.
HostName	A host name, up to 256 characters in length	No default value.	Specifies the local host name for the private network. Define this name in the `/etc/hosts` file.
ServiceName	A service, up to 256 characters in length	CMSrvr	Specifies the service name to be used for communication between Cluster Managers. If a Cluster Manager cannot find the service name in the `/etc/services` file, it uses the port specified by the ServicePort parameter. ServiceName is a fixed-value parameter in this release. Use the ServicePort parameter if you need to choose an alternative port for the Cluster Manager to use.

Table F–2 Cluster Manager Parameters of the cmcfg.oraFile (Cont.)

Parameter	Valid Values	Default Value	Description
ServicePort	Any valid port number	9998	Specifies the number of the port to be used for communication between cluster managers when the ServiceName parameter does not specify a service.
WatchdogTimerMargin	1000 to 180000ms	No default value	The same as the value of the soft_margin parameter specified at Linux `softdog` startup. The value of the soft_margin parameter is specified in seconds and the value of the WatchdogTimerMargin parameter is specified in milliseconds. This parameter is part of the formula that specifies the time between when the Cluster Manager on the local node detects an Oracle instance failure or join on any node and when it reports the cluster reconfiguration to the Oracle instance on the local node. See "Configuring Timing for Cluster Reconfiguration" on page F-9 for information on this formula.
WatchdogSafetyMargin	1000 to 180000ms	5000ms	Specifies the time between when the cluster manager detects a remote node failure and when the cluster reconfiguration is started. This parameter is part of the formula that specifies the time between when the Cluster Manager on the local node detects an Oracle instance failure or join on any node and when it reports the cluster reconfiguration to the Oracle instance on the local node. See "Configuring Timing for Cluster Reconfiguration" on page F-9 for information on this formula.

Starting the Cluster Manager

To start the Cluster Manager:

1. Confirm that the Watchdog daemon is running.
2. Confirm that the host names specified by the PublicNodeNames and PrivateNodeNames parameters in the `cmcfg.ora` file are listed in the `/etc/hosts` file.

3. As the `root` user, start the `oracm` process as a background process. Redirect any output to a log file. For example, enter the following:

```
$ su root
# cd $ORACLE_HOME/oracm/bin
# oracm </dev/null >$ORACLE_HOME/oracm/log/cm.out 2>&1 &
```

In the preceding example, all of the output messages and error messages are written to the `$ORACLE_HOME/oracm/log/cm.out` file.

The `oracm` process spawns multiple threads. To list all of the threads, enter the `ps -elf` command.

Table F–3 describes the arguments of the `oracm` executable.

Table F–3 Arguments for the oracm Executable

Argument	Values	Default Value	Description
/a:*action*	0,1	0	Specifies the action taken when the LMON process or another Oracle process that can write to the shared disk terminates abnormally. If `action` is set to 0 (the default), no action is taken. If `action` is set to 1, the Cluster Manager requests the Watchdog daemon to stop the node completely.
/l:*filename*	Any	`/$ORACLE_HOME/oracm/log/cm.log`	Specifies the pathname of the log file for the Cluster Manager. The maximum pathname length is 192 characters.
`/?`	None	None	Shows help for the arguments of the `oracm` executable. The Cluster Manager does not start if you specify this argument.
`/m`	Any	25000000	The size of the `oracm` log file in bytes.

Configuring Timing for Cluster Reconfiguration

To avoid database corruption when a node fails, there is a delay before the Oracle9*i* Real Application Clusters reconfiguration commences. Without this delay, simultaneous access of the same data block by the failed node and the node performing the recovery can cause database corruption. The length of the delay is defined by the sum of the following:

- Value of the WatchdogTimerMargin parameter

- Value of the WatchdogSafetyMargin parameter
- Value of the Watchdog daemon `-m` command-line argument

> **See also:** Table F–2 on page F-7 for more information on the WatchdogTimerMargin and WatchdogSafetyMargin parameters, and Table F–1 on page F-6 for more information on the Watchdog daemon `-m` command-line argument.

If you use the default values for the Linux kernel soft_margin and Cluster Manager parameters, the time between when the failure is detected and the start of the cluster reconfiguration is 70 seconds. For most workloads this time can be significantly reduced. The following example shows how to decrease the time of the reconfiguration delay from 70 seconds to 20 seconds:

- Set the value of WatchdogTimerMargin (soft_margin) parameter to 10 seconds.
- Leave the value of the WatchdogSafetyMargin parameter at the default value, 5000ms.
- Leave the value of the Watchdog daemon `-m` command-line argument at the default value, 5000ms.

To change the values of the WatchdogTimerMargin (soft_margin) and the WatchdogSafetyMargin:

1. Stop the Oracle instance.
2. Reload the `softdog` module with the new value of soft_margin. For example, enter:

   ```
   #/sbin/insmod softdog soft_margin=10
   ```

3. Change the value of the WatchdogTimerMargin in the `$ORACLE_HOME/oracm/admin/cmcfg.ora` file. For example, edit the following line:

   ```
   WatchdogTimerMargin=50000
   ```

4. Restart `watchdogd` with the `-m` command-line argument set to 5000.
5. Restart the `oracm` executable.
6. Restart the Oracle instance.

Watchdog Daemon and Cluster Manager Starting Options

OCMS supports node fencing by completely resetting the node if an Oracle instance fails and the Cluster Manager thread malfunctions. This approach guarantees that the database is not corrupted.

However, it is not always necessary to reset the node if an Oracle instance fails. If the Oracle instance uses synchronous I/O, a node reset is not required. In addition, in some cases where the Oracle instance uses asynchronous I/O, it is not necessary to reset the node, depending on how asynchronous I/O is implemented in the Linux kernel. For a list of certified Linux kernels that do not require node-reset, see the Oracle Technology Network Web site at the following URL:

```
http://otn.oracle.com
```

The `/a:action` flag in the following command defines OCMS behavior when an Oracle process fails:

```
$ oracm /a:[action]
```

In the preceding example, if the *`action`* argument is set to 0, the node does not reset.

By default, the `watchdog` daemon starts with the `-l 1` option and the `oracm` process starts with the `/a:0` option. With these default values, the node resets only if the `oracm` or `watchdogd` process terminates. It does not reset if an Oracle process that can write to the disk terminates. This is safe if you are using a certified Linux kernel that does not require node-reset.

In the preceding example, if the *`action`* argument is set to 1, the node resets if the `oracm` command, `watchdogd` daemon, or Oracle process that can write to the disk terminates. In these situations, a SHUTDOWN ABORT command on an Oracle instance resets the node and terminates all Oracle instances that are running on that node.

G

Optimal Flexible Architecture

This appendix contains information on the Optimal Flexible Architecture (OFA) standard. The OFA standard is a set of configuration guidelines created to ensure fast, reliable Oracle databases that require little maintenance. This appendix contains the following sections:

Optimal Flexible Architecture

Oracle Corporation recommends that you implement the OFA standard when you install and configure Oracle9*i*.

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 such as software and data backup, which are often vulnerable to data corruption
- Facilitate switching between 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 an OFA-Compliant Database

This section describes characteristics of a database that complies with the OFA standard.

File System Organization

The file system is organized to enable easy administration for issues such as:

- 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

In most cases, investment in new hardware is not required to take advantage of the OFA standard.

Safeguards Against Drive Failures

By distributing applications across more than one drive, drive failures affect as few applications as possible.

Distribution of Home Directories

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

- The collection of home directories
- The contents of an individual home directory

Integrity of Login Home Directories

You can 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

You can execute multiple versions of application software simultaneously, enabling you 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.

Separates Administrative Information for Each Database

The ability to separate administrative information on 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 datafiles are identifiable as such
- The association of datafile 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

I/O Loads Tuned Across All Drives

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

Additional Benefits of OFA for Oracle9*i* Real Application Clusters

For Oracle9*i* Real Application Clusters Installations:

- Administrative data is stored in a central location, accessible to all database administrators
- Administrative data for a particular instance can be identified by filename

Optimal Flexible Architecture Implemented on UNIX

This section describes the naming strategy recommended by the OFA standard.

Mount Points

This section describes the naming conventions for mount points.

Create Mount Points

An Oracle9*i* installation requires at least two mount points: one for the software and at least one for the database files. To implement the OFA recommendations fully, Oracle9*i* requires at least four mount points: one for the software and at least three for database files.

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 prevent I/O bottlenecks, use the syntax */pm/q/dm* for naming mount points. Table G–1 describes the variables used in this syntax.

Table G–1 Syntax for Naming Mount Points

Variable	Description
pm	A mount point name
q	A string denoting that Oracle data is stored in this directory
dm	The value of the initialization parameter DB_NAME (synonymous with the instance *sid* for single-instance databases)

For example, mount points named `/u01/oradata/test` and `/u02/oradata/test` allocate two drives for the Oracle test database.

Naming Directories

This section describes the naming conventions for OFA compliant directories.

Home Directory Syntax

Name home directories using the syntax */pm/h/u.* Table G–2 describes the variables used in this syntax.

Table G–2 Syntax for Naming Home Directories

Variable	Description
pm	A mount point name
h	A standard directory name
u	The name of the owner of the directory

For example, `/u01/app/oracle` is the Oracle software owner home directory (also referred to as the Oracle base directory, the default directory used 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 because it allows the collection of applications owner login home directories on different mount points to be referred to using the single pattern matching string, `/*/app/*`.

Referring to Pathnames

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

Software Directories

To help fulfill the OFA feature of simultaneously executing multiple versions of application software, store each version of the Oracle9*i* Server software in a directory matching the pattern */pm/h/u/*`product/`*v*.

Table G–3 describes the variables used in this syntax.

Table G–3 Syntax for Naming Oracle9i Server Software Directories

Variable	Description
pm	A mount point name
h	A standard directory name
u	The name of the owner of the directory
v	The version of the software

For example, `/u01/app/oracle/product/9.2.0.1.0` indicates the Oracle9*i* parent directory. Set the ORACLE_HOME environment variable to this directory.

Naming Subdirectories

To facilitate the organization of administrative data, Oracle Corporation recommends that you store database-specific administration files in subdirectories matching the pattern */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 database administration files. Table G–4 describes the database administration file subdirectories.

Table G–4 Subdirectories for Database Administration Files

Subdirectory	Description
`adhoc`	Ad hoc SQL scripts for a particular database
`arch`	Archived redo log files
`adump`	Audit files (Set the AUDIT_FILE_DEST initialization parameter to the `adump` directory. Clean out this subdirectory periodically.)

Table G–4 Subdirectories for Database Administration Files (Cont.)

Subdirectory	Description
bdump	Background process trace files
cdump	Core dump files
create	Programs used to create the database
exp	Database export files
logbook	Files recording the status and history of the database
pfile	Instance parameter files
udump	User SQL trace files

For example, the adhoc subdirectory has the pathname /u01/app/oracle/admin/sab/adhoc/ if the adhoc subdirectory is part of the database named sab.

Naming Database Files

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

File Type	File Naming Convention
Control files	/*pm*/*q*/*d*/control.ctl
Redo log files	/*pm*/*q*/*d*/redo*n*.log
Datafiles	/*pm*/*q*/*d*/*tn*.dbf

The following table describes this syntax:

Variable	Description
pm	A mount point name described previously in this chapter
q	A string distinguishing Oracle data from all other files (usually named ORACLE or oradata)
d	The value of the initialization parameter DB_NAME (synonymous with the instance *sid* for single-instance databases)
t	An Oracle tablespace name
n	A two-digit string

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

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

Table G–5 describes the special tablespaces that the Database Configuration Assistant creates for each Oracle database. If you manually create a database, you must create the required tablespaces. These tablespaces are in addition to those required for application segments.

See Also: *Oracle9i Database Administrator's Guide* for information on creating tablespaces manually.

Table G–5 Special Tablespaces

Tablespace	Required	Description
CWMLITE	No	OLAP catalog metadata repository (CWMLite)
DEMO	No	Demo schema
DRSYS	No	Oracle Text segment
INDX	No	Index associated with data in the USERS tablespace
OEM_REPOSITORY	No	Repository for Oracle Enterprise Manager
RBS	Yes	Rollback segments
SYSTEM	Yes	Data dictionary segments
TEMP	Yes	Temporary segments
USERS	No	Miscellaneous user segments
XDB	No	The XDB tablespace holds the data that is stored in the Oracle XML DB repository through SQL or through protocols such as HTTP and WebDAV.

Creating these special tablespaces is effective because data dictionary segments are never dropped, and no other segments that can be dropped are allowed in the SYSTEM tablespace. Doing 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 using complicated tools.

Naming Tablespaces

Name tablespaces descriptively using a maximum of eight characters. Although Oracle9*i* tablespace names can be 30 characters long, portable UNIX filenames are restricted to 14 characters. The recommended standard for a datafile 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 enable the datafile 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 on type.

Exploiting the OFA Structure for Oracle Files

Table G–6 describes the syntax used for identifying classes of files.

Table G–6 Directory Structure Syntax for Identifying Classes of Files

Directory Structure Syntax	Description
/u[0-9][0-9]	User data directories
/*/home/*	User home directories
/*/app/*	User application software directories
/*/app/applmgr	Oracle applications software subtrees
/*/app/oracle/product	Oracle software subtrees

Table G–6 Directory Structure Syntax for Identifying Classes of Files (Cont.)

Directory Structure Syntax	Description
/*/app/oracle/product/9.2.0.1.0	Oracle Server release 9.2.0.1.0 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 G–7 shows a hierarchical file mapping of a sample OFA-compliant database, including each file mount point, application, database, and tablespace. The filenames indicate the file type (control, log, or data).

Table G–7 Hierarchical File Mapping for OFA Installation

Directory	Description
/	Root mount point
/u01/	User data mount point 1
/u01/app/	Subtree for application software
/u01/app/oracle/	Home for `oracle` software user
/u01/app/oracle/admin/	Subtree for database administration files
/u01/app/oracle/admin/TAR	Subtree for support log files
/u01/app/oracle/admin/*db_name1*/	`admin` subtree for *db_name1* database
/u01/app/oracle/admin/*db_name2*/	`admin` subtree for *db_name2* database
/u01/app/oracle/doc/	Online documentation
/u01/app/oracle/product/	Distribution files
/u01/app/oracle/product/8.1.6/	Oracle home directory for release 8.1.6 instances
/u01/app/oracle/product/8.1.7/	Oracle home directory for release 8.1.7 instances
/u01/app/oracle/product/9.2.0.1.0	Oracle home directory for release 9.2.0.1.0 instances

Table G–7 Hierarchical File Mapping for OFA Installation (Cont.)

Directory	Description
/u01/app/ltb/	Home directory for a user
/u01/app/sbm/	Home directory for a user
/u01/oradata/	Subtree for Oracle data
/u01/oradata/*db_name1*/	Subtree for *db_name1* database files
/u01/oradata/*db_name2*/	Subtree for *db_name2* database files
/u02/	User data mount point 2
/u02/home/	Subtree for login home directories
/u02/home/cvm/	Home directory for a user
/u02/home/vrm/	Home directory for a user
/u02/oradata/	Subtree for Oracle data
/u02/oradata/*db_name1*/	Subtree for *db_name1* database files
/u02/oradata/*db_name2*/	Subtree for *db_name2* database files
/u03/	User data mount point 3
/u03/oradata/	Subtree for Oracle data
/u03/oradata/*db_name1*/	Subtree for *db_name1* database files
/u03/oradata/*db_name2*/	Subtree for *db_name2* database files

File Mapping for a Multiple-Instance OFA Database

When using the Oracle9*i* Real Application Clusters, 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 the `~/admin/d/` directory. Mount the `admin` directory for the administrative home as the `admin` directory for every instance. Table G–10 shows a sample directory structure.

Table G–8 Administrative Directory Structure for Dual-Instance Oracle9i Real Application Clusters

Directory Path	Description
/u01/app/oracle/admin/sab/	Administrative directory for sab database
/u01/adhoc/	Directory for miscellaneous scripts
/u01/arch/	Log archive destination for all instances
/u01/arch/redo001.arc	Archived redo log file
/u01/bdump/	Directory for background dump files
/u01/bdump/inst1/	Background dump destination for inst1 instance
/u01/bdump/inst2/	Background dump destination for inst2 instance
/u01/cdump/	Directory for core dump files
/u01/cdump/inst1/	Core dump destination for inst1 instance
/u01/cdump/inst2/	Core dump destination for inst2 instance
/u01/create/	Directory for creation scripts
/u01/create/1-rdbms.sql	SQL script to create *inst* database
/u01/exp/	Directory for exports
/u01/exp/20000120full.dmp	January 20, 2000 full export dump file
/u01/exp/export/	Directory for export parfiles
/u01/exp/import/	Directory for import parfiles
/u01/logbook/	Directory for logbook entries
/u01/logbook/inst1/	Directory for inst1 instance reports
/u01/logbook/inst1/params.1st	V$PARAMETER report for inst1 instances
/u01/logbook/inst2/	Directory for inst2 instances reports
/u01/logbook/inst2/params.1st	V$PARAMETER report for inst2 instances
/u01/logbook/user.1st	DBA_USERS report
/u01/pfile/	Directory for instance parameter files
/u01/pfile/inst1/	Directory for inst1 instance parameters

Table G–8 Administrative Directory Structure for Dual-Instance Oracle9i Real Application Clusters (Cont.)

Directory Path	Description
/u01/pfile/inst1/initinst1.ora	Instance parameters for inst1 instance
/u01/pfile/inst2/	Directory for inst2 instance parameters
/u01/pfile/inst2/initinst2.ora	Instance parameters for inst2 instance
/u01/udump/	Directory for user dump files
/u01/udump/inst1/	User dump destination for inst1 instance
/u01/udump/inst2/	User dump destination for inst2 instance

Directory Structure

The following sections describe the directory structure for OFA compliant installations.

ORACLE Base Directory

The Oracle base directory is the root of the Oracle directory structure. When installing an OFA-compliant database using the Oracle Universal Installer, the default Oracle base directory is set to */pm*/app/oracle**.** Table G–9 describes an Oracle base directory structure and content.

Table G–9 Oracle Base Directory Structure and Content

Directory	Description
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 */pm*/app/oracle/product/*release_number*. Table G–10 describes the Oracle home directory structure and content. Under UNIX, the Oracle home directory contains the subdirectories described in Table G–10, as well as a subdirectory for each Oracle product installed.

Table G–10 Oracle Home Directory Structure and Content

Directory	Description
assistants	Configuration Assistants
bin	Binaries for all products
ctx	Oracle Text files
dbs	Initialization files
install	Installation-related files
lib	Oracle product libraries
jlib	Java classes
md	Spatial options
mlx	Xerox Stemmer (for Oracle Text files)
network	Oracle Net Services files
ocommon	Common files for all products
odg	Data gatherer files
oracore	Core libraries
ord	Oracle *inter*Media files
otrace	Oracle TRACE files
plsql	PL / SQL files
precomp	Precompiler files
rdbms	Server files and libraries required for the database
slax	SLAX parser files
sqlplus	SQL*Plus files

Examples of Product Subdirectories

Table G–11 shows examples of product subdirectories and their contents.

Table G–11 Examples of Product Subdirectories

Directory	Description
rdbms	admin, doc, install, lib, log, mesg
sqlplus	admin, demo, doc, install, lib, mesg

Contents of Product Subdirectories

Table G–12 describes the subdirectories contained in the `rdbms` and `sqlplus` product subdirectories.

Table G–12 Contents of Product Subdirectories

Directory	Description
`admin`	Administrative SQL and shell scripts (for example, `catalog.sql`, `catexp.sql`, and `demo.sql`)
`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, `readmeunix.doc`)
`install`	Product installation scripts
`jlib`	Product Java classes
`lib`	Product libraries and distributed make files
`log`	Trace files and log files (for example, `orasrv.log` and `*.trc` files)
`mesg`	U.S. message files and binary files (for example, `oraus.msg` and `oraus.msb`)

File Naming Conventions in the admin Directory

Table G–13 shows the SQL scripts located in the `$ORACLE_HOME/rdbms/admin` directory.

Table G–13 admin Directory, File Naming Conventions

File	Description
`cat*.sql`	Creates catalog and data dictionary tables and views. The following files are run automatically during installation: ■ `catalog.sql` (for all installations) ■ `catproc.sql` (for all installations) ■ `catclust.sql` (for Oracle9*i* Real Application Clusters option installations) ■ `catrep.sql` (for all installations) The `catproc.sql` file in turn runs the scripts for creating the standard PL/SQL packages, such as DBMS_SQL and DBMS_OUTPUT.

Table G–13 admin Directory, File Naming Conventions (Cont.)

File	Description
d*.sql	Downgrade scripts
dbms*.sql	Additional database packages
u*.sql	Upgrade scripts
utl*.sql	Creates tables and views for database utilities

Filename Extensions

Table G–14 describes filename extensions.

Table G–14 Filename Extensions

Extension	Description
.a	Object file libraries; Ada runtime libraries
.aud	Oracle audit file
.bdf	X11 font description file
.bmp	X11 bitmap file
.c	C source file
.ctl	SQL*Loader control file; Oracle Server control file
.dat	SQL*Loader datafile
.dbf	Oracle Server tablespace file
.dmp	Export file
.doc	ASCII text file
.env	Shell script file for setting environment
.h	C header file; also, sr.h is a SQL*Report Writer help file
.jar	Java class archive
.l	UNIX manual page
.lis	Output of SQL*Plus script
.log	Installation log files; Oracle Server redo log file
.mk	Make file
.msb	Multilingual Option message file (binary)
.msg	Multilingual Option message file (text)
.o	Object module

Table G–14 Filename Extensions (Cont.)

Extension	Description
.ora	Oracle configuration file
.orc	Installation prototype file
.pc	Pro*C source file
.pco	Pro*COBOL source file
.ppd	Printer driver file
.sh	Bourne shell script file
.sql	SQL script file
.sys	Bourne shell script file
.tab	SQL script file
.trc	Trace file
.utd	Uniform Terminal Definition file
.zip	Zip file

Index

Symbols

A

B

C

D

E

F

G

H

I

J

L

M

N

O

P

R

S

T

U

V

W

X