

Oracle® Database

管理者リファレンス

10g リリース 1 (10.1) for UNIX Systems

部品番号 : B13705-04

2005 年 1 月

Oracle Database 管理者リファレンス , 10g リリース 1 (10.1) for UNIX Systems

部品番号 : B13705-04

原本名 : Oracle Database Administrator's Reference, 10g Release 1 (10.1) for UNIX Systems: AIX-Based Systems, Apple Mac OS X, hp HP-UX, hp Tru64 UNIX, Linux, and Solaris Operating System

原本部品番号 : B10812-06

Copyright © 1996, 2004, Oracle. All rights reserved.

制限付権利の説明

このプログラム（ソフトウェアおよびドキュメントを含む）には、オラクル社およびその関連会社に所有権のある情報が含まれています。このプログラムの使用または開示は、オラクル社およびその関連会社との契約に記された制約条件に従うものとします。著作権、特許権およびその他の知的財産権と工業所有権に関する法律により保護されています。

独立して作成された他のソフトウェアとの互換性を得るために必要な場合、もしくは法律によって規定される場合を除き、このプログラムのリバース・エンジニアリング、逆アセンブル、逆コンパイル等は禁止されています。

このドキュメントの情報は、予告なしに変更される場合があります。オラクル社およびその関連会社は、このドキュメントに誤りが無いことの保証は致し兼ねます。これらのプログラムのライセンス契約で許諾されている場合を除き、プログラムを形式、手段（電子的または機械的）、目的に関係なく、複製または転用することはできません。

このプログラムが米国政府機関、もしくは米国政府機関に代わってこのプログラムをライセンスまたは使用する者に提供される場合は、次の注意が適用されます。

U.S. GOVERNMENT RIGHTS

Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the Programs, including documentation and technical data, shall be subject to the licensing restrictions set forth in the applicable Oracle license agreement, and, to the extent applicable, the additional rights set forth in FAR 52.227-19, Commercial Computer Software--Restricted Rights (June 1987). Oracle Corporation, 500 Oracle Parkway, Redwood City, CA 94065.

このプログラムは、核、航空産業、大量輸送、医療あるいはその他の危険が伴うアプリケーションへの用途を目的としておりません。このプログラムをかかる目的で使用する際、上述のアプリケーションを安全に使用するために、適切な安全装置、バックアップ、冗長性（**redundancy**）、その他の対策を講じることは使用者の責任となります。万一かかるプログラムの使用に起因して損害が発生いたしましても、オラクル社およびその関連会社は一切責任を負いかねます。

Oracle は Oracle Corporation およびその関連会社の登録商標です。その他の名称は、Oracle Corporation または各社が所有する商標または登録商標です。

このプログラムは、第三者の Web サイトへリンクし、第三者のコンテンツ、製品、サービスへアクセスすることがあります。オラクル社およびその関連会社は第三者の Web サイトで提供されるコンテンツについては、一切の責任を負いかねます。当該コンテンツの利用は、お客様の責任になります。第三者の製品またはサービスを購入する場合は、第三者と直接の取引となります。オラクル社およびその関連会社は、第三者の製品およびサービスの品質、契約の履行（製品またはサービスの提供、保証義務を含む）に関しては責任を負いかねます。また、第三者との取引により損失や損害が発生いたしましても、オラクル社およびその関連会社は一切の責任を負いかねます。

目次

はじめに	ix
対象読者	x
用語	x
表記規則	x
コマンド構文	xi
関連ドキュメント	xi
サード・パーティ・ソフトウェア情報	xii
1 UNIX システムでの Oracle の管理	
概要	1-2
環境変数	1-2
Oracle Database の環境変数	1-2
UNIX 環境変数	1-4
共通の環境設定	1-6
oraenv および coraenv スクリプト・ファイル	1-6
ローカル bin ディレクトリ	1-7
システム・タイム・ゾーンの設定	1-7
初期化パラメータ	1-7
DB_BLOCK_SIZE 初期化パラメータの最大値	1-7
ASM_DISKSTRING 初期化パラメータのデフォルト値	1-8
LOG_ARCHIVE_DEST_ <i>n</i> 初期化パラメータの ASYNC の最大値	1-8
CLUSTER_INTERCONNECTS 初期化パラメータ	1-8
フェイルオーバーおよびフェイルバックと CLUSTER_INTERCONNECTS	1-9
オペレーティング・システムのアカунとグループ	1-10
Oracle ソフトウェア所有者アカウンと	1-10
OSDBA、OSOPER および Oracle インベントリ・グループ	1-10
グループとセキュリティ	1-11
外部認証	1-11
orapwd ユーティリティの実行	1-11
orapwd ユーティリティおよび Oracle Real Application Clusters	1-12
パスワード管理	1-12
UNIX アカウンとの追加	1-12
Oracle ユーザーのアカウンとの構成	1-12

RAW デバイスの使用	1-13
RAW デバイスを使用する場合のガイドライン	1-13
RAW デバイスの設定	1-14
AIX または Tru64 UNIX システムでの RAW デバイスのデータファイル	1-14
トレース・ファイルおよびアラート・ファイルの使用	1-15
トレース・ファイル	1-15
アラート・ファイル	1-15
 2 Oracle ソフトウェアの起動と停止	
Oracle プロセスの停止と起動	2-2
Mac OS X での Oracle プロセスの起動	2-2
データベース・インスタンスまたは Automatic Storage Management インスタンス	2-2
Oracle Net Listener	2-4
iSQL*Plus	2-5
Oracle Ultra Search	2-5
Oracle Enterprise Manager Database Control	2-6
Oracle Management Agent	2-7
起動と停止の自動化	2-8
Mac OS X におけるデータベースの起動と停止の自動化	2-8
その他のオペレーティング・システムにおけるデータベースの起動と停止の自動化	2-10
 3 UNIX システムでの Oracle 製品の構成	
Oracle 製品の追加に対するデータベースの構成	3-2
スタンドアロン・ツールとしての Configuration Assistant の使用	3-2
Oracle Net Configuration Assistant の使用	3-2
Database Upgrade Assistant の使用	3-3
Database Configuration Assistant の使用	3-3
新規またはアップグレード済データベースの構成	3-3
実行可能ファイルの再リンク	3-4
 4 SQL*Plus の管理	
コマンドライン SQL*Plus の管理	4-2
設定ファイルの使用	4-2
サイト・プロファイル・ファイルの使用	4-2
ユーザー・プロファイル・ファイルの使用	4-2
PRODUCT_USER_PROFILE 表の使用	4-2
デモンストレーション表の使用	4-3
SQL*Plus のコマンドライン・ヘルプ	4-3
SQL*Plus のコマンドライン・ヘルプのインストール	4-3
SQL*Plus のコマンドライン・ヘルプの削除	4-3
コマンドライン SQL*Plus の使用	4-4
SQL*Plus からのシステム・エディタの使用	4-4
SQL*Plus からのオペレーティング・システム・コマンドの実行	4-4
SQL*Plus への割込み	4-4
SPOOL コマンドの使用	4-5

SQL*Plus の制限事項	4-5
ウィンドウのサイズ変更	4-5
リターン・コード	4-5
パスワードの非表示	4-5

5 Oracle Net Services の構成

Oracle Net Services 構成ファイルの保存場所	5-2
adapters ユーティリティ	5-3
Oracle protocol support	5-4
IPC プロトコル・サポート	5-5
TCP/IP プロトコル・サポート	5-5
SSL 付き TCP/IP プロトコル・サポート	5-6
TCP/IP または SSL 付き TCP/IP 用のリスナーの設定	5-6
Oracle Advanced Security	5-6

6 Oracle プリコンパイラおよび Oracle Call Interface の使用

Oracle プリコンパイラの概要	6-2
プリコンパイラ構成ファイル	6-2
プリコンパイラ実行可能ファイルの再リンク	6-2
プリコンパイラの README ファイル	6-3
すべてのプリコンパイラに共通の問題	6-3
大文字から小文字への変換	6-3
ベンダー提供のデバッガ・プログラム	6-3
IRECLEN および ORECLEN の値	6-3
静的および動的リンク	6-3
クライアント共有ライブラリとクライアント静的ライブラリ	6-4
クライアント・アプリケーションのビット長サポート	6-5
Pro*C/C++ プリコンパイラ	6-6
Pro*C/C++ のデモ・プログラム	6-6
Pro*C/C++ のユーザー・プログラム	6-7
Pro*COBOL プリコンパイラ	6-8
Pro*COBOL の環境変数	6-9
Acucorp ACUCOBOL-GT COBOL コンパイラ	6-9
Micro Focus Server Express COBOL コンパイラ	6-10
Pro*COBOL の Oracle ランタイム・システム	6-11
Pro*COBOL のデモ・プログラム	6-11
Pro*COBOL のユーザー・プログラム	6-12
FORMAT プリコンパイラ・オプション	6-13
Pro*FORTRAN プリコンパイラ	6-13
Pro*FORTRAN のデモ・プログラム	6-13
Pro*FORTRAN のユーザー・プログラム	6-14
AIX の場合のみ: SQL*Module for Ada	6-15
SQL*Module for Ada のデモ・プログラム	6-15
SQL*Module for Ada のユーザー・プログラム	6-16
Oracle Call Interface と Oracle C++ Call Interface	6-16
OCI と OCCI のデモ・プログラム	6-16
OCI と OCCI のユーザー・プログラム	6-17

64 ビット・ドライバを使用する Oracle JDBC/OCI プログラム (AIX、HP-UX、Solaris SPARC および zSeries Linux のみ)	6-18
カスタム Make ファイル	6-19
未定義シンボルの修正	6-19
マルチスレッド・アプリケーション	6-20
シグナル・ハンドラの使用	6-20
XA 機能	6-22

7 SQL*Loader および PL/SQL のデモ

SQL*Loader のデモ	7-2
PL/SQL のデモ	7-2
PL/SQL からの 32 ビット外部プロシージャのコール	7-5

8 UNIX での Oracle Database のチューニング

チューニングの重要性	8-2
オペレーティング・システムのツール	8-2
共通のツール	8-2
AIX のツール	8-5
AIX System Management Interface Tool	8-5
Base Operation System ツール	8-5
AIX Performance Toolbox	8-5
HP-UX のツール	8-6
パフォーマンス・チューニング・ツール	8-6
HP-UX パフォーマンス分析ツール	8-7
Linux のツール	8-7
Mac OS X のツール	8-7
Solaris のツール	8-7
メモリー管理のチューニング	8-8
十分なスワップ領域の割当て	8-8
ページングの制御	8-9
Oracle ブロック・サイズの調整	8-9
ディスク I/O のチューニング	8-10
Automatic Storage Management の使用	8-10
適切なファイル・システム・タイプの選択	8-10
ディスク・パフォーマンスの監視	8-11
システム・グローバル領域	8-12
SGA サイズの確認	8-13
AIX での共有メモリー	8-13
オペレーティング・システムのバッファ・キャッシュのチューニング	8-14

A AIX システムでの Oracle Database の管理

メモリーとページング	A-2
バッファ・キャッシュのページング・アクティビティの制御	A-2
AIX ファイル・バッファ・キャッシュのチューニング	A-2
minperm および maxperm パラメータのチューニング	A-3
十分なページング領域の割当て (スワップ領域)	A-3
ページングの制御	A-4
データベース・ブロック・サイズの設定	A-4

ログ・アーカイブ・バッファのチューニング	A-5
I/O バッファおよび SQL*Loader	A-5
インポート・ユーティリティ用の BUFFER パラメータ	A-5
ディスク I/O の問題	A-5
AIX 論理ボリューム・マネージャ	A-5
ストライプ化された論理ボリュームの設計	A-6
その他の考慮事項	A-6
ジャーナル・ファイル・システムを使用した場合と RAW 論理ボリュームを使用した場合の相違	A-6
ファイル・システム・オプション	A-7
ジャーナル・ファイル・システムから RAW 論理ボリュームへの移動	A-8
RAW 論理ボリュームからジャーナル・ファイル・システムへの移動	A-8
非同期 I/O の使用	A-9
I/O スレーブ	A-10
DB_FILE_MULTIBLOCK_READ_COUNT パラメータの使用	A-11
後書きの使用	A-11
順次先読みのチューニング	A-11
ディスク I/O の歩調合せのチューニング	A-12
リモート I/O 操作の最小化	A-12
Oracle Database でのミラー復元	A-13
RAW デバイスのバックアップ	A-13
CPU のスケジューリングおよびプロセスの優先順位	A-14
プロセスのランタイム・スライスの変更	A-14
SMP システムでのプロセッサ・バインディングの使用	A-14
Oracle Real Application Clusters に関する情報	A-14
UDP のチューニング	A-14
透過的アプリケーション・フェイルオーバーのためのネットワーク・チューニング	A-15
Oracle Real Application Clusters および HACMP または PSSP	A-15
Oracle Real Application Clusters とフォルト・トレラント IPC	A-16
AIXTHREAD_SCOPE 環境変数の設定	A-16

B HP-UX システムでの Oracle Database の管理

Oracle インスタンス用の HP-UX 共有メモリー・セグメント	B-2
HP-UX SCHED_NOAGE スケジューリング・ポリシー	B-2
Oracle Database での SCHED_NOAGE の有効化	B-3
軽量タイマーの実装	B-3
非同期 I/O	B-4
MLOCK 権限	B-4
非同期 I/O の実装	B-4
非同期 I/O の検証	B-6
HP-UX 非同期ドライバが Oracle Database に対して設定されているかどうかの検証	B-6
Oracle Database が非同期 I/O を使用しているかどうかの検証	B-6
SGA の非同期フラグ	B-7
大規模メモリーの割当てと Oracle Database のチューニング	B-7
永続的な専用 SQL 領域とメモリー	B-7
デフォルトの大規模仮想メモリー・ページ・サイズ	B-8
チューニングに関する推奨事項	B-9

HP-UX での Oracle Real Application Clusters	B-9
PA-RISC の場合のみ: Hyper Messaging Protocol パラメータのチューニング	B-9
CPU_COUNT 初期化パラメータおよび HP-UX 動的プロセッサ再構成	B-10

C Linux システムでの Oracle Database の管理

x86 の場合のみ: 拡張バッファ・キャッシュのサポート	C-2
x86 の場合のみ: Red Hat Enterprise Linux AS 2.1 および SUSE Linux Enterprise Server 8 での ラージ・ページの有効化	C-3
Red Hat Enterprise Linux AS 2.1 (Itanium) または SUSE Linux Enterprise Server 9 での hugetlbfs の 使用	C-4
Red Hat Enterprise Linux AS 3 での hugetlbfs の使用	C-5
Linux x86 の場合のみ: SGA アドレス空間の増加	C-6
非同期 I/O サポート	C-7
ダイレクト I/O サポート	C-8
semtimedop のサポート	C-8
x86 の場合のみ: 高速ネットワークのサポート	C-8

D Mac OS X での Oracle Database の管理

使用可能および使用済のスワップ領域の決定	D-2
----------------------------	-----

E Tru64 UNIX システムでの Oracle Database の管理

Oracle Database 指定配置最適化の有効化	E-2
指定配置最適化を実行するための要件	E-2
Oracle 指定配置最適化の有効化	E-2
Oracle 指定配置最適化の無効化	E-3
Oracle 指定配置最適化の使用	E-3
Oracle 初期化パラメータ	E-3
Tru64 UNIX サブシステム属性	E-3
RAD に対するプロセスの親和性	E-4
システムに存在する RAD 数のサブセットへの Oracle Database の実行制限	E-4
複合 CPU システムのサポート	E-5
Tru64 UNIX でのデータベース統計の収集	E-5
Tru64 UNIX での Oracle Real Application Clusters	E-6
Reliable Data Gram	E-6
要件	E-6
UDP IPC の有効化	E-6
Oracle Real Application Clusters と NUMA の同時有効化	E-7
非同期 I/O のチューニング	E-7
aio_task_max_num 属性	E-7
ダイレクト I/O サポートおよびコンカレント・ダイレクト I/O サポート	E-8
単一インスタンスの要件	E-8
クラスタ化されたシステム	E-8
Tru64 UNIX V5.1B クラスタ化システム	E-8
複数インスタンスの要件 (Oracle Real Application Clusters)	E-9
ダイレクト I/O サポートの無効化	E-9
リアルタイム・クロックへのアクセスの有効化	E-10
RAW デバイスの設定	E-11

Spike 最適化ツール	E-13
Spike の使用	E-13
F Solaris システムでの Oracle Database の管理	
緊密共有メモリー	F-2
G データベースの制限	
データベースの制限	G-2
索引	

はじめに

このマニュアルでは、UNIX システムで Oracle Database 10g リリース 1 (10.1) を管理および構成する方法について説明します。

対象読者

このマニュアルは、UNIX システムで Oracle Database 10g リリース 1 (10.1) の管理および構成を担当する方を対象としています。Oracle Real Application Clusters を UNIX クラスタ上に構成している場合は、このマニュアルおよび『Oracle Real Application Clusters インストレーションおよび構成』でインストールおよび構成の詳細を参照してください。

用語

このマニュアルでは UNIX オペレーティング・システムの名前を次のように短縮して使用しています。

オペレーティング・システム	短縮名
AIX-Based Systems	AIX
Apple Mac OS X	Mac OS X
HP-UX PA-RISC (64-bit) HP-UX Itanium	HP-UX 注意: 特定のアーキテクチャにおける HP-UX の情報の違いについては、本文中に記載されています。
HP Tru64 UNIX	Tru64 UNIX
Linux x86 Linux Itanium IBM zSeries Based Linux	Linux 注意: 特定のアーキテクチャにおける Linux の情報の違いについては、本文中に記載されています。
Solaris Operating System (SPARC) Solaris Operating System (x86)	Solaris 注意: 特定のアーキテクチャにおける Solaris の情報の違いについては、本文中に記載されています。

表記規則

このマニュアルでは、次の表記規則を使用しています。

規則	説明
固定幅フォント	固定幅フォントは、UNIX コマンド、ディレクトリ名、ユーザー名、パス名およびファイル名を示します。
イタリック体	イタリック体は、ファイル名の可変部分などの変数を示します。
大文字	大文字は、Structured Query Language (SQL) の予約語、初期化パラメータおよび環境変数を示します。

コマンド構文

UNIX のコマンド構文は、固定幅フォントで表示されます。ドル記号 (\$)、シャープ記号 (#) またはパーセント記号 (%) は、UNIX のコマンド・プロンプトです。これらの記号をコマンドの一部として入力しないでください。このマニュアルでは、コマンド構文に次の表記規則を使用しています。

規則	説明
バックスラッシュ \ 中カッコ { } 大カッコ [] 省略記号 ... イタリック体 縦線	バックスラッシュは、UNIX コマンドの行の継続を表す記号です。コマンド例が 1 行に入りきらない場合に使用します。コマンドは、表示どおりにバックスラッシュを付けて入力するか、またはバックスラッシュなしで 1 行に入力します。 <pre>dd if=/dev/rdisk/c0t1d0s6 of=/dev/rst0 bs=10b \ count=10000</pre> 中カッコは、必須の入力項目を表します。 <pre>.DEFINE {macro1}</pre> 大カッコは、カッコ内の項目を任意に選択することを表します。 <pre>cvtcrt termname [outfile]</pre> 省略記号は、同じ項目を任意の数だけ繰り返すことを表します。 <pre>CHKVAL fieldname value1 value2 ... valueN</pre> イタリック体は、変数を表します。変数には値を代入します。 <pre>library_name</pre> 縦線は、大カッコまたは中カッコ内の複数の選択項目の区切りに使用します。 <pre>FILE filesize [K M]</pre>

関連ドキュメント

Oracle Database 10g 製品に関するプラットフォーム固有のドキュメントには、次のマニュアルが含まれます。

- Oracle Database
 - 『Oracle Database リリース・ノート』（プラットフォーム固有）
 - 『Oracle Database クイック・インストール・ガイド』（プラットフォーム固有）
 - 『Oracle Database インストール・ガイド for UNIX Systems』
 - 『Oracle Database インストール・ガイド for Apple Mac OS X』
 - 『Oracle Real Application Clusters インストールおよび構成』
 - 『Oracle Database 管理者リファレンス for UNIX Systems』
 - 『Oracle Procedural Gateway for APPC Installation and Configuration Guide for UNIX』
 - 『Oracle Procedural Gateway for APPC User's Guide for UNIX』
 - 『Oracle Procedural Gateway for APPC Messages Guide』
 - 『Oracle Transparent Gateway for DRDA Installation and User's Guide for UNIX』
- Oracle Client
 - 『Oracle Database クライアント・クイック・インストール・ガイド』（プラットフォーム固有）
 - 『Oracle Database クライアント・インストール・ガイド for UNIX Systems』
 - 『Oracle Database クライアント・インストール・ガイド for Apple Mac OS X』

- Oracle Database 10g Companion CD
 - 『Oracle Database Companion CD インストレーション・ガイド for UNIX Systems』
 - 『Oracle Database Companion CD インストレーション・ガイド for Apple Mac OS X』
 - 『Oracle Database Companion CD クイック・インストレーション・ガイド』
(プラットフォーム固有)

このマニュアルのリリース時に記載されなかった重要な情報は、使用しているプラットフォームに関する Oracle Database リリース・ノートを参照してください。Oracle Database 10g のリリース・ノートは定期的に更新されます。最新バージョンは次の URL で OTN-J からダウンロードできます。

<http://otn.oracle.co.jp/document/>

サード・パーティ・ソフトウェア情報

このプログラムには、HP 社のサード・パーティ・ソフトウェアが含まれています。Oracle プログラム (HP 社のソフトウェアを含む) を使用する権利は、この製品に付随する Oracle プログラム・ライセンスによって許諾されます。これと異なる規定が Oracle プログラム・ライセンス内にある場合でも、HP 社のソフトウェアは現状のままであり、この規定によっていかなる種類の知的財産権保護、保証またはサポートもオラクル社または HP 社から提供されることはありません。

このプログラムには、IBM 社のサード・パーティ・ソフトウェアが含まれています。Oracle プログラム (IBM 社のソフトウェアを含む) を使用する権利は、この製品に付随する Oracle プログラム・ライセンスによって許諾されます。

これと異なる規定が Oracle プログラム・ライセンス内にある場合でも、IBM 社のソフトウェアは現状のままであり、この規定によっていかなる種類の知的財産権保護、保証またはサポートもオラクル社または IBM 社から提供されることはありません。

UNIX システムでの Oracle の管理

この章では、UNIX システムで Oracle Database を管理する方法について説明します。次の項目について説明します。

- [概要](#)
- [環境変数](#)
- [初期化パラメータ](#)
- [オペレーティング・システムのアカウントとグループ](#)
- [RAW デバイスの使用](#)
- [トレース・ファイルおよびアラート・ファイルの使用](#)

概要

Oracle Database を使用するには、Oracle Database の環境変数、パラメータおよびユーザー設定を設定する必要があります。この章では、UNIX で動作する Oracle Database の各種設定について説明します。

Oracle Database のファイルおよびプログラムでは、疑問符 (?) は、環境変数 ORACLE_HOME の値を表します。たとえば、Oracle Database では、次の SQL 文中の疑問符は、Oracle ホーム・ディレクトリのフルパス名に展開されます。

```
SQL> ALTER TABLESPACE TEMP ADD DATAFILE '?/dbs/temp02.dbf' SIZE 200M
```

同様に、アットマーク (@) 記号は環境変数 ORACLE_SID を表します。たとえば、ファイルが現行のインスタンスに属していることを指定する場合は、次のように入力します。

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

環境変数

この項では、通常使用される Oracle Database および UNIX の環境変数について説明します。Oracle Database をインストールする前に、これらの環境変数をいくつか定義する必要があります。これらの環境変数は、『Oracle Database インストレーション・ガイド』に記載されています。

環境変数の現在の設定値を表示するには、env コマンドを使用します。たとえば、環境変数 ORACLE_SID の値を表示するには、次のように入力します。

```
$ env | grep ORACLE_SID
```

すべての環境変数の現在の設定値を表示するには、env コマンドを次のように使用します。

```
$ env | more
```

Oracle Database の環境変数

表 1-1 に、Oracle Database で使用する環境変数に必要な構文および例を示します。

表 1-1 UNIX での Oracle Database 環境変数

変数	項目	定義
NLS_LANG	機能	クライアント環境の言語、地域およびキャラクタ・セットを指定します。 NLS_LANG で指定するキャラクタ・セットは、端末または端末エミュレータのキャラクタ・セットと一致している必要があります。NLS_LANG で指定されたキャラクタ・セットがデータベースのキャラクタ・セットと異なる場合、そのキャラクタ・セットは自動的に変換されます。 この変数の値リストの詳細は、『Oracle Database グローバリゼーション・サポート・ガイド』を参照してください。
	構文	<code>language_territory.characterset</code>
	例	<code>french_france.we8dec</code>
ORA_NLS10	機能	言語、地域、キャラクタ・セットおよび言語の定義ファイルが保存されているディレクトリを指定します。
	構文	<code>directory_path</code>
	例	<code>\$ORACLE_HOME/nls/data</code>

表 1-1 UNIX での Oracle Database 環境変数 (続き)

変数	項目	定義
ORA_TZFILE	機能	タイム・ゾーン・ファイルのフルパスおよびファイル名を指定します。この環境変数は、データベース内のデータに対して小規模タイム・ゾーン・ファイル (\$ORACLE_HOME/oracore/zoneinfo/timzone.dat) を使用する場合に設定する必要があります。Oracle Database 10g では、デフォルトで大規模タイム・ゾーン・ファイル (\$ORACLE_HOME/oracore/zoneinfo/timzlg.dat) が使用されます。このファイルには、小規模タイム・ゾーン・ファイルよりも多数のタイム・ゾーンに関する情報が含まれています。 情報を共有するデータベースはすべて、同一のタイム・ゾーン・ファイルを使用する必要があります。この環境変数の値を変更した場合は、データベースを停止し、再起動する必要があります。
	構文	<code>directory_path</code>
	例	<code>\$ORACLE_HOME/oracore/zoneinfo/timzlg.dat</code>
ORACLE_BASE	機能	Optimal Flexible Architecture (OFA) に準拠したインストールのディレクトリ構造の基本となるディレクトリを指定します。
	構文	<code>directory_path</code>
	例	<code>/u01/app/oracle</code>
ORACLE_HOME	機能	Oracle ソフトウェアが格納されているディレクトリを指定します。
	構文	<code>directory_path</code>
	例	<code>\$ORACLE_BASE/product/10.1.0/db_1</code>
ORACLE_PATH	機能	SQL*Plus などの Oracle アプリケーションが使用するファイルの検索パスを指定します。ファイルのフルパス名が指定されていない場合やファイルが現行のディレクトリにない場合、Oracle アプリケーションでは、ORACLE_PATH を使用してそのファイルの場所を特定します。
	構文	ディレクトリをコロンで区切ったリスト: <code>directory1:directory2:directory3</code>
	例	<code>/u01/app/oracle/product/10.1.0/db_1/bin:.</code> 注意: 最後にピリオドを付けることによって、検索パスに現行のディレクトリが追加されます。
ORACLE_SID	機能	Oracle システムの識別子を指定します。
	構文	英字で始まり、数字と英字で構成される文字列を指定します。システム識別子は、8 文字以内で指定することをお勧めします。この環境変数の詳細は、『Oracle Database インストレーション・ガイド』を参照してください。
	例	<code>SAL1</code>
ORACLE_TRACE	機能	インストール時のシェル・スクリプトのトレースを有効にします。この環境変数を T に設定した場合は、ほとんどの Oracle シェル・スクリプトで <code>set -x</code> コマンドが使用されます。これによって、シェル・スクリプトの実行時にコマンドとそれらの引数が印刷されます。T 以外の値を設定した場合、または値を設定しない場合、そのスクリプトでは、 <code>set -x</code> コマンドが使用されません。
	構文	T または T 以外
ORAENV_ASK	機能	oraenv または coraenv スクリプトで、環境変数 ORACLE_SID の値を入力するためのプロンプトを表示するかどうかを制御します。NO を設定した場合、環境変数 ORACLE_SID の値を入力するためのプロンプトは表示されません。NO 以外の値を設定した場合、または値を設定しない場合は、環境変数 ORACLE_SID の値を入力するためのプロンプトが表示されます。
	構文	NO または NO 以外
	例	NO
SQLPATH	機能	SQL*Plus での login.sql ファイルの検索先ディレクトリまたはディレクトリのリストを指定します。
	構文	ディレクトリをコロンで区切ったリスト: <code>directory1:directory2:directory3</code>
	例	<code>/home:/home/oracle:/u01/oracle</code>

表 1-1 UNIX での Oracle Database 環境変数（続き）

変数	項目	定義
TNS_ADMIN	機能	Oracle Net Services 構成ファイルが格納されているディレクトリを指定します。
	構文	<i>directory_path</i>
	例	\$ORACLE_HOME/network/admin
TWO_TASK	機能	接続文字列に使用するデフォルトの接続識別子を指定します。この環境変数が設定されている場合は、接続識別子を接続文字列に指定する必要はありません。たとえば、環境変数 TWO_TASK が sales に設定されている場合は、CONNECT <i>username/password@sales</i> コマンドではなく、CONNECT <i>username/password</i> コマンドを使用してデータベースに接続できます。
	構文	任意の接続識別子
	許容値	ネーミング・メソッドを使用して解決できる有効な接続識別子（tnsnames.ora ファイルやディレクトリ・サーバーなど）
	例	PRODDb_TCP

注意： 競合を防ぐため、Oracle サーバーのプロセス名と同じ名前環境変数を定義しないでください。Oracle サーバーのプロセス名には、ARCH、PMON および DBWR などがあります。

UNIX 環境変数

表 1-2 に、Oracle Database で使用する UNIX 環境変数に必要な構文および例を示します。

表 1-2 Oracle Database で使用する UNIX 環境変数

変数	項目	定義
ADA_PATH (AIX のみ)	機能	Ada コンパイラが格納されているディレクトリを指定します。
	構文	<i>directory_path</i>
	例	/usr/lpp/powerada
CLASSPATH	機能	Java アプリケーションで使います。この環境変数に必要な設定は、Java アプリケーションによって異なります。詳細は、使用している Java アプリケーション製品のドキュメントを参照してください。
	構文	ディレクトリまたはファイルをコロンで区切ったリスト： <i>directory1:directory2:file1:file2</i>
	例	デフォルトの設定はありません。CLASSPATH には、次のディレクトリが含まれている必要があります。 \$ORACLE_HOME/JRE/lib:\$ORACLE_HOME/jlib
DISPLAY	機能	X ベースのツールで使います。入出力に使用するディスプレイ・デバイスを指定します。詳細は、使用している X Window System のドキュメントを参照してください。
	構文	<i>hostname:server[.screen]</i> <i>hostname</i> はシステム名（IP アドレスまたは別名）、 <i>server</i> はサーバーの順次コード番号、 <i>screen</i> は画面の順次コード番号です。モニターが 1 つの場合は、サーバーと画面のどちらにも値 0 を使用します（0.0）。 注意： モニターが 1 つの場合、 <i>screen</i> はオプションです。
	例	135.287.222.12:0.0 bambi:0
DYLD_LIBRARY_PATH (Mac OS X のみ)	機能	共有ライブラリ・ローダーの実行時に共有オブジェクト・ライブラリの検索先となるディレクトリのリストを指定します。この環境変数の詳細は、dyld の man ページを参照してください。
	構文	ディレクトリをコロンで区切ったリスト： <i>directory1:directory2:directory3</i>
	例	/usr/lib:\$ORACLE_HOME/lib

表 1-2 Oracle Database で使用する UNIX 環境変数 (続き)

変数	項目	定義
HOME	機能	ユーザーのホーム・ディレクトリを指定します。
	構文	<code>directory_path</code>
	例	<code>/home/oracle</code>
LANG または LANGUAGE	機能	メッセージなどの出力でオペレーティング・システムが使用する言語およびキャラクタ・セットを指定します。詳細は、オペレーティング・システムのドキュメントを参照してください。 注意: これらの環境変数は Mac OS X では使用されません。
LD_OPTIONS	機能	デフォルトのリンカー・オプションを指定します。この環境変数の詳細は、ld の <code>man</code> ページを参照してください。
LPDEST (Solaris のみ)	機能	デフォルトのプリンタの名前を指定します。
	構文	<code>string</code>
	例	<code>docprinter</code>
LD_LIBRARY_PATH (AIX および Mac OS X を 除くすべてのプラット フォーム。HP-UX の場合 は、64 ビット共有ライブ ラリのパスを指定)	機能	共有ライブラリ・ローダーの実行時に共有オブジェクト・ライブラリの検索先となるディレクトリのリストを指定します。この環境変数の詳細は、ld の <code>man</code> ページを参照してください。
	構文	ディレクトリをコロンで区切ったリスト: <code>directory1:directory2:directory3</code>
	例	<code>/usr/dt/lib:\$ORACLE_HOME/lib</code>
LD_LIBRARY_PATH_64 (Solaris システム、SPARC システムのみ)	機能	共有ライブラリ・ローダーの実行時に特定の 64 ビットの共有オブジェクト・ライブラリを検索するディレクトリのリストを指定します。この環境変数の詳細は、ld の <code>man</code> ページを参照してください。
	構文	コロンで区切ったディレクトリのリスト: <code>directory1:directory2:directory3</code>
	例	<code>/usr/dt/lib:\$ORACLE_HOME/lib64</code>
LIBPATH (AIX のみ)	機能	共有ライブラリ・ローダーの実行時に共有オブジェクト・ライブラリの検索先となるディレクトリのリストを指定します。この環境変数の詳細は、ld の <code>man</code> ページを参照してください。
	構文	ディレクトリをコロンで区切ったリスト: <code>directory1:directory2:directory3</code>
	例	<code>/usr/dt/lib:\$ORACLE_HOME/lib</code>
PATH	機能	シェルで、実行可能プログラムの場所を特定するために使用されます。 <code>\$ORACLE_HOME/bin</code> ディレクトリが含まれている必要があります。
	構文	ディレクトリをコロンで区切ったリスト: <code>directory1:directory2:directory3</code>
	例	<code>/bin:/usr/bin:/usr/local/bin:/usr/bin/X11:\$ORACLE_HOME/bin:\$HOME/bin:.</code> 注意: 最後にピリオドを付けることによって、検索パスに現行のディレクトリが追加されます。
PRINTER	機能	デフォルトのプリンタの名前を指定します。
	構文	<code>string</code>
	例	<code>docprinter</code>
SHLIB_PATH (HP-UX の 32 ビットのライブラリの み)	機能	共有ライブラリ・ローダーの実行時に共有オブジェクト・ライブラリの検索先となるディレクトリのリストを指定します。この環境変数の詳細は、ld の <code>man</code> ページを参照してください。
	構文	ディレクトリをコロンで区切ったリスト: <code>directory1: directory2: directory3</code>
	例	<code>/usr/dt/lib:\$ORACLE_HOME/lib32</code>

表 1-2 Oracle Database で使用する UNIX 環境変数（続き）

変数	項目	定義
TEMP、TMP および TMPDIR	機能	テンポラリ・ファイル用のデフォルト・ディレクトリを指定します。設定すると、テンポラリ・ファイルを作成するツールは、指定したデフォルト・ディレクトリの 1 つにテンポラリ・ファイルを作成します。
	構文	<code>directory_path</code>
	例	<code>/u02/oracle/tmp</code>
XENVIRONMENT	機能	X Window System のリソース定義を含むファイルを指定します。詳細は、使用している X Window System のドキュメントを参照してください。

共通の環境設定

この項では、デフォルト・シェ尔に応じて `oraenv` または `coraenv` スクリプトを使用し、共通の UNIX 環境を設定する方法について説明します。

- Bourne、Bash または Korn シェ尔の場合は、`oraenv` コマンドを使用します。
- C シェ尔の場合は、`coraenv` コマンドを使用します。

oraenv および coraenv スクリプト・ファイル

`oraenv` および `coraenv` スクリプトは、インストール時に作成されます。この 2 つのスクリプトは、`oratab` ファイルの内容に基づいて環境変数を設定し、次の機能を提供します。

- データベースの変更をすべてのユーザー・アカウントに反映して更新するための主な方法
- `oratab` ファイルに指定されているデータベース間で切替えを行うためのメカニズム

開発システムからデータベースに対して頻繁に追加や削除を行ったり、同一システム上にインストールされた複数の異なる Oracle データベース間でユーザーが切替えを行う場合があります。`oraenv` または `coraenv` スクリプトを使用すると、ユーザー・アカウントが更新されていることを確認し、データベース間で切替えを行うことができます。

注意： `oraenv` または `coraenv` スクリプトは、Oracle ソフトウェア所有者（通常は `oracle`）ユーザーのシェ尔起動スクリプトからはコールしないでください。これらのスクリプトでは値の入力を促すプロンプトが表示されるため、システムの起動時に `dbstart` スクリプトが自動的にデータベースを起動できなくなります。

`oraenv` または `coraenv` スクリプトは通常、ユーザーのシェ尔起動ファイル（`.profile` または `.login` など）からコールされます。このスクリプトは、環境変数 `ORACLE_SID` および `ORACLE_HOME` を設定し、`$ORACLE_HOME/bin` ディレクトリを環境変数 `PATH` の設定に含めます。データベース間で切替えを行う場合に、`oraenv` または `coraenv` スクリプトを実行して、これらの環境変数を設定できます。

注意： このいずれかのスクリプトを実行するには、適切なコマンドを入力します。

- `coraenv` スクリプトの場合

```
% source /usr/local/bin/coraenv
```
 - `oraenv` スクリプトの場合

```
$ . /usr/local/bin/oraenv
```
-

ローカル bin ディレクトリ

oraenv、coraenv および dbhome スクリプトを含むディレクトリは、ローカル bin ディレクトリと呼ばれます。すべてのデータベース・ユーザーは、このディレクトリへの読取りアクセス権が必要です。ローカル bin ディレクトリのパスをユーザーの環境変数 PATH の設定に追加してください。インストール後に root.sh スクリプトを実行すると、ローカル bin ディレクトリのパスを要求するプロンプトが表示されます。指定したディレクトリに、oraenv、coraenv および dbhome スクリプトが自動的にコピーされます。デフォルトのローカル bin ディレクトリは、/usr/local/bin です。root.sh スクリプトを実行しない場合は、手動でも、oraenv または coraenv スクリプトと dbhome スクリプトを、\$ORACLE_HOME/bin ディレクトリからローカル bin ディレクトリにコピーできます。

システム・タイム・ゾーンの設定

環境変数 TZ には、タイム・ゾーンを設定します。これによって、時間を夏時間に変更したり、別のタイム・ゾーンにすることができます。調整した時刻は、タイムスタンプ・ファイルに使用したり、date コマンドの出力を生成したり、現在の SYSDATE の値を取得するときに使用します。

個人用の TZ 値は変更しないことをお勧めします。GMT+24 などの異なる TZ 値を使用すると、トランザクションが記録される日付が変更される場合があります。日付が変更されると、SYSDATE を使用する Oracle アプリケーションが影響を受けます。この問題を回避するために、表の順序付けには、日付列ではなく順序番号を使用してください。

初期化パラメータ

次の各項では、プラットフォーム固有の Oracle Database 初期化パラメータについて説明します。

- [DB_BLOCK_SIZE](#) 初期化パラメータの最大値
- [ASM_DISKSTRING](#) 初期化パラメータのデフォルト値
- [LOG_ARCHIVE_DEST_n](#) 初期化パラメータの ASYNC の最大値
- [CLUSTER_INTERCONNECTS](#) 初期化パラメータ

DB_BLOCK_SIZE 初期化パラメータの最大値

DB_BLOCK_SIZE 初期化パラメータは、データベースの標準ブロック・サイズを指定します。このブロック・サイズは SYSTEM 表領域に使用され、その他の表領域ではデフォルトで使用されます。

DB_BLOCK_SIZE パラメータに設定できる最大値は、次の表に示すように、UNIX プラットフォームによって異なります。

注意： データベースの作成後は、DB_BLOCK_SIZE パラメータの値を変更できません。

プラットフォーム	最大値
Linux x86、Mac OS X および Solaris x86	16KB
その他のオペレーティング・システム	32KB

ASM_DISKSTRING 初期化パラメータのデフォルト値

表 1-3 に、Automatic Storage Management の ASM_DISKSTRING 初期化パラメータに対するプラットフォーム固有のデフォルト値を示します。

注意： ASM_DISKSTRING 初期化パラメータは、ASM インスタンスでのみサポートされます。

表 1-3 ASM_DISKSTRING 初期化パラメータのデフォルト値

プラットフォーム	デフォルト検索文字列
AIX	/dev/rhdisk*
HP-UX	/dev/rdisk/*
Linux	/dev/raw/*
Mac OS X	/dev/rdisk*s*s1
Solaris	/dev/rdisk/*
Tru64 UNIX	/dev/rdisk/*

LOG_ARCHIVE_DEST_n 初期化パラメータの ASYNC の最大値

LOG_ARCHIVE_DEST_n 初期化パラメータの ASYNC に設定できる最大値は、次の表に示すように、UNIX のプラットフォームによって異なります。

プラットフォーム	最大値
zSeries Linux	12800
HP-UX および Tru64 UNIX	51200
その他のオペレーティング・システム	102400

CLUSTER_INTERCONNECTS 初期化パラメータ

Real Application Clusters 環境では、CLUSTER_INTERCONNECTS 初期化パラメータを使用して、プライベート・ネットワークに対して代替インターコネクトを指定できます。

CLUSTER_INTERCONNECTS パラメータには、デバイス名ではなく、インターコネクトの IP アドレスが必要です。このパラメータでは、複数の IP アドレスをコロンで区切って指定できます。Oracle Real Application Clusters のネットワーク通信量は、指定されたすべての IP アドレス間で分散されます。

CLUSTER_INTERCONNECTS パラメータは、UDP IPC が使用可能になっている Oracle Real Application Clusters 環境にのみ有効です。これらを使用可能にすると、Oracle のグローバル・キャッシュ・サービス (GCS)、グローバル・エンキュー・サービス (GES) およびインタープロセッサ・パラレル問合せ (IPQ) などのすべての IPC 通信にインターコネクトを指定できます。

CLUSTER_INTERCONNECTS パラメータを設定して、Oracle GCS、GES および IPQ を強制的に別のインターコネクトに分散すると、クラスタ全体の安定性とパフォーマンスが向上する場合があります。たとえば、すべての GCS、GES および IPQ IPC 通信に対して、IP アドレスが 129.34.137.212 であるネットワーク・インタフェースを使用するには、CLUSTER_INTERCONNECTS パラメータを次のように設定します。

```
CLUSTER_INTERCONNECTS=129.34.137.212
```

デバイスの IP アドレスを表示するには、`ifconfig` または `netstat` コマンドを使用します。このコマンドを実行すると、デバイス名と IP アドレスとのマップが表示されます。たとえば、Tru64 UNIX 上のデバイスの IP アドレスを確認するには、次のコマンドを root ユーザーで入力します。

```
# /usr/sbin/ifconfig -a
fta0: flags=c63<UP,BROADCAST,NOTRAILERS,RUNNING,MULTICAST,SIMPLEX>
      inet 129.34.137.212 netmask fffffffc00 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

ics0:  flags=1100063<UP,BROADCAST,NOTRAILERS,RUNNING,NOCHECKSUM,CLUIF>
      inet 10.0.0.1 netmask ffffffff00 broadcast 10.0.0.255 ipmtu 7000

sl0:   flags=10<POINTOPOINT>

tun0: flags=80<NOARP>
```

この例では、インタフェース `fta0:` の IP アドレスは `129.34.137.212` で、インタフェース `ics0:` の IP アドレスは `10.0.0.1` です。

`CLUSTER_INTERCONNECTS` 初期化パラメータを使用するときは、次の重要事項を覚えておいてください。

- あるデータベースの複数のインスタンスが異なるノード上にある場合、それぞれのインスタンスに指定する IP アドレスは、同じネットワークに接続するネットワーク・アダプタに属している必要があります。この規則に従わない場合は、ノード間通信がブリッジやルーターを通過し、2つのノード間でパスが認識されないことがあります。
- `CLUSTER_INTERCONNECTS` パラメータは、データベース・インスタンスごとに異なる値をパラメータ・ファイルに指定します。
- このパラメータに複数の IP アドレスを指定する場合は、同じデータベースのインスタンスすべてに対し、それぞれの IP アドレスを、すべて同じ順序で並べて指定します。たとえば、Tru64 UNIX でノード 1 にあるインスタンス 1 のパラメータに `alt0:`、`fta0:` および `ics0:` の各デバイスの IP アドレスをこの順序で並べて指定した場合、ノード 2 にあるインスタンス 2 のパラメータも、同等のネットワーク・アダプタの IP アドレスを同じ順序で並べて指定する必要があります。
- 指定したインターコネクットの IP アドレスが正しくなかったり、システムに存在しない場合、Oracle Database では、デフォルトのクラスタ・インターコネクット・デバイスが使用されます。たとえば、Tru64 UNIX では、デフォルトのデバイスは `ics0:` です。

フェイルオーバーおよびフェイルバックと `CLUSTER_INTERCONNECTS`

一部のオペレーティング・システムは、実行時フェイルオーバーおよびフェイルバックをサポートしています。ただし、`CLUSTER_INTERCONNECTS` 初期化パラメータを使用する場合、フェイルオーバーおよびフェイルバックは使用不可になります。

関連項目： AIX システムでの実行時フェイルオーバーおよびフェイルバックの詳細は、[付録 A「AIX システムでの Oracle Database の管理」](#)を参照してください。

オペレーティング・システムのアカウントとグループ

Oracle Database では、次のような特有のオペレーティング・システム・アカウントとグループが必要です。

- Oracle ソフトウェア所有者アカウント
- OSDBA、OSOPER および Oracle インベントリ・グループ

Oracle ソフトウェア所有者アカウント

Oracle ソフトウェア所有者アカウントは、通常 `oracle` という名前で、Oracle ソフトウェアのインストールに使用します。ソフトウェアを個別に Oracle ホーム・ディレクトリにインストールするたびに、異なる Oracle ソフトウェア所有者アカウントを使用できます。ただし、インストール後の Oracle ホーム・ディレクトリの保守作業では、Oracle ホーム・ディレクトリごとに、ソフトウェアをインストールしたときと同じアカウントを使用する必要があります。

Oracle ソフトウェア所有者は、Oracle インベントリ・グループをプライマリ・グループとして、OSDBA グループをセカンダリ・グループとして所有することをお勧めします。

OSDBA、OSOPER および Oracle インベントリ・グループ

表 1-4 に、Oracle Database に必要な、特殊な UNIX グループを示します。

表 1-4 UNIX グループ

グループ	代表的な名前	説明
OSDBA	<code>dba</code>	OSDBA グループのメンバーであるオペレーティング・システム・アカウントには、特殊なデータベース権限があります。このグループのメンバーは、SYSDBA 権限を使用してデータベースに接続できます。Oracle ソフトウェア所有者は、このグループの必須メンバーです。必要に応じて、他のアカウントを追加できます。
OSOPER	<code>oper</code>	OSOPER グループは、オプション・グループです。OSOPER グループのメンバーであるオペレーティング・システム・アカウントには、特殊なデータベース権限があります。このグループのメンバーは、SYSOPER 権限を使用してデータベースに接続できます。
Oracle インベントリ	<code>oinstall</code>	UNIX システムで Oracle ソフトウェアをインストールする全ユーザーは、Oracle インベントリ・グループと呼ばれる同一の UNIX グループに属している必要があります。このグループは、インストール時に Oracle ソフトウェア所有者のプライマリ・グループにする必要があります。インストール後、このグループはシステムにインストールされたすべての Oracle ファイルを所有します。

関連項目： OSDBA グループと SYSDBA 権限、および OSOPER グループと SYSOPER 権限の詳細は、『Oracle Database 管理者ガイド』および『Oracle Database インストレーション・ガイド』を参照してください。

Oracle Database では、UNIX オペレーティング・システムの機能をいくつか使用して、ユーザーに安全性の高い環境を提供します。その機能には、ファイル所有権、グループ・アカウント、および実行時にそのユーザー ID を変更するプログラム機能が含まれます。

Oracle Database の 2 タスク構造によって、ユーザー・プログラムと `oracle` プログラム間で作業（およびアドレス領域）を分割し、セキュリティを高めることができます。すべてのデータベース・アクセスは、このシャドウ・プロセスおよび `oracle` プログラムへの特殊権限によって行うことができます。

関連項目： セキュリティ問題の詳細は、『Oracle Database 管理者ガイド』を参照してください。

グループとセキュリティ

Oracle プログラムは、セキュリティの観点から、すべてのユーザー（UNIX 用語では `other`）が実行できるプログラムと、DBA のみが実行できるプログラムに分けられます。セキュリティを保護するために次の方法で分類することをお勧めします。

- `oracle` アカウントのプライマリ・グループは、`oinstall` である必要があります。
- `oracle` アカウントは、`dba` グループをセカンダリ・グループとする必要があります。
- SYSDBA 権限を必要とするユーザー・アカウントは、`dba` グループに属することができますが、`oinstall` グループに属することができるのは、`oracle` などの Oracle ソフトウェア所有者アカウントのみです。

外部認証

外部認証を使用する場合は、`OS_AUTHENT_PREFIX` 初期化パラメータの値を Oracle ユーザー名の接頭辞として使用する必要があります。このパラメータを明示的に設定しない場合は、UNIX のデフォルト値 `ops$`（大 / 小文字区別あり）が使用されます。

オペレーティング・システムと Oracle 認証の両方に同一のユーザー名を使用するには、次のように、この初期化パラメータを `NULL` 文字列に設定します。

```
OS_AUTHENT_PREFIX=""
```

関連項目： 外部認証の詳細は、『Oracle Database 管理者ガイド』を参照してください。

orapwd ユーティリティの実行

パスワード・ファイルを使用して、データベースへの接続時に SYSDBA および SYSOPER 権限を使用できるユーザーを識別できます。Database Configuration Assistant (DBCA) を使用してデータベースを作成すると、新規データベースに対してパスワード・ファイルが作成されます。データベースを手動で作成する場合は、次のようにデータベースのパスワード・ファイルを作成してください。

1. Oracle ソフトウェア所有者でログインします。
2. 次のように、`orapwd` ユーティリティを使用してパスワード・ファイルを作成します。

```
$ $ORACLE_HOME/bin/orapwd file=filename password=password entries=max_users
```

次の表に、このコマンドで指定する必要がある値を示します。

値	説明
<code>filename</code>	パスワード情報が書き込まれているファイルの名前です。ファイル名は <code>orapwsid</code> で、フルパス名を指定する必要があります。その内容は暗号化されていて、ユーザーには読めません。パスワード・ファイルは通常、 <code>\$ORACLE_HOME/dbs</code> ディレクトリに作成されます。
<code>password</code>	SYS ユーザーのパスワードです。データベースに接続した後で、ALTER USER 文を使用して SYS ユーザーのパスワードを変更した場合は、データ・ディクショナリに格納されているパスワードとパスワード・ファイルに格納されているパスワードの両方が更新されます。このパラメータの入力は必須です。
<code>max_users</code>	パスワード・ファイルで許可されるエントリの最大数を設定します。これは、SYSDBA 権限または SYSOPER 権限により同時にデータベースに接続できる別々のユーザーの最大数です。

関連項目： `orapwd` ユーティリティの使用方法は、『Oracle Database 管理者ガイド』を参照してください。

orapwd ユーティリティおよび Oracle Real Application Clusters

データベース・ファイルの格納にファイル・システムを使用しており、Real Application Clusters の Oracle ホーム・ディレクトリがクラスタ・ファイル・システムに存在しない場合は、パスワード・ファイルが共有ファイルへのシンボリック・リンクになるように設定できません。この実装を選択すると、SQL を介してファイルに行った変更内容はすべてのインスタンス間で共有されます。選択しない場合は、SQL を実行するインスタンスを含むノード上のファイルのみが更新されるため、そのファイルをデータベースのインスタンスを実行する他のすべてのノードにコピーする必要があります。このため、`$ORACLE_HOME/dbs/orapwsid` を共有ファイルへのシンボリック・リンクになるように設定することをお勧めします。

パスワード管理

Database Configuration Assistant を使用してデータベースを作成する場合、ユーザーは SYS および SYSTEM アカウントのパスワードを変更する必要があります。デフォルトのパスワード `CHANGE_ON_INSTALL` および `MANAGER` は使用できません。

Database Configuration Assistant では、セキュリティ上の理由により、データベースの作成後にほとんどの Oracle ユーザー・アカウントがロックされます。ただし、SYS または SYSTEM の各アカウントはロックされません。ロックされたアカウントにログインする場合は、事前にそれらのロックを解除し、パスワードを変更する必要があります。

DBCA でパスワードを変更するには、Database Configuration Assistant の「サマリー」ウィンドウの「パスワード管理」ボタンをクリックします。

あるいは、SQL*Plus を使用して SYS としてデータベースに接続し、次のコマンドを入力してアカウントをロック解除し、パスワードを再設定します。

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

UNIX アカウントの追加

必要に応じて、UNIX アカウントを追加します。管理者権限を使用してデータベースに接続するには、ユーザーが OSDBA または OSOPER グループのメンバーである必要があります。

Oracle ユーザーのアカウントの構成

oracle ユーザーおよび Oracle ユーザーの UNIX アカウントの起動ファイルを更新し、環境ファイルに適切な環境変数を指定します。

Bourne、Bash または Korn シェルの場合は、環境変数を `.profile` ファイルに追加します (Red Hat Enterprise Linux および Mac OS X の Bash シェルの場合は、`.bash_profile` ファイルに追加します)。

C シェルの場合は、環境変数を `.login` ファイルに追加します。

注意： `oraenv` または `coraenv` スクリプトを使用すると、Oracle ユーザーのアカウントが更新されたことを確認できます。

RAW デバイスの使用

次の各項では、RAW デバイス（RAW パーティションまたは RAW ボリューム）の使用方法について説明します。

注意： RAW デバイスのチューニングの詳細は、次の付録を参照してください。

- [付録 A「AIX システムでの Oracle Database の管理」](#)
 - [付録 C「Linux システムでの Oracle Database の管理」](#)
 - [付録 E「Tru64 UNIX システムでの Oracle Database の管理」](#)
-

RAW デバイスを使用する場合のガイドライン

UNIX で RAW デバイス（RAW パーティションまたは RAW ボリューム）を使用する場合は、次のようなデメリットがあります。

- RAW デバイスでは、ファイル・サイズの書込み制限に関する問題を解決できない場合があります。

注意： 現在のファイル・サイズ制限を表示するには、次のコマンドを入力します。

- Bourne、Bash または Korn シェルの場合

```
$ ulimit -a
```

- C シェルの場合

```
% limit
```

- 小規模クライアント・システムでは、十分な大きさの RAW デバイスのパーティションまたはボリュームを使用できない場合があります。
- 特定のディスク・ドライブに I/O アクティビティが集中していて、Oracle データファイルを別のドライブに移動すればパフォーマンスが向上するような場合に、I/O アクティビティの少ないドライブに、受入れに十分なサイズのパーティションまたはボリュームがないことがあります。このため、RAW デバイスを使用している場合は、他のディスク・ドライブにファイルを移動できない場合があります。
- RAW デバイスは、ファイル・システムまたは Automatic Storage Management (ASM) ディスク・グループに格納されているデータファイルに比べて、管理が難しくなります。

RAW デバイスを使用するかどうかの決定には、次のことを考慮する必要があります。

- Oracle Real Application Clusters のインストール

Oracle Real Application Clusters (RAC) では、インスタンスごとに個別のログ・ファイルが使用されます。そのため、表領域および制御ファイル用のデバイスの他に、各インスタンスのログ・ファイル用に、2 つ以上のパーティションが必要です。これらのファイルはすべて、クラスタの全ノードで共有できるディスク上にある必要があります。

- RAW ディスク・パーティションの可用性

少なくとも Oracle データファイルと同数の RAW ディスク・パーティションがある場合のみ、Oracle ファイルに RAW パーティションを使用してください。ディスク領域を見直す場合で、RAW ディスク・パーティションがすでに作成済である場合は、データファイル・サイズとパーティション・サイズをできるだけ一致させ、無駄な領域をなくすようにします。

また、多数のディスク上の領域を一部のみ使用した場合と少数のディスク上の領域をすべて使用した場合の、パフォーマンスへの影響も考慮する必要があります。

- 論理ボリューム・マネージャ

論理ボリューム・マネージャは、論理レベルでディスク領域を管理するため、RAW デバイスの複雑さを意識する必要がありません。論理ボリュームを使用することによって、RAW パーティションの可用性に基づいて論理ディスクを作成できます。論理ボリューム・マネージャは、次の方法で固定ディスク・リソースを管理します。

- 論理記憶域と物理記憶域との間でデータをマッピングします。
- データを複数のディスクにまたがって不連続に格納、レプリケートおよび動的に拡張できるようにします。

RAC の場合、1 つの UNIX システムに関連付けられているドライブにも、UNIX クラスタ内の複数のシステムで共有できるドライブにも、論理ボリュームを使用できます。共有ドライブを使用すると、RAC データベースに関連付けられたすべてのファイルを、これらの共有論理ボリュームに配置できます。

- 動的パフォーマンスのチューニング

ディスク・パフォーマンスを最適化するには、アクティビティの高いディスク・ドライブからアクティビティの低いディスク・ドライブにファイルを移動します。論理ディスク機能を提供しているほとんどのハードウェア・ベンダーからも、チューニングに使用できるグラフィカル・ユーザー・インタフェースが提供されています。

- ミラー化およびオンラインでのディスクの交換

データ損害からデータを保護するために、論理ボリュームをミラー化できます。ミラー化された一方のデータに障害が起きた場合は、動的に再同期化できます。ベンダーによっては、ミラー化機能を使用してオンラインでドライブを交換する機能を提供している場合もあります。

RAW デバイスの設定

RAW デバイスを作成する場合は、次の項目に注意してください。

- デバイスの作成時に、所有者が Oracle ソフトウェア所有者ユーザー（oracle）であり、グループが OSDBA グループ（dba）であることを確認します。
- RAW パーティションに作成する Oracle データファイルのサイズは、RAW パーティションのサイズよりも Oracle ブロック・サイズ 2 つ分以上、小さくする必要があります。

関連項目： RAW デバイスの作成方法の詳細は、使用しているオペレーティング・システムのドキュメントを参照してください。

AIX または Tru64 UNIX システムでの RAW デバイスのデータファイル

AIX および Tru64 UNIX システムの場合、RAW 論理ボリューム上のデータファイルには最初の Oracle データ・ブロックにオフセットが存在する場合があります。このオフセットは、論理ボリューム・マネージャに必要です。

`$ORACLE_HOME/bin/offset` ユーティリティを使用すると、データファイルを異なるデバイスに転送する場合などに、オフセット値を指定できます。

トレース・ファイルおよびアラート・ファイルの使用

この項では、操作上の問題を簡単に診断して解決できるように、Oracle Database が作成するトレース・ファイル（ダンプ・ファイル）およびアラート・ファイルについて説明します。

トレース・ファイル

各サーバーとバックグラウンド・プロセスでは、トレース・ファイルに情報を書き込みます。プロセスが内部エラーを検出すると、そのエラーに関する情報がトレース・ファイルに書き込まれます。トレース・ファイルのファイル名の形式は `sid_processname_unixpid.trc` です。各項目の説明は次のとおりです。

- `sid` は、インスタンスのシステム識別子です。
- `processname` は、トレース・ファイルを作成した Oracle Database プロセスを識別するための、3～4 文字の短縮されたプロセス名（`pmon`、`dbwr`、`ora`、`reco` など）です。
- `unixpid` は、UNIX プロセスの ID 番号です。

トレース・ファイル名のサンプルは、`$ORACLE_BASE/admin/TEST/bdump/test_lgwr_1237.trc` です。

バックグラウンド・プロセス用のトレース・ファイルはすべて、`BACKGROUND_DUMP_DEST` 初期化パラメータで指定した接続先ディレクトリに書き込まれます。この初期化パラメータを設定しない場合は、デフォルトのディレクトリ `$ORACLE_HOME/rdbms/log` が使用されます。

ユーザー・プロセス用のトレース・ファイルはすべて、`USER_DUMP_DEST` 初期化パラメータで指定した接続先ディレクトリに書き込まれます。`USER_DUMP_DEST` 初期化パラメータを設定しない場合は、デフォルトのディレクトリ `$ORACLE_HOME/rdbms/log` が使用されます。`MAX_DUMP_FILE` 初期化パラメータを 5000 以上に設定して、トレース・ファイルがエラー情報を格納するのに十分な大きさになるようにします。

アラート・ファイル

`alert_sid.log` ファイルには、重要なデータベース・イベント情報やメッセージを格納します。データベース・インスタンスまたはデータベースに影響を与えるイベントは、このファイルに記録されます。このファイルは、データベースに関連付けられ、`BACKGROUND_DUMP_DEST` 初期化パラメータで指定したディレクトリに配置されます。この初期化パラメータを設定しない場合は、デフォルトのディレクトリ `$ORACLE_HOME/rdbms/log` が使用されます。

Oracle ソフトウェアの起動と停止

この章では、UNIX システム上で稼動している Oracle Database プロセスを識別する方法とプロセスを停止および再起動する基本的な方法について説明します。製品の管理の詳細は、該当する製品に固有のマニュアルを参照してください。この章では、次の項目について説明します。

- [Oracle プロセスの停止と起動](#)
- [起動と停止の自動化](#)

Oracle プロセスの停止と起動

この項では、次の Oracle 製品のプロセスを停止および起動する方法について説明します。

- [Mac OS X での Oracle プロセスの起動](#)
- [データベース・インスタンスまたは Automatic Storage Management インスタンス](#)
- [Oracle Net Listener](#)
- [iSQL*Plus](#)
- [Oracle Ultra Search](#)
- [Oracle Enterprise Manager Database Control](#)
- [Oracle Management Agent](#)

Mac OS X での Oracle プロセスの起動

注意： Oracle データベース・インスタンスまたは ASM インスタンス、あるいは Oracle Net Listener プロセスを起動する場合は、毎回必ずこの項の指示に従ってください。

特定のシェル制限が Oracle プロセスの実行に必要な値に設定されていることを確認するには、システムがローカル・システムである場合でも、ssh、rlogin または telnet コマンドを使用して、プロセスを起動するシステムに接続する必要があります。たとえば、次のように入力します。

```
$ ssh localhost
```

データベース・インスタンスまたは Automatic Storage Management インスタンス

この項では、データベースまたは Automatic Storage Management (ASM) インスタンスを停止および起動する方法について説明します。

データベースまたは ASM インスタンスの停止

Oracle Database または ASM インスタンスを停止するには、次の手順を実行します。

1. 必要に応じて、停止するインスタンスの SID および Oracle ホーム・ディレクトリを識別するには、次のコマンドを入力します。

- Solaris

```
$ cat /var/opt/oracle/oratab
```
- その他のオペレーティング・システム

```
$ cat /etc/oratab
```

oratab ファイルには、次のような行が含まれています。これによって、システム上の各データベース・インスタンスまたは ASM インスタンスの SID および対応する Oracle ホーム・ディレクトリを識別します。

```
sid:oracle_home_directory:[Y|N]
```

注意： ASM インスタンスの SID には、1 文字目にプラス記号 (+) を使用することをお勧めします。

注意： 記憶域管理に使用している Oracle データベース・インスタンスをすべて停止するまでは、ASM インスタンスを停止しないでください。

2. デフォルト・シェルに応じて oraenv または coraenv スクリプトを実行し、停止するインスタンスの環境変数を設定します。

- Bourne、Bash または Korn シェルの場合

```
$ . /usr/local/bin/oraenv
```

- C シェルの場合

```
% source /usr/local/bin/coraenv
```

プロンプトが表示されたら、インスタンスの SID を指定します。

3. インスタンスを停止するには、次のコマンドを入力します。

```
$ sqlplus /nolog
SQL> CONNECT SYS/sys_password as SYSDBA
SQL> SHUTDOWN NORMAL
```

インスタンスの停止後、SQL*Plus を終了できます。

データベースまたは ASM インスタンスの再起動

Oracle Database または Automatic Storage Management のインスタンスを再起動するには、次の手順を実行します。

1. 必要に応じて、前項の手順 1 と 2 を繰り返し、環境変数 ORACLE_SID および ORACLE_HOME を設定して、起動するインスタンスの SID および Oracle ホーム・ディレクトリを識別します。

注意： データベース・インスタンスで記憶域管理に ASM を使用している場合は、データベース・インスタンスを起動する前に、ASM インスタンスを起動する必要があります。

2. インスタンスを起動するには、次のコマンドを入力します。

```
$ sqlplus /nolog
SQL> CONNECT SYS/sys_password as SYSDBA
SQL> STARTUP
```

インスタンスの起動後、SQL*Plus を終了できます。

Oracle Net Listener

この項では、Oracle Net Listener を停止および起動する方法について説明します。

Oracle Net Listener の停止

Oracle Net Listener を停止するには、次の手順を実行します。

1. 停止する Oracle Net Listener のリスナー名とホーム・ディレクトリを判別するには、次のコマンドを入力します。

- Mac OS X:

```
$ ps -auxwww | grep tnslnsr
```

- その他のオペレーティング・システム

```
$ ps -ef | grep tnslnsr
```

このコマンドによって、次のような、システムで稼働中の Oracle Net Listener のリストが表示されます。 *listenername1* および *listenername2* はリスナー名です。

```
94248 ?? I 0:00.18 oracle_home1/bin/tnslnsr listenername1 -inherit
94248 ?? I 0:00.18 oracle_home2/bin/tnslnsr listenername2 -inherit
```

2. 環境変数 ORACLE_HOME を設定して、停止するリスナーの Oracle ホーム・ディレクトリを指定します。

- Bourne、Bash または Korn シェルの場合

```
$ ORACLE_HOME=oracle_home1
$ export ORACLE_HOME
```

- C シェルの場合

```
% setenv ORACLE_HOME oracle_home1
```

3. 次のコマンドを入力して、Oracle Net Listener を停止します。

```
$ $ORACLE_HOME/bin/lsnrctl stop listenername
```

注意： リスナー名がデフォルトの LISTENER の場合、このコマンドで名前を指定する必要はありません。

Oracle Net Listener の再起動

Oracle Net Listener を起動するには、次の手順を実行します。

1. 必要に応じて、起動するリスナーの Oracle ホーム・ディレクトリを指定するには、環境変数 ORACLE_HOME を設定します。

- Bourne、Bash または Korn シェルの場合

```
$ ORACLE_HOME=oracle_home1
$ export ORACLE_HOME
```

- C シェルの場合

```
% setenv ORACLE_HOME oracle_home1
```

2. 次のコマンドを入力して、Oracle Net Listener を再起動します。

```
$ $ORACLE_HOME/bin/lsnrctl start [listenername]
```

リスナー名の指定が必要になるのは、リスナー名がデフォルトのリスナー名 LISTENER と異なる場合のみです。リスナー名を確認するには、次のコマンドを入力して、*listener.ora* ファイルを表示します。

```
$ more $ORACLE_HOME/network/admin/listener.ora
```

iSQL*Plus

この項では、iSQL*Plus を停止および起動する方法について説明します。

iSQL*Plus の停止

iSQL*Plus を停止するには、次の手順を実行します。

1. 環境変数 ORACLE_HOME を設定して、iSQL*Plus の Oracle ホーム・ディレクトリを指定します。
 - Bourne、Bash または Korn シェルの場合


```
$ ORACLE_HOME=oracle_home
$ export ORACLE_HOME
```
 - C シェルの場合


```
% setenv ORACLE_HOME oracle_home
```
2. 次のコマンドを入力して、iSQL*Plus を停止します。


```
$ $ORACLE_HOME/bin/isqlplusctl stop
```

iSQL*Plus の起動

iSQL*Plus を起動するには、次の手順を実行します。

1. 必要に応じて、起動する iSQL*Plus インスタンスの Oracle ホーム・ディレクトリを指定するには、環境変数 ORACLE_HOME を設定します。
 - Bourne、Bash または Korn シェルの場合


```
$ ORACLE_HOME=oracle_home
$ export ORACLE_HOME
```
 - C シェルの場合


```
% setenv ORACLE_HOME oracle_home
```
2. 次のコマンドを入力して、iSQL*Plus を起動します。


```
$ $ORACLE_HOME/bin/isqlplusctl start
```

Oracle Ultra Search

この項では、Oracle Ultra Search を停止および起動する方法について説明します。

Oracle Ultra Search の停止

Oracle Ultra Search を停止するには、次の手順を実行します。

1. 必要に応じて、Oracle Ultra Search の Oracle ホーム・ディレクトリを指定するには、環境変数 ORACLE_HOME を設定します。
 - Bourne、Bash または Korn シェルの場合


```
$ ORACLE_HOME=oracle_home
$ export ORACLE_HOME
```
 - C シェルの場合


```
% setenv ORACLE_HOME oracle_home
```
2. 次のコマンドを入力して、Oracle Ultra Search を停止します。


```
$ $ORACLE_HOME/bin/searchctl stop
```

Oracle Ultra Search の起動

Oracle Ultra Search を起動するには、次の手順を実行します。

1. 必要に応じて、Oracle Ultra Search の Oracle ホーム・ディレクトリを指定するには、環境変数 ORACLE_HOME を設定します。

- Bourne、Bash または Korn シェルの場合

```
$ ORACLE_HOME=oracle_home
$ export ORACLE_HOME
```

- C シェルの場合

```
% setenv ORACLE_HOME oracle_home
```

2. 次のコマンドを入力して、Oracle Ultra Search を起動します。

```
$ $ORACLE_HOME/bin/searchctl start
```

Oracle Enterprise Manager Database Control

この項では、Oracle Enterprise Manager Database Control (Database Control) の起動方法と停止方法について説明します。

Database Control の停止

Database Control を停止するには、次の手順を実行します。

1. デフォルト・シェルに応じて oraenv または coraenv スクリプトを実行し、停止する Database Control で管理されているデータベースの環境を設定します。

- coraenv スクリプトの場合

```
% source /usr/local/bin/coraenv
```

- oraenv スクリプトの場合

```
$ . /usr/local/bin/oraenv
```

2. 次のコマンドを入力して、Database Control を停止します。

```
$ $ORACLE_HOME/bin/emctl stop dbconsole
```

Database Control の起動

Database Control を起動するには、次の手順を実行します。

1. 必要に応じて、環境変数 ORACLE_SID および ORACLE_HOME を設定し、起動する Database Control の SID および Oracle ホーム・ディレクトリを識別します。

- Bourne、Bash または Korn シェルの場合

```
$ ORACLE_HOME=oracle_home
$ ORACLE_SID=sid
$ export ORACLE_HOME ORACLE_SID
```

- C シェルの場合

```
% setenv ORACLE_HOME oracle_home
% setenv ORACLE_SID sid
```

2. 次のコマンドを入力して、Database Control を起動します。

```
$ $ORACLE_HOME/bin/emctl start dbconsole
```

Oracle Management Agent

Oracle Enterprise Manager Grid Control を使用して中央から複数の Oracle 製品を管理する場合、ホスト・システムごとにインストールされた Oracle Management Agent が必要です。通常、Oracle Management Agent は固有の Oracle ホーム・ディレクトリにインストールされます。

この項では、Oracle Management Agent を停止および起動する方法について説明します。

Oracle Management Agent の停止

Oracle Management Agent を停止するには、次の手順を実行します。

1. 必要に応じて、次のコマンドを入力して、Oracle Management Agent の Oracle ホーム・ディレクトリを判別します。

- Mac OS X:


```
$ ps -auxwww | grep emagent
```
- その他のオペレーティング・システム


```
$ ps -ef | grep emagent
```

このコマンドによって、Oracle Management Agent プロセスの情報が次のように表示されます。

```
94248 ?? I 0:00.18 oracle_home/agent/bin/emagent ...
```

2. 環境変数 ORACLE_HOME を設定して、Oracle Management Agent の Oracle ホーム・ディレクトリを指定します。

- Bourne、Bash または Korn シェルの場合


```
$ ORACLE_HOME=oracle_home
$ export ORACLE_HOME
```
- C シェルの場合


```
% setenv ORACLE_HOME oracle_home
```

3. 次のコマンドを入力して、Oracle Management Agent を停止します。

```
$ $ORACLE_HOME/agent/bin/emctl stop agent
```

Oracle Management Agent の起動

Oracle Management Agent を起動するには、次の手順を実行します。

1. 必要に応じて、環境変数 ORACLE_HOME を設定して、Oracle Management Agent の Oracle ホーム・ディレクトリを指定します。

- Bourne、Bash または Korn シェルの場合


```
$ ORACLE_HOME=oracle_home
$ export ORACLE_HOME
```
- C シェルの場合


```
% setenv ORACLE_HOME oracle_home
```

2. 次のコマンドを入力して、Oracle Management Agent を起動します。

```
$ $ORACLE_HOME/agent/bin/emctl start agent
```

起動と停止の自動化

注意： この項で説明する手順は、単一インスタンスのデータベースにのみ該当します。

Oracle データベースは、システムの起動時に自動的に起動し、停止時に自動的に停止するようにシステムを構成することをお薦めします。データベースの起動と停止を自動化することによって、データベースの不正な停止を防ぐことができます。

データベースの起動と停止を自動化するには、\$ORACLE_HOME/bin ディレクトリにある dbstart および dbshut スクリプトを使用します。これらのスクリプトは、oratab ファイル内の同じエントリを参照します。したがって、同じデータベース・セットに適用する必要があります。たとえば、dbstart スクリプトによって、sid1、sid2 および sid3 を自動的に起動し、dbshut スクリプトによって、sid1 のみを停止することはできません。ただし、dbstart スクリプトがまったく使用されていない場合は、dbshut スクリプトを使用してデータベース・セットの停止を指定することはできます。これを実行するには、システムの停止ファイルに dbshut エントリを追加し、システムの起動ファイルから dbstart エントリを除外します。

次の項では、Oracle Database の起動と停止を自動化する方法について説明します。

- [Mac OS X におけるデータベースの起動と停止の自動化](#)
- [その他のオペレーティング・システムにおけるデータベースの起動と停止の自動化](#)

Mac OS X におけるデータベースの起動と停止の自動化

Mac OS X で dbstart および dbshut スクリプトを使用してデータベースの起動と停止を自動化する手順は、次のとおりです。

1. root ユーザーでログインします。
2. テキスト・エディタで oratab ファイルを開きます。

```
# vi /etc/oratab
```

oratab ファイル内のデータベース・エントリは、次の形式で表示されます。

```
SID:ORACLE_HOME:{Y|N}
```

この例の Y または N は、スクリプトでデータベースの起動または停止を実行するかどうかを指定します。起動および停止するデータベースごとに、データベースのインスタンス識別子 (SID) を検索します。最初のフィールドの SID で識別し、最後のフィールドをそれぞれ Y に変更します。

注意： 新規データベース・インスタンスをシステムに追加する場合、これらのインスタンスを自動的に起動するには、必ず oratab ファイルでインスタンスのエントリを編集してください。

3. /Library/StartupItems/Oracle ディレクトリを作成し、ディレクトリをこれに変更します。

```
# mkdir /Library/StartupItems/Oracle
# cd /Library/StartupItems/Oracle
```

4. テキスト・エディタを使用して、このディレクトリに次のように Oracle という起動スクリプトを作成します。

```
#!/bin/bash

# source the common startup script

. /etc/rc.common

# Change the value of ORACLE_HOME to specify the correct Oracle home
# directory for you installation

ORACLE_HOME=/Volumes/u01/app/oracle/product/10.1.0/db_1
#
# change the value of ORACLE to the login name of the
# oracle owner at your site

ORACLE=oracle

PATH=${PATH}:${ORACLE_HOME}/bin
export ORACLE_HOME PATH

# Set shell limits for Oracle Database

ulimit -c unlimited
ulimit -d unlimited
ulimit -s 65536

StartService()
{
    if [ -f $ORACLE_HOME/bin/tnslsnr ] ; then
        ConsoleMessage "Starting Oracle Net"
        su $ORACLE -c "$ORACLE_HOME/bin/lsnrctl start"
    fi
    ConsoleMessage "Starting Oracle Databases"
    su $ORACLE -c "$ORACLE_HOME/bin/dbstart"
}

StopService()
{
    ConsoleMessage "Stopping Oracle Databases"
    su $ORACLE -c "$ORACLE_HOME/bin/dbshut"
    if [ -f $ORACLE_HOME/bin/tnslsnr ] ; then
        ConsoleMessage "Stopping Oracle Net"
        su $ORACLE -c "$ORACLE_HOME/bin/lsnrctl stop"
    fi
}

RestartService()
{
    StopService
    StartService
}

RunService "$1"
```

注意： このスクリプトは、Oracle Net Listener のパスワードが設定されていない場合にのみ有効です。リスナーにパスワードが設定されている場合、このスクリプトによってリスナーを停止することはできません。また、リスナー名がデフォルト名 **LISTENER** でない場合、停止および起動コマンドでリスナー名を指定する必要があります。

```
$ORACLE_HOME/bin/lsnrctl {start|stop} listener_name
```

5. テキスト・エディタを使用して、このディレクトリに次のように StartupParameters.plist という起動項目パラメータ・リスト・ファイルを作成します。

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple Computer//DTD PLIST 1.0//EN"
"http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
  <key>Description</key>
  <string>Oracle Database Startup</string>
  <key>Provides</key>
  <array>
    <string>Oracle Database</string>
  </array>
  <key>Requires</key>
  <array>
    <string>Disks</string>
  </array>
  <key>Uses</key>
  <array>
    <string>Disks</string>
    <string>Network</string>
  </array>
  <key>OrderPreference</key>
  <string>Late</string>
</dict>
</plist>
```

6. 作成したファイルの所有者、グループおよび権限を次のように変更します。

```
# chown root:wheel *
# chmod 700 *
```

その他のオペレーティング・システムにおけるデータベースの起動と停止の自動化

関連項目： システムの起動と停止の手順については、UNIX システムのドキュメントにある init コマンドを参照してください。

AIX、HP-UX、Linux、Solaris または Tru64 UNIX で dbstart および dbshut スクリプトを使用してデータベースの起動と停止を自動化する手順は、次のとおりです。

1. root ユーザーでログインします。
2. テキスト・エディタで oratab ファイルを開きます。

- Solaris

```
# vi /var/opt/oracle/oratab
```

- その他のオペレーティング・システム

```
# vi /etc/oratab
```

oratab ファイル内のデータベース・エントリは、次の形式で表示されます。

```
SID:ORACLE_HOME:{Y|N}
```

この例の Y または N は、スクリプトでデータベースの起動または停止を実行するかどうかを指定します。起動および停止するデータベースごとに、データベースのインスタンス識別子 (SID) を検索します。最初のフィールドの SID で識別し、最後のフィールドをそれぞれ Y に変更します。

注意： 新規データベース・インスタンスをシステムに追加する場合、これらのインスタンスを自動的に起動するには、必ず `oratab` ファイルでインスタンスのエントリを編集してください。

3. 使用しているプラットフォームに応じて、ディレクトリを次のいずれかに変更してください。

プラットフォーム	初期化ファイルのディレクトリ
AIX	/etc
Linux および Solaris	/etc/init.d
HP-UX および Tru64 UNIX	/sbin/init.d

4. 次の内容を含む `dbora` というファイルを作成します。ORACLE_HOME 環境変数の値を、各インストールの Oracle ホーム・ディレクトリに変更します。また ORACLE 環境変数の値を Oracle ホーム・ディレクトリにインストールされているデータベースの所有者のユーザー名 (通常は `oracle`) に変更します。

```
#!/bin/sh -x
#
# Change the value of ORACLE_HOME to specify the correct Oracle home
# directory for you installation

ORACLE_HOME=/u01/app/oracle/product/10.1.0/db_1
#
# change the value of ORACLE to the login name of the
# oracle owner at your site
#

ORACLE=oracle

PATH=${PATH}:${ORACLE_HOME}/bin
HOST=`hostname`
PLATFORM=`uname`
export ORACLE_HOME PATH
#
if [ ! "$2" = "ORA_DB" ] ; then
    if [ "$PLATFORM" = "HP-UX" ] ; then
        remsh $HOST -l $ORACLE -n "$0 $1 ORA_DB"
        exit
    else
        rsh $HOST -l $ORACLE $0 $1 ORA_DB
        exit
    fi
fi
#
LOG=$ORACLE_HOME/startup.log
touch $LOG
chmod a+r $LOG
#

case $1 in
'start')
    echo "$0: starting up" >> $LOG
```

```

date >> $LOG
# Start Oracle Net
if [ -f $ORACLE_HOME/bin/tnslsnr ] ; then
    echo "starting Oracle Net Listener"
    $ORACLE_HOME/bin/lsnrctl start >> $LOG 2>&1 &
fi
echo "Starting Oracle databases"
$ORACLE_HOME/bin/dbstart >> $LOG 2>&1 &
;;
'stop')
echo "$0: shutting down" >> $LOG
date >> $LOG
# Stop Oracle Net
if [ -f $ORACLE_HOME/bin/tnslsnr ] ; then
    echo "stopping Oracle Net Listener"
    $ORACLE_HOME/bin/lsnrctl stop >> $LOG 2>&1 &
fi
echo "stopping Oracle databases"
$ORACLE_HOME/bin/dbshut >> $LOG 2>&1 &
;;
*)
echo "usage: $0 {start|stop}"
exit
;;
esac
#
exit

```

注意： このスクリプトは、Oracle Net Listener のパスワードが設定されていない場合にのみ有効です。リスナーにパスワードが設定されている場合、このスクリプトによってリスナーを停止することはできません。また、リスナー名がデフォルト名 LISTENER でない場合、停止および起動コマンドでリスナー名を指定する必要があります。

```
$ORACLE_HOME/bin/lsnrctl {start|stop} listener_name
```

5. dbora ファイルのグループを OSDBA グループ（通常は dba）に変更し、その権限を 750 に設定します。

```

# chgrp dba dbora
# chmod 750 dbora

```

6. 次のように、dbora スクリプトへのシンボリック・リンクを適切な起動レベルのスクリプト・ディレクトリに作成します。

プラットフォーム	シンボリック・リンク・コマンド
AIX	<pre># ln -s /etc/dbora /etc/rc.d/rc2.d/S99dbora # ln -s /etc/dbora /etc/rc.d/rc2.d/K01dbora</pre>
HP-UX	<pre># ln -s /sbin/init.d/dbora /sbin/rc3.d/S99dbora # ln -s /sbin/init.d/dbora /sbin/rc3.d/K001dbora</pre>
Linux	<pre># ln -s /etc/init.d/dbora /etc/rc.d/rc3.d/K01dbora # ln -s /etc/init.d/dbora /etc/rc.d/rc3.d/S99dbora # ln -s /etc/init.d/dbora /etc/rc.d/rc5.d/K01dbora # ln -s /etc/init.d/dbora /etc/rc.d/rc5.d/S99dbora</pre>
Solaris	<pre># ln -s /etc/init.d/dbora /etc/rc3.d/K01dbora # ln -s /etc/init.d/dbora /etc/rc3.d/S99dbora</pre>
Tru64 UNIX	<pre># ln -s /sbin/init.d/dbora /sbin/rc3.d/S99dbora # ln -s /sbin/init.d/dbora /sbin/rc3.d/K01dbora</pre>

UNIX システムでの Oracle 製品の構成

この章では、UNIX システムで Oracle Database および Oracle ソフトウェアを管理する方法について説明します。次の項目について説明します。

- [Oracle 製品の追加に対するデータベースの構成](#)
- [スタンドアロン・ツールとしての Configuration Assistant の使用](#)
- [実行可能ファイルの再リンク](#)

Oracle 製品の追加に対するデータベースの構成

初期インストールの後で Oracle 製品を追加インストールする場合は、次のように、Database Configuration Assistant を使用して新しい製品用にデータベースを構成します。

1. 必要な場合は、データベースを起動します。
2. 次のコマンドを入力して、Database Configuration Assistant を起動します。

```
$ $ORACLE_HOME/bin/dbca
```
3. 「データベース・オプションの構成」を選択します。
4. 使用可能なデータベースのリストから、構成するデータベースを選択します。
5. リストから有効にする製品を選択し、「完了」をクリックします。

スタンドアロン・ツールとしての Configuration Assistant の使用

Configuration Assistant は通常、インストール・セッション中に実行されますが、スタンドアロン・モードでも実行できます。Oracle Universal Installer と同様、すべての Configuration Assistant は、応答ファイルを使用して、非対話方式で実行できます。この項では、次の Configuration Assistant の使用方法について説明します。

- [Oracle Net Configuration Assistant の使用](#)
- [Database Upgrade Assistant の使用](#)
- [Database Configuration Assistant の使用](#)
- [新規またはアップグレード済データベースの構成](#)

Oracle Net Configuration Assistant の使用

Oracle Net Server または Oracle Net Client をインストールすると、Oracle Universal Installer によって、Oracle Net Configuration Assistant が自動的に起動されます。

Oracle Client のインストールを個別に実行した場合、Oracle Net Configuration Assistant が、インストール時の選択内容に沿う構成を自動的に作成します。Installer は Oracle Net Configuration Assistant を自動的に実行して、クライアント・インストールの \$ORACLE_HOME/network/admin ディレクトリにあるローカル・ネーミング・ファイル内に ネット・サービス名を設定します。

インストールの完了後、Oracle Net Configuration Assistant を使用して詳細な構成を作成できます。次のコマンドを入力します。

```
$ $ORACLE_HOME/bin/netca
```

注意： Database Configuration Assistant を使用してデータベースを作成すると、ネットワーク構成ファイルが自動的に更新され、新しいデータベースに関する情報が追加されます。

Database Upgrade Assistant の使用

Installer では、Oracle Database のインストール時にデータベースを以前のリリースから現在のリリースにアップグレードできます。ただし、インストール時にデータベースをアップグレードしない場合、または複数のデータベースをアップグレードする場合は、インストール後に Database Upgrade Assistant を実行できます。

Oracle Database ソフトウェアをインストールし、インストール時にデータベースのアップグレードを選択しなかった場合は、マウントする前にデータベースをアップグレードする必要があります。

Database Upgrade Assistant を起動するには、次のコマンドを入力します。

```
$ $ORACLE_HOME/bin/dbua
```

Database Upgrade Assistant で使用できるコマンドライン・オプションに関する情報を取得するには、次のように、`-help` または `-h` コマンドライン引数を使用します。

```
$ $ORACLE_HOME/bin/dbua -help
```

関連項目： アップグレードの詳細は、『Oracle Database インストレーション・ガイド』および『Oracle Database アップグレード・ガイド』を参照してください。

Database Configuration Assistant の使用

Database Configuration Assistant を使用すると、次のことができます。

- デフォルトの、またはカスタマイズされたデータベースの作成
- Oracle 製品を使用する既存のデータベースの構成
- Automatic Storage Management のディスク・グループの作成
- 後のデータベース作成時に検査、修正および実行できるシェル・スクリプトと SQL スクリプトのセットの生成

Database Configuration Assistant を起動するには、次のコマンドを入力します。

```
$ $ORACLE_HOME/bin/dbca
```

Database Configuration Assistant で使用できるコマンドライン・オプションに関する情報を取得するには、次のように、`-help` または `-h` コマンドライン引数を使用します。

```
$ $ORACLE_HOME/bin/dbca -help
```

新規またはアップグレード済データベースの構成

データベースを作成またはアップグレードした後は、`utlrp.sql` スクリプトを実行することをお勧めします。このスクリプトを実行すると、無効な状態にある、パッケージ、プロシージャおよびタイプなどの、PL/SQL モジュールがすべて再コンパイルされます。これはオプションの手順ですが、後日ではなく、データベースの作成時に実行することをお勧めします。

`utlrp.sql` スクリプトを実行するには、次の手順を実行します。

1. ユーザーを `oracle` に切り替えます。
2. `oraenv` または `coraenv` スクリプトを使用して、`utlrp.sql` スクリプトを実行するデータベースの環境を設定します。
 - Bourne、Bash または Korn シェルの場合


```
$ . /usr/local/bin/oraenv
```
 - C シェルの場合


```
% source /usr/local/bin/coraenv
```

プロンプトが表示されたら、データベースの SID を指定します。

3. 次のように、SQL*Plus を起動します。

```
$ sqlplus "/ AS SYSDBA"
```

4. 必要に応じて、データベースを起動します。

```
SQL> STARTUP
```

5. utlrlp.sql スクリプトを実行します。

```
SQL> @?/rdbms/admin/utlrlp.sql
```

実行可能ファイルの再リンク

\$ORACLE_HOME/bin ディレクトリ内の relink シェル・スクリプトを使用して、製品の実行可能ファイルを手動で再リンクできます。製品の実行可能ファイルの再リンクは、オペレーティング・システムのパッチを適用するたびに、またはオペレーティング・システムのアップグレード後に必要となります。Oracle Real Application Clusters インストールを実行している場合は、Oracle ホーム・ディレクトリがクラスタ・ファイル・システムによってクラスタ内で共有されている場合を除き、各クラスタ・ノードにある製品の実行可能ファイルを再リンクする必要があります。

注意： Oracle ホーム・ディレクトリで実行されている実行可能ファイルのうち、再リンクする実行可能ファイルをすべて停止してください。また、Oracle 共有ライブラリにリンクされているアプリケーションも停止してください。

relink スクリプトを使用すると、Oracle 製品の実行可能ファイルを手動で再リンクできますが、その方法は Oracle ホーム・ディレクトリにインストールされている製品によって異なります。

製品の実行可能ファイルを再リンクするには、次のコマンドを入力します。argument には、表 3-1 に示した値のいずれかを指定します。

```
$ relink argument
```

表 3-1 再リンク・スクリプトの引数

引数	説明
all	インストールされているすべての製品の実行可能ファイル
oracle	Oracle Database の実行可能ファイルのみ
network	net_client、net_server、cman
client	net_client、plssql
client_sharedlib	クライアント共有ライブラリ
interMedia	ctx
ctx	Oracle Text ユーティリティ
precomp	インストールされているすべてのプリコンパイラ
utilities	インストールされているすべてのユーティリティ
oemagent	oemagent
	注意： nmo および nmb 実行可能ファイルに正しい権限を付与するには、oemagent を再リンクした後で、root.sh スクリプトを実行する必要があります。
ldap	ldap、oid

SQL*Plus の管理

この章では、UNIX システムで SQL*Plus を使用および管理する方法について説明します。次の項目について説明します。

- [コマンドライン SQL*Plus の管理](#)
- [コマンドライン SQL*Plus の使用](#)
- [SQL*Plus の制限事項](#)

関連項目： SQL*Plus の詳細は、『SQL*Plus ユーザーズ・ガイドおよびリファレンス』を参照してください。

コマンドライン SQL*Plus の管理

この項では、コマンドライン SQL*Plus の管理方法について説明します。この項の例で、疑問符 (?) が表示されている箇所では、環境変数 ORACLE_HOME の値が使用されます。

設定ファイルの使用

SQL*Plus を起動すると、最初にサイト・プロファイル設定ファイル glogin.sql が実行され、次にユーザー・プロファイル設定ファイル login.sql が実行されます。

サイト・プロファイル・ファイルの使用

グローバルなサイト・プロファイル・ファイルは、\$ORACLE_HOME/sqlplus/admin/glogin.sql です。この場所にサイト・プロファイル・ファイルがすでに存在する場合は、SQL*Plus をインストールしたときにそのファイルが上書きされます。SQL*Plus を削除すると、そのサイト・プロファイル・ファイルも削除されます。

ユーザー・プロファイル・ファイルの使用

ユーザー・プロファイル・ファイルは、login.sql です。SQL*Plus は、最初に現行のディレクトリを検索し、次に環境変数 SQLPATH で指定したディレクトリを検索してこのファイルを検出します。この環境変数には、ディレクトリをコロンで区切ったリストを設定します。SQL*Plus は、これらのディレクトリがリストされている順序で検索して、login.sql ファイルを検出します。

login.sql ファイルに設定されているオプションは、glogin.sql ファイルに設定されているオプションよりも優先されます。

関連項目： プロファイル・ファイルの詳細は、『SQL*Plus ユーザーズ・ガイド およびリファレンス』を参照してください。

PRODUCT_USER_PROFILE 表の使用

Oracle Database には、指定した SQL コマンドや SQL*Plus コマンドを無効にするための PRODUCT_USER_PROFILE 表が用意されています。この表は、事前構成済データベースをインストールするインストール・タイプを選択した場合に、自動的に作成されます。

関連項目： インストール・オプションの詳細は、『Oracle Database インストール・ガイド』を参照してください。

PRODUCT_USER_PROFILE 表を再作成するには、SYSTEM スキーマにある \$ORACLE_HOME/sqlplus/admin/pupbld.sql スクリプトを実行します。たとえば、次のコマンドを入力します。SYSTEM_PASSWORD は、SYSTEM ユーザーのパスワードです。

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

また、\$ORACLE_HOME/bin/pupbld シェル・スクリプトを使用して、手動で PRODUCT_USER_PROFILE 表を SYSTEM スキーマに再作成することもできます。このスクリプトは、SYSTEM パスワードを要求するプロンプトを表示します。プロンプトを表示せずに pupbld スクリプトを実行する必要がある場合は、環境変数 SYSTEM_PASS に SYSTEM ユーザー名とパスワードを設定してください。

デモンストレーション表の使用

Oracle Database には、テスト用のデモンストレーション表が用意されています。デモンストレーション表をデータベースにインストールするには、事前構成済データベースをインストールするインストール・タイプを選択する必要があります。

関連項目： インストール・オプションの詳細は、『Oracle Database インストール・ガイド』を参照してください。

SQL*Plus のコマンドライン・ヘルプ

この項では、SQL*Plus のコマンドライン・ヘルプのインストールおよび削除方法について説明します。

関連項目： SQL*Plus のコマンドライン・ヘルプの詳細は、『SQL*Plus ユーザーズ・ガイドおよびリファレンス』を参照してください。

SQL*Plus のコマンドライン・ヘルプのインストール

SQL*Plus のコマンドライン・ヘルプは、次の 3 つの方法でインストールできます。

- 事前構成済データベースのインストールを完了します。

インストールの一部として、事前構成済データベースをインストールすると、SQL*Plus のコマンドライン・ヘルプが SYSTEM スキーマに自動的にインストールされます。

- \$ORACLE_HOME/bin/helpins シェル・スクリプトを使用して、SYSTEM スキーマにコマンドライン・ヘルプを手動でインストールします。

helpins スクリプトは、SYSTEM パスワードを要求するプロンプトを表示します。プロンプトを表示せずにこのスクリプトを実行する必要がある場合は、環境変数 SYSTEM_PASS に SYSTEM ユーザー名とパスワードを設定してください。たとえば、次のように入力します。

- Bourne、Bash または Korn シェルの場合

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

- C シェルの場合

```
% setenv SYSTEM_PASS SYSTEM/system_password
```

- \$ORACLE_HOME/sqlplus/admin/help/helpbld.sql スクリプトを使用して、SYSTEM スキーマにコマンドライン・ヘルプを手動でインストールします。

たとえば、次のコマンドを入力します。system_password は、SYSTEM ユーザーのパスワードです。

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

注意： helpins シェル・スクリプトおよび helpbld.sql スクリプトは、新しい表を作成する前に既存のコマンドライン・ヘルプの表を削除します。

SQL*Plus のコマンドライン・ヘルプの削除

SYSTEM スキーマから SQL*Plus のコマンドライン・ヘルプの表を手動で削除するには、\$ORACLE_HOME/sqlplus/admin/help/helpdrop.sql スクリプトを実行します。たとえば、次のコマンドを入力します。system_password は、SYSTEM ユーザーのパスワードです。

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

コマンドライン SQL*Plus の使用

この項では、UNIX システムでコマンドライン SQL*Plus を使用方法について説明します。

SQL*Plus からのシステム・エディタの使用

SQL*Plus プロンプトで ED または EDIT コマンドを入力すると、ed、emacs、ned、または vi などのオペレーティング・システム・エディタが起動します。ただし、環境変数 PATH には、エディタの実行可能ファイルが格納されているディレクトリを指定する必要があります。

エディタを起動すると、現行の SQL バッファがエディタに格納されます。エディタを終了すると、変更された SQL バッファが SQL*Plus に戻されます。

SQL*Plus _EDITOR 変数を定義することにより、起動するエディタを指定できます。この変数は、glogin.sql サイト・プロファイル、login.sql ユーザー・プロファイルに定義することも、SQL*Plus セッション時に定義することもできます。たとえば、デフォルト・エディタを vi に設定するには、次のように入力します。

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

_EDITOR 変数を設定しない場合は、環境変数 EDITOR または VISUAL のいずれかの値が使用されます。両方の環境変数が設定されている場合は、環境変数 EDITOR の値が使用されます。_EDITOR、EDITOR および VISUAL のいずれも指定されていない場合、デフォルト・エディタは、ed になります。

エディタを起動すると、SQL*Plus は一時ファイル afiedt.buf を使用してエディタにテキストを渡します。SET EDITFILE コマンドを使用すると、別のファイル名を指定できます。たとえば、次のように入力します。

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

SQL*Plus は、一時ファイルを削除しません。

SQL*Plus からのオペレーティング・システム・コマンドの実行

SQL*Plus プロンプトの後の最初の文字として、HOST コマンドまたは感嘆符 (!) を使用すると、後続の文字がサブシェルに渡されます。オペレーティング・システム・コマンドを実行するときに使用するシェルは、環境変数 SHELL によって設定されます。デフォルト・シェルは、Bourne シェル (/bin/sh) です。シェルが実行できない場合は、エラー・メッセージが表示されます。

SQL*Plus に戻るには、exit コマンドを入力するか、[Ctrl] キーを押しながら [D] キーを押します。

たとえば、特定のコマンドを実行するには、次のように入力します。

```
SQL> ! command
```

この例の command は、実行するオペレーティング・システム・コマンドです。

SQL*Plus から複数のオペレーティング・システム・コマンドを実行するには、HOST または ! コマンドを入力してから、[Enter] キーを押します。オペレーティング・システム・プロンプトに戻ります。

SQL*Plus への割込み

SQL*Plus の実行中に、[Ctrl] キーを押しながら [C] キーを押すと、スクロール中のレコード表示を停止し、SQL 文を終了できます。

SPOOL コマンドの使用

SPOOL コマンドで生成されるファイルのデフォルトのファイル拡張子は、`.lst` です。この拡張子を変更するには、ピリオド (.) を含めたスプール・ファイル名を指定します。たとえば、次のように入力します。

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

SQL*Plus の制限事項

この項では、SQL*Plus の制限事項を説明します。

ウィンドウのサイズ変更

SQL*Plus のシステム変数 `LINESIZE` および `PAGESIZE` のデフォルト値では、ウィンドウのサイズは自動的に調整されません。

リターン・コード

UNIX のリターン・コードは 1 バイトですが、Oracle エラー・コードを返すには、1 バイトでは不十分です。リターン・コードの範囲は、0 ~ 255 です。

パスワードの非表示

環境変数 `SYSTEM_PASS` に `SYSTEM` ユーザーのユーザー名とパスワードを設定した場合、`ps` コマンドの出力にこの情報が表示される場合があります。権限のないアクセスを防ぐため、SQL*Plus によってプロンプトが表示された場合のみ `SYSTEM` パスワードを入力してください。

自動的にスクリプトを実行する場合は、Oracle Database へのログインが外部で認証されるなど、パスワードを格納する必要のない認証方法の使用を考慮してください。セキュリティの低い環境では、スクリプト・ファイルに UNIX パイプを使用して、パスワードを SQL*Plus に渡すことを検討してください。たとえば、次のように入力します。

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

あるいは、コマンド・プロンプトで次の行を入力します。

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

この例の `system_password` は、`SYSTEM` ユーザーのパスワードです。

Oracle Net Services の構成

この章では、UNIX システムで Oracle Net Services を構成する方法について説明します。次の項目について説明します。

- [Oracle Net Services 構成ファイルの保存場所](#)
- [adapters ユーティリティ](#)
- [Oracle protocol support](#)
- [TCP/IP または SSL 付き TCP/IP 用のリスナーの設定](#)
- [Oracle Advanced Security](#)

関連項目： Oracle ネットワーキングの詳細は、『Oracle Net Services 管理者ガイド』を参照してください。

Oracle Net Services 構成ファイルの保存場所

Oracle Net Services 構成ファイルは、多くの場合、\$ORACLE_HOME/network/admin ディレクトリにあります。このファイルに応じて、Oracle Net は、異なる検索順序でファイルを検出します。

sqlnet.ora および ldap.ora の各ファイルの検索順序は、次のとおりです。

- 1. 環境変数 TNS_ADMIN で指定したディレクトリ（設定されている場合）
- 2. \$ORACLE_HOME/network/admin ディレクトリ

cman.ora、listener.ora および tnsnames.ora の各ファイルの検索順序は、次のとおりです。

- 1. 環境変数 TNS_ADMIN で指定したディレクトリ（設定されている場合）
- 2. 次のいずれかのディレクトリ
 - Solaris
/var/opt/oracle
 - その他のオペレーティング・システム
/etc

- 3. \$ORACLE_HOME/network/admin ディレクトリ

一部のシステム・レベルの構成ファイルでは、対応するユーザー・レベルの構成ファイル（ユーザーのホーム・ディレクトリに格納されている）を作成できます。ユーザー・レベルのファイルの設定によって、システム・レベルのファイルの設定が上書きされます。次の表に、システム・レベルの構成ファイルとそれに対応するユーザー・レベルの構成ファイルを示します。

システム・レベルの構成ファイル	ユーザー・レベルの構成ファイル
sqlnet.ora	\$HOME/.sqlnet.ora
tnsnames.ora	\$HOME/.tnsnames.ora

サンプル構成ファイル

\$ORACLE_HOME/network/admin/samples ディレクトリには、cman.ora、listener.ora、sqlnet.ora、tnsnames.ora の各構成ファイルのサンプルが含まれています。

注意： cman.ora ファイルがインストールされるのは、カスタム・インストールの一部として Connection Manager を選択した場合のみです。

adapters ユーティリティ

使用しているシステムで Oracle Database がサポートしているトランスポート・プロトコル、ネーミング・メソッドおよび Oracle Advanced Security のオプションを表示する場合は、adapters ユーティリティを使用します。adapters ユーティリティを使用するには、次のコマンドを入力します。

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

adapters ユーティリティを実行すると、次のような出力が表示されます。

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
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
RADIUS authentication
ENTRUST authentication
```

Oracle Client システムで、構成済の Oracle トランスポート・プロトコル、ネーミング・メソッドおよびセキュリティ・オプションを表示する場合は、adapters ユーティリティを実行します。Oracle Client システムで adapters ユーティリティを実行するには、次のコマンドを入力します。

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

adapters ユーティリティを実行すると、次のような出力が表示されます。

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
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
RADIUS authentication
```

関連項目： adapters ユーティリティの詳細は、『Oracle Net Services 管理者ガイド』を参照してください。

Oracle protocol support

Oracle protocol support は、Oracle Net のコンポーネントの 1 つです。次の要素で構成されています。

- IPC プロトコル・サポート
- TCP/IP プロトコル・サポート
- SSL 付き TCP/IP プロトコル・サポート

IPC、TCP/IP および SSL 付き TCP/IP の各プロトコル・サポートには、それぞれアドレス指定が必要です。このアドレスは、Oracle Net Services 構成ファイルおよび DISPATCHER 初期化パラメータで使用されます。次の各項では、各プロトコル・サポートのアドレス指定について説明します。

注意： HP-UX (PA-RISC) および Tru64 UNIX システムで DCE がインストールされている場合は、DCE を Oracle Net プロトコルとして使用できません。DCE プロトコル・サポートの構成の詳細は、『Oracle Advanced Security 管理者ガイド』を参照してください。

関連項目： Oracle protocol support の詳細は、『Oracle Net Services 管理者ガイド』を参照してください。

IPC プロトコル・サポート

IPC プロトコル・サポートは、クライアント・プログラムと Oracle Database が同じシステムにインストールされている場合にのみ使用できます。このプロトコル・サポートには、リスナーが必要です。このプロトコル・サポートは、すべてのクライアント・ツールおよび oracle 実行可能ファイルにインストールおよびリンクされます。

IPC プロトコル・サポートには、次の書式のアドレス指定が必要です。

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

次の表に、このアドレス指定で使用するパラメータを示します。

パラメータ	説明
PROTOCOL	使用するプロトコルを表します。この値は IPC です。大 / 小文字は区別されません。
KEY	同じシステムで IPC KEY として使用されている他の名前とは異なる、一意の名前を表します。

次に、IPC プロトコル・アドレスのサンプルを示します。

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

TCP/IP プロトコル・サポート

TCP/IP は、ネットワークを介したクライアント / サーバー通信に使用される標準的な通信プロトコルです。TCP/IP プロトコル・サポートを使用すると、クライアント・プログラムと Oracle Database が同じシステムまたは別のシステムのどちらにインストールされていても、それらの間で通信を行うことができます。使用しているシステムに TCP/IP プロトコルをインストールした場合、TCP/IP プロトコル・サポートがすべてのクライアント・ツールおよび oracle 実行可能ファイルにインストールおよびリンクされます。

TCP/IP プロトコル・サポートには、次の書式のアドレス指定が必要です。

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

次の表に、このアドレス指定で使用するパラメータを示します。

パラメータ	説明
PROTOCOL	使用するプロトコル・サポートを表します。この値は TCP です。大 / 小文字は区別されません。
HOST	ホスト名またはホストの IP アドレスを表します。
PORT	TCP/IP ポートを表します。このポートには、番号または /etc/services ファイルでこのポートにマップされた別名を指定します。推奨値は 1521 です。

次に、TCP/IP プロトコル・アドレスのサンプルを示します。

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

SSL 付き TCP/IP プロトコル・サポート

SSL 付き TCP/IP プロトコル・サポートを使用すると、クライアントの Oracle アプリケーションは、TCP/IP および SSL を介したリモートの Oracle データベースと通信できます。SSL 付き TCP/IP を使用するには、Oracle Advanced Security をインストールしておく必要があります。

SSL 付き TCP/IP プロトコル・サポートには、次の書式のアドレス指定が必要です。

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

次の表に、このアドレス指定で使用するパラメータを示します。

パラメータ	説明
PROTOCOL	使用するプロトコルを表します。この値は TCPS です。大 / 小文字は区別されません。
HOST	ホスト名またはホストの IP アドレスを表します。
PORT	SSL 付き TCP/IP ポートを表します。このポートには、番号または <code>/etc/services</code> ファイルでこのポートにマップされた別名を指定します。推奨値は 2484 です。

次に、SSL 付き TCP/IP プロトコル・アドレスのサンプルを示します。

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

TCP/IP または SSL 付き TCP/IP 用のリスナーの設定

リスナー用のポートは、ネットワーク上の各 Oracle Net Services ノードの `/etc/services` ファイルで予約することをお勧めします。デフォルトのポートは 1521 です。リスナー名とポート番号のエントリは次のように入力します。

```
oraclelistener 1521/tcp
```

この例の `oraclelistener` は、`listener.ora` ファイルに定義されているリスナーの名前です。複数のリスナーを起動する場合は、複数のポートを予約してください。

SSL を使用する場合は、SSL 付き TCP/IP 用のポートを `/etc/services` ファイルに定義する必要があります。推奨値は 2484 です。たとえば、次のように入力します。

```
oraclelistenerssl 2484/tcps
```

この例の `oraclelistenerssl` は、`listener.ora` ファイルに定義されているリスナーの名前です。複数のリスナーを起動する場合は、複数のポートを予約してください。

Oracle Advanced Security

Oracle Advanced Security をインストールすると、3 つの `.bak` ファイル (`naeet.o.bak`、`naect.o.bak` および `naedhs.o.bak`) が作成されます。これらのファイルは、`$ORACLE_HOME/lib` ディレクトリに格納されます。これらのファイルは、Oracle Advanced Security を削除するときの再リンクに必要です。これらのファイルを削除しないでください。

Oracle プリコンパイラおよび Oracle Call Interface の使用

この章では、Oracle プリコンパイラおよび Oracle Call Interface について説明します。次の項目について説明します。

注意： この章で説明するデモンストレーションを使用するには、Oracle Database 10g Companion CD に収録されている Oracle Database Examples をインストールする必要があります。

- Oracle プリコンパイラの概要
- クライアント・アプリケーションのビット長サポート
- Pro*C/C++ プリコンパイラ
- Pro*COBOL プリコンパイラ
- Pro*FORTRAN プリコンパイラ
- AIX の場合のみ: SQL*Module for Ada
- Oracle Call Interface と Oracle C++ Call Interface
- 64 ビット・ドライバを使用する Oracle JDBC/OCI プログラム (AIX、HP-UX、Solaris SPARC および zSeries Linux のみ)
- カスタム Make ファイル
- 未定義シンボルの修正
- マルチスレッド・アプリケーション
- シグナル・ハンドラの使用
- XA 機能

Oracle プリコンパイラの概要

Oracle プリコンパイラは、Oracle データベースの SQL 文を高水準言語で記述されたプログラムと組み合わせるためのアプリケーション開発ツールです。Oracle プリコンパイラは、ANSI SQL と互換性があり、Oracle Database やその他の ANSI SQL データベース管理システムで実行するオープンでカスタマイズされたアプリケーションを開発するために使用します。

プリコンパイラ構成ファイル

Oracle プリコンパイラの構成ファイルは、`$ORACLE_HOME/precomp/admin` ディレクトリにあります。表 6-1 に、各プリコンパイラの構成ファイルの名前を示します。

表 6-1 Oracle プリコンパイラのシステム構成ファイル

製品	構成ファイル
Pro*C/C++	pccscfg.cfg
Pro*COBOL (AIX、HP-UX、Linux x86、Solaris SPARC、Tru64 UNIX および zSeries Linux のみ)	pcbcfg.cfg
Pro*FORTRAN (AIX、HP-UX、Solaris、Tru64 UNIX)	pccfor.cfg
Object Type Translator	ottcfg.cfg
SQL*Module for Ada (AIX の場合のみ)	pmscfg.cfg

プリコンパイラ実行可能ファイルの再リンク

すべてのプリコンパイラ実行可能ファイルを再リンクするには、Make ファイル `$ORACLE_HOME/precomp/lib/ins_precomp.mk` を使用します。特定のプリコンパイラ実行可能ファイルを手動で再リンクするには、次のコマンドを入力します。

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

このコマンドを実行すると、最初に、新しい実行可能ファイルが `$ORACLE_HOME/precomp/lib` ディレクトリに作成され、次にそのファイルが `$ORACLE_HOME/bin` ディレクトリに移動されます。

この例の `executable` は、表 6-2 に示す製品の実行可能ファイルです。

表 6-2 製品とそれに対応する実行可能ファイル

製品	実行可能ファイル
Pro*C/C++	proc
Pro*COBOL (AIX、HP-UX、Linux x86、Solaris SPARC、Tru64 UNIX および zSeries Linux)	procob または rtsora
Pro*COBOL 32 ビット (AIX、HP-UX PA-RISC、Solaris SPARC および zSeries Linux)	procob32 または rtsora32
Pro*FORTRAN (AIX、HP-UX、Solaris および Tru64 UNIX の場合のみ)	profor
Pro*FORTRAN 32 ビット (HP-UX の場合のみ)	profor32
SQL*Module for Ada (AIX の場合のみ)	modada

プリコンパイラの README ファイル

表 6-3 に、プリコンパイラの README ファイルの場所を示します。README ファイルには、前回のリリース以降にプリコンパイラに加えられた変更が記載されています。

表 6-3 プリコンパイラの README ファイルの保存場所

プリコンパイラ	README ファイル
Pro*C/C++	\$ORACLE_HOME/precomp/doc/proc2/readme.doc
Pro*COBOL	\$ORACLE_HOME/precomp/doc/procob2/readme.doc
Pro*FORTRAN	\$ORACLE_HOME/precomp/doc/pro1x/readme.txt

すべてのプリコンパイラに共通の問題

次の問題は、すべてのプリコンパイラに共通しています。

大文字から小文字への変換

C 言語以外では、コンパイラによって、大文字の関数やサブプログラム名が小文字に変換されます。この結果、ユーザーが存在しない旨のエラー・メッセージが表示されることがあります。このエラー・メッセージが表示された場合は、オプション・ファイル内の関数またはサブプログラム名の大 / 小文字が、IAPXTB 表の文字と一致しているかどうかを確認してください。

ベンダー提供のデバッガ・プログラム

プリコンパイラとベンダー提供のデバッガに互換性がない場合があります。デバッガを使用して動作したプログラムが、デバッガを使用しないときに同様に動作するとはかぎりません。

IRECLEN および ORECLEN の値

IRECLEN および ORECLEN の各パラメータには、最大値がありません。

静的および動的リンク

Oracle ライブラリは、プリコンパイラや OCI または OCCI アプリケーションと静的または動的にリンクできます。静的リンクの場合、アプリケーション全体のライブラリおよびオブジェクトは、1 つの実行可能プログラムにリンクされます。その結果、アプリケーションの実行可能ファイルが非常に大きくなることがあります。

動的リンクの場合、実行コードの一部が実行可能プログラムに格納され、残りの部分はアプリケーションの実行時に動的にリンクされるライブラリに格納されます。実行時にリンクされるライブラリを、動的ライブラリまたは共有ライブラリといいます。動的リンクには、次のようないくつかのメリットがあります。

- ディスク領域要件が少なくなります。複数のアプリケーションで、または同一アプリケーションの起動で、同じ動的ライブラリを使用できます。
- メイン・メモリー要件が少なくなります。同一の動的ライブラリ・イメージをメイン・メモリーに一度ロードしておく、複数のアプリケーションでそれを共有できます。

クライアント共有ライブラリとクライアント静的ライブラリ

クライアント共有ライブラリとクライアント静的ライブラリは、使用しているプラットフォームに応じて、`$ORACLE_HOME/lib` ディレクトリまたは `$ORACLE_HOME/lib32` ディレクトリにあります。オラクル社が提供している Make ファイル `demo_product.mk` を使用してアプリケーションをリンクする場合は、デフォルトでクライアント共有ライブラリがリンクされます。

使用しているプラットフォームの共有ライブラリ・パスの環境変数設定に、クライアント共有ライブラリが含まれるディレクトリが指定されていない場合、実行可能ファイルの起動時に、次のようなエラー・メッセージのいずれかが表示される場合があります。

```
Cannot load library libclntsh.a
Can't open shared library: ../libclntsh.sl.10.1
libclntsh.so.10.1: can't open file: errno=2
can't open library: ../libclntsh.dylib.10.1
Cannot map libclntsh.so
```

これらのエラーを防ぐには、使用しているプラットフォームの共有ライブラリ・パスの環境変数を設定し、適切なディレクトリを指定します。次の表に、各プラットフォームの環境変数名およびディレクトリを示します。使用しているプラットフォームで 32 ビットおよび 64 ビットの実行可能アプリケーションがどちらもサポートされている場合、実行するアプリケーションに応じて、正しいディレクトリを指定してください。

プラットフォーム	環境変数	サンプル設定
AIX (64 ビット・アプリケーション)	LIBPATH	<code>\$ORACLE_HOME/lib</code>
AIX (32 ビット・アプリケーション)	LIBPATH	<code>\$ORACLE_HOME/lib32</code>
HP-UX (64 ビット・アプリケーション)、Linux、Solaris および Tru64 UNIX	LD_LIBRARY_PATH	<code>\$ORACLE_HOME/lib</code>
zSeries Linux (31 ビット・アプリケーション)	LD_LIBRARY_PATH	<code>\$ORACLE_HOME/lib32</code>
HP-UX (32 ビット・アプリケーション)	SHLIB_PATH	<code>\$ORACLE_HOME/lib32</code>
Mac OS X	DYLD_LIBRARY_PATH	<code>\$ORACLE_HOME/lib</code>

クライアント共有ライブラリは、インストール時に自動的に作成されます。クライアント共有ライブラリを再作成する場合は、次の手順を実行します。

1. クライアント共有ライブラリを使用するすべてのクライアント・アプリケーションを終了します。この中には、SQL*Plus や Recovery Manager などの Oracle Client アプリケーションもすべて含まれます。
2. oracle ユーザーでログインし、次のコマンドを入力します。

```
$ $ORACLE_HOME/bin/genclntsh
```

HP-UX PA-RISC のみ：非スレッド化クライアント共有ライブラリ

HP-UX PA-RISC では、非スレッド化クライアント共有ライブラリを使用できます。ただし、このライブラリは、スレッドを使用する OCI アプリケーションやスレッドに依存する OCI アプリケーションでは使用できません。

スレッドを使用しないアプリケーションにこのライブラリを使用するには、次のいずれかのコマンドを使用して OCI アプリケーションを作成します。

- 32 ビット・アプリケーションの場合


```
$ make -f demo_rdbms32.mk build_nopthread EXE=oci02 OBJS=oci02.o
```
- 64 ビット・アプリケーションの場合


```
$ make -f demo_rdbms.mk build_nopthread EXE=oci02 OBJS=oci02.o
```

クライアント・アプリケーションのビット長サポート

次の表に、各プラットフォームでクライアント・アプリケーションについてサポートされるビット長（31 ビット、32 ビットまたは 64 ビット）を示します。

クライアント・アプリケーションのタイプ	サポートされるプラットフォーム
32 ビットのみ	Linux x86、Mac OS X および Solaris x86
64 ビットのみ	Linux Itanium および Tru64 UNIX
32 ビットおよび 64 ビット	AIX、HP-UX および Solaris SPARC
31 ビットおよび 64 ビット	zSeries Linux

AIX、HP-UX、Solaris SPARC および zSeries Linux では、Oracle Database 10g リリース 1 (10.1) とともに提供されているデモンストレーションおよびクライアント・アプリケーションはすべて、64 ビット・モードでリンクおよび実行します。ただし、AIX、HP-UX および Solaris SPARC では、同じ Oracle ホーム・ディレクトリに 32 ビットおよび 64 ビットのクライアント・アプリケーションを作成できます。同様に、zSeries Linux では、同じ Oracle ホーム・ディレクトリに 31 ビットおよび 64 ビットのクライアント・アプリケーションを作成できます。

次の表に、これらのプラットフォームの 32 ビット（または 31 ビット）および 64 ビットのクライアント共有ライブラリを示します。

プラットフォーム	32 ビット（または 31 ビット）の クライアント共有ライブラリ	64 ビットの クライアント共有ライブラリ
AIX	\$ORACLE_HOME/lib32/libclntsh.a \$ORACLE_HOME/lib32/libclntsh.so	\$ORACLE_HOME/lib/libclntsh.a \$ORACLE_HOME/lib/libclntsh.so
HP-UX PA-RISC	\$ORACLE_HOME/lib32/libclntsh.sl	\$ORACLE_HOME/lib/libclntsh.sl
HP-UX Itanium、 Solaris SPARC および zSeries Linux	\$ORACLE_HOME/lib32/libclntsh.so 注意： zSeries Linux では、 \$ORACLE_HOME/lib32 ディレクトリ に 31 ビットのライブラリが含まれます。	\$ORACLE_HOME/lib/libclntsh.so

異なるワード・サイズが混在しているインストールを実装するには、次の手順を実行します。

1. 次のコマンドを入力して、32 ビット（または zSeries Linux では 31 ビット）および 64 ビットのクライアント共有ライブラリを作成します。

```
$ $ORACLE_HOME/bin/genclntsh
```

2. 使用しているプラットフォームに応じて、次のいずれかの環境変数に、必要な 32 ビットおよび 64 ビットのクライアント共有ライブラリのパスを入力します。

プラットフォーム	環境変数
AIX	LIBPATH
HP-UX（32 ビットのクライアント・アプリケーション）	SHLIB_PATH
HP-UX、Solaris SPARC および zSeries Linux	LD_LIBRARY_PATH

32 ビットの Pro*C および OCI カスタマ・アプリケーションの作成

32 ビットおよび 64 ビットの Pro*C および Oracle Call Interface (OCI) カスタマ・アプリケーションをサポートするプラットフォームでは、次のファイルに 32 ビット Pro*C および OCI アプリケーションの作成方法が記載されています。

作成内容	参照先の Make ファイル
32 ビットの Pro*C アプリケーションの作成	\$ORACLE_HOME/precomp/demo/proc/demo_proc32.mk
32 ビットの OCI アプリケーションの作成	\$ORACLE_HOME/rdbms/demo/demo_rdbms32.mk

32 ビットの実行可能ファイルおよびライブラリ (AIX、HP-UX、Solaris SPARC および zSeries Linux のみ)

32 ビットおよび 64 ビットのアプリケーションをサポートするプラットフォームでは、\$ORACLE_HOME/bin ディレクトリに 32 ビットおよび 64 ビットの実行可能ファイルが格納されています。これらのプラットフォームでは、次のディレクトリに 32 ビットのライブラリが格納されています。

- \$ORACLE_HOME/lib32
- \$ORACLE_HOME/rdbms/lib32
- \$ORACLE_HOME/hs/lib32
- \$ORACLE_HOME/network/lib32
- \$ORACLE_HOME/precomp/lib32

Pro*C/C++ プリコンパイラ

Pro*C/C++ プリコンパイラを使用する場合は、事前にオペレーティング・システムの適切なバージョンのコンパイラが正しくインストールされていることを確認してください。

関連項目： 各プラットフォームでサポートされるコンパイラのバージョンの詳細は、『Oracle Database インストール・ガイド』を参照してください。Pro*C/C++ プリコンパイラおよびインタフェース機能の詳細は、『Pro*C/C++ Precompiler プログラマーズ・ガイド』を参照してください。

Pro*C/C++ のデモ・プログラム

デモ・プログラムは、Pro*C/C++ プリコンパイラの機能を紹介するために用意されています。デモ・プログラムには、C、C++ および Object プログラムの 3 種類があります。デモ・プログラムはすべて、\$ORACLE_HOME/precomp/demo/proc ディレクトリにあります。デフォルトでは、すべてのプログラムがクライアント共有ライブラリに動的にリンクされます。

プログラムを実行するには、\$ORACLE_HOME/sqlplus/demo/demobld.sql スクリプトで作成したデモンストレーション表が SCOTT スキーマにあり、そのパスワードが TIGER である必要があります。

注意： デモンストレーションを作成する前に、SCOTT アカウントのロックを解除してパスワードを設定する必要があります。

デモ・プログラムを作成するには、\$ORACLE_HOME/precomp/demo/proc/ ディレクトリにある Make ファイル demo_proc.mk を使用します。たとえば、sample1 のデモ・プログラムをプリコンパイル、コンパイルおよびリンクするには、次のコマンドを入力します。

注意： AIX システムの場合のみ、デモ・プログラムが正しくコンパイルされるようにするため、次の例で `make` コマンドに `-r` オプションを指定してください。たとえば、次のように入力します。

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

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

Pro*C/C++ の C デモ・プログラムをすべて作成するには、次のように入力します。

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

Pro*C/C++ の C++ デモ・プログラムをすべて作成するには、次のように入力します。

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

Pro*C/C++ の Object デモ・プログラムをすべて作成するには、次のように入力します。

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

一部のデモ・プログラムでは、`$ORACLE_HOME/precomp/demo/sql` ディレクトリ内にある SQL スクリプトを実行する必要があります。このスクリプトを実行しないと、実行を要求するメッセージが表示されます。デモ・プログラムを作成し、それに対応する SQL スクリプトを実行するには、「`make` マクロ引数 `RUNSQL=run`」をコマンドラインに追加します。たとえば、`sample9` デモ・プログラムを作成し、必要な `$ORACLE_HOME/precomp/demo/sql/sample9.sql` スクリプトを実行するには、次のように入力します。

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

Object デモ・プログラムをすべて作成し、それに対応する必要な SQL スクリプトをすべて実行するには、次のように入力します。

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

Pro*C/C++ のユーザー・プログラム

Make ファイル `$ORACLE_HOME/precomp/demo/proc/demo_proc.mk` を使用して、ユーザー・プログラムを作成できます。この Make ファイルは、使用しているプラットフォームに応じて、32 ビットまたは 64 ビットのユーザー・プログラムを作成します。一部のプラットフォームでは、Make ファイル `demo_proc32.mk` を使用して、32 ビット（または 31 ビット）のユーザー・プログラムを作成することもできます。次の表に、各プラットフォームで Pro*C/C++ により 32 ビット（または 31 ビット）および 64 ビットのユーザー・プログラムを作成するための Make ファイルを示します。

プラットフォーム	64 ビット Make ファイル	32 ビット Make ファイル
AIX、HP-UX および Solaris SPARC	<code>demo_proc.mk</code>	<code>demo_proc32.mk</code>
Linux x86、Mac OS X および Solaris x86	該当なし	<code>demo_proc.mk</code>
Linux Itanium および Tru64 UNIX	<code>demo_proc.mk</code>	該当なし
zSeries Linux	<code>demo_proc.mk</code>	<code>demo_proc32.mk</code>

注意： この Make ファイルは 31 ビットのユーザー・プログラムを作成します。

注意： ユーザー・プログラムの作成方法は、その Make ファイルを参照してください。

Make ファイル `demo_proc.mk` を使用してプログラムを作成する場合は、次のようなコマンドを入力します。

注意： AIX システムの場合のみ、プログラムが正しくコンパイルされるようにするため、次の例で `make` コマンドに `-r` オプションを指定してください。

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

各項目の意味は次のとおりです。

- `target` は、使用する Make ファイルのターゲットです。
- `objfilen` は、プログラムをリンクするためのオブジェクト・ファイルです。
- `exename` は、実行可能プログラムです。

たとえば、Pro*C/C++ ソース・ファイル `myprog.pc` からプログラム `myprog` を作成する場合は、作成する実行可能ファイルのソースとタイプに応じて、次のいずれかのコマンドを入力します。

- C ソースの場合、クライアント共有ライブラリに動的にリンクさせるには、次のように入力します。

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

- C ソースの場合、クライアント共有ライブラリに静的にリンクさせるには、次のように入力します。

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

- C++ ソースの場合、クライアント共有ライブラリに動的にリンクさせるには、次のように入力します。

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

- C++ ソースの場合、クライアント共有ライブラリに静的にリンクさせるには、次のように入力します。

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

Pro*COBOL プリコンパイラ

注意： Linux Itanium、Mac OS X または Solaris x86 では、Pro*COBOL はサポートされていません。

表 6-4 に、Pro*COBOL プリコンパイラのネーミング規則を示します。

表 6-4 Pro*COBOL のネーミング規則

項目	ネーミング規則
実行可能ファイル	procob または procob32
デモンストレーション・ディレクトリ	procob2
Make ファイル	demo_procob.mk または demo_procob_32.mk

Pro*COBOL では、静的リンク、動的リンクまたは動的読取りプログラムをサポートしています。動的リンク・プログラムは、クライアント共有ライブラリを使用します。動的読取りプログラムは、`$ORACLE_HOME/bin` ディレクトリにある `rtsora` (32 ビットの COBOL コンパイラの場合は、`rtsora32`) 実行可能ファイルを使用します。

Pro*COBOL の環境変数

この項では、Pro*COBOL で必要な環境変数について説明します。

Acucorp ACUCOBOL-GT COBOL コンパイラ

Acucorp ACUCOBOL-GT COBOL コンパイラを使用するには、A_TERMCAP、A_TERM、PATH および LD_LIBRARY_PATH 環境変数を設定する必要があります。環境変数 LD_LIBRARY_PATH に正しいディレクトリが設定されていない場合は、プログラムのコンパイルまたは実行時に、次のようなエラー・メッセージが表示されます。

```
runcbl: error while loading shared libraries: libclntsh.so.10.1:
cannot open shared object file: No such file or directory
```

A_TERMCAP および A_TERM a_termcap ファイルの場所を指定するには A_TERMCAP 環境変数を設定し、そのファイルからサポートされる端末を指定するには A_TERM 環境変数を設定します。たとえば、次のように入力します。

- Bourne、Bash または Korn シェルの場合

```
$ A_TERMCAP=/opt/COBOL/etc/a_termcap
$ A_TERM=vt100
$ export A_TERMCAP A_TERM
```

- C シェルの場合

```
% setenv A_TERMCAP /opt/COBOL/etc/a_termcap
% setenv A_TERM vt100
```

PATH /opt/COBOL/bin ディレクトリを含むように PATH 環境変数を設定します（zSeries Linux システムで 31 ビット・コンパイラを使用している場合は、/opt/COBOL31/bin ディレクトリ）。

- Bourne、Bash または Korn シェルの場合

```
$ PATH=/opt/COBOL/bin:$PATH
$ export PATH
```

- C シェルの場合

```
% setenv PATH opt/COBOL/bin:${PATH}
```

LD_LIBRARY_PATH 環境変数 LD_LIBRARY_PATH には、コンパイラ・ライブラリがインストールされているディレクトリを設定します。たとえば、コンパイラ・ライブラリが /opt/COBOL/lib ディレクトリにインストールされている場合は、次のように入力します。

- Bourne、Bash または Korn シェルの場合

```
$ LD_LIBRARY_PATH=${LD_LIBRARY_PATH}:/opt/COBOL/lib
$ export LD_LIBRARY_PATH
```

- C シェルの場合

```
% setenv LD_LIBRARY_PATH ${LD_LIBRARY_PATH}:/opt/COBOL/lib
```

Micro Focus Server Express COBOL コンパイラ

Micro Focus Server Express COBOL コンパイラを使用するには、環境変数 COBDIR および PATH と、使用しているプラットフォームに該当する共有ライブラリ・パス環境変数を設定する必要があります。次の表に、各プラットフォームに該当する共有ライブラリ・パス環境変数を示します。

プラットフォーム	環境変数
AIX	LIBPATH
HP-UX (32 ビットのクライアント・アプリケーション)	SHLIB_PATH
HP-UX、Linux、Solaris SPARC および zSeries Linux	LD_LIBRARY_PATH

共有ライブラリ・パス環境変数に \$COBDIR/coblib ディレクトリが設定されていない場合は、プログラムのコンパイルまたは実行時に、次のようなエラー・メッセージが表示されます。

- Linux の場合

```
rtsora: error while loading shared libraries: libcobrts_t.so:
cannot open shared object file: No such file or directory
```

- Tru64 UNIX の場合

```
356835:rtsora: /sbin/loader: Fatal Error:
Cannot map library libcobrts64_t.so.2
```

- AIX、HP-UX PA-RISC および Solaris SPARC の場合

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

- HP-UX Itanium の場合

```
/usr/lib/hpux64/dld.so: Unable to find library 'libcobrts64_t.so.2'.
Killed
```

COBDIR 環境変数 COBDIR には、コンパイラがインストールされているディレクトリを設定します。たとえば、コンパイラが /opt/lib/cobol ディレクトリにインストールされている場合は、次のように入力します。

- Bourne、Bash または Korn シェルの場合

```
$ COBDIR=/opt/lib/cobol
$ export COBDIR
```

- C シェルの場合

```
% setenv COBDIR /opt/lib/cobol
```

PATH \$COBDIR/bin ディレクトリを含むように PATH 環境変数を設定します。

- Bourne、Bash または Korn シェルの場合

```
$ PATH=$COBDIR/bin:$PATH
$ export PATH
```

- C シェルの場合

```
% setenv PATH ${COBDIR}/bin:${PATH}
```

共有ライブラリ・パス 環境変数 LIBPATH、LD_LIBRARY_PATH または SHLIB_PATH には、コンパイラ・ライブラリがインストールされているディレクトリを設定します。たとえば、プラットフォームで環境変数 LD_LIBRARY_PATH が使用されており、コンパイラ・ライブラリが \$COBDIR/coblib ディレクトリにインストールされている場合は、次のように入力します。

- Bourne、Bash または Korn シェルの場合

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

- C シェルの場合

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

Pro*COBOL の Oracle ランタイム・システム

Oracle では、動的読取り Pro*COBOL プログラムを実行するために、rtsora (64 ビット・システムの 32 ビット COBOL コンパイラの場合は rtsora32) という専用のランタイム・システムを用意しています。動的読取り Pro*COBOL プログラムを実行するには、cobrun ランタイム・システムのかわりに、rtsora (または rtsora32) ランタイム・システムを使用します。cobrun を使用して Pro*COBOL プログラムを実行すると、次のようなエラー・メッセージが表示されます。

```
$ 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 のデモ・プログラム

デモ・プログラムは、Pro*COBOL プリコンパイラの機能を紹介するために用意されています。デモ・プログラムは、\$ORACLE_HOME/precomp/demo/procob2 ディレクトリにあります。デフォルトでは、すべてのプログラムがクライアント共有ライブラリに動的にリンクされます。

プログラムを実行するには、\$ORACLE_HOME/sqlplus/demo/demobld.sql スクリプトで作成したデモンストレーション表が SCOTT スキーマにあり、そのパスワードが TIGER であることが必要です。

注意： デモンストレーションを作成する前に、SCOTT アカウントのロックを解除してパスワードを設定する必要があります。

次の Make ファイルを使用してデモ・プログラムを作成します。

```
$ORACLE_HOME/precomp/demo/procob2/demo_procob.mk
```

sample1 という Pro*COBOL 用のデモ・プログラムをプリコンパイル、コンパイルおよびリンクするには、次のように入力します。

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

Pro*COBOL のデモ・プログラムをすべて作成するには、次のように入力します。

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

rtsora ランタイム・システムで使用する動的読取りプログラム sample1.gnt を作成し、実行するには、次のように入力します。

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

一部のデモ・プログラムでは、\$ORACLE_HOME/precomp/demo/sql ディレクトリ内にある SQL スクリプトを実行する必要があります。このスクリプトを実行しないと、実行を要求するメッセージが表示されます。デモ・プログラムを作成し、それに対応する SQL スクリプトを実

行するには、「make マクロ引数 RUNSQL=run」をコマンドラインに追加します。たとえば、sample9 デモ・プログラムを作成し、必要な \$ORACLE_HOME/precomp/demo/sql/sample9.sql スクリプトを実行するには、次のように入力します。

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

Pro*COBOL デモ・プログラムをすべて作成し、必要な SQL スクリプトをすべて実行するには、次のように入力します。

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

Pro*COBOL のユーザー・プログラム

Make ファイル \$ORACLE_HOME/precomp/demo/procob2/demo_procob.mk を使用して、ユーザー・プログラムを作成できます。この Make ファイルは、使用しているプラットフォームに応じて、32 ビットまたは 64 ビットのユーザー・プログラムを作成します。一部のプラットフォームでは、Make ファイル demo_procob_32.mk を使用して、32 ビット（または 31 ビット）のユーザー・プログラムを作成することもできます。次の表に、各プラットフォームで Pro*COBOL により 32 ビット（または 31 ビット）および 64 ビットのユーザー・プログラムを作成するための Make ファイルを示します。

プラットフォーム	64 ビット Make ファイル	32 ビット Make ファイル
AIX、HP-UX および Solaris SPARC	demo_procob.mk	demo_procob_32.mk
Linux x86	該当なし	demo_procob.mk
Tru64 UNIX	demo_procob.mk	該当なし
zSeries Linux	demo_procob.mk	demo_procob_32.mk

注意：この Make ファイルは 31 ビットのユーザー・プログラムを作成します。

注意：ユーザー・プログラムの作成方法は、その Make ファイルを参照してください。

Make ファイル demo_procob.mk を使用してプログラムを作成する場合は、次のようなコマンドを入力します。

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

各項目の意味は次のとおりです。

- **target** は、使用する Make ファイルのターゲットです。
- **cobfilen** は、プログラムの COBOL ソース・ファイルです。
- **exename** は、実行可能プログラムです。

たとえば、プログラム myprog を作成する場合は、作成する実行可能ファイルのソースとタイプに応じて、次のいずれかのコマンドを入力します。

- COBOL ソースの場合、クライアント共有ライブラリに動的にリンクさせるには、次のように入力します。

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

- COBOL ソースの場合、静的にリンクさせるには、次のように入力します。

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

- COBOL ソースの場合、rtsora (32 ビットの COBOL コンパイラの場合は rtsora32) で使用するための動的読取りプログラムを作成するには、次のように入力します。

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

FORMAT プリコンパイラ・オプション

FORMAT プリコンパイラ・オプションは、COBOL の入力行の形式を指定します。デフォルト値の ANSI を指定した場合、列 1 ～ 6 はオプションの順序番号、列 7 はコメントまたは継続行を示します。段落名は列 8 ～ 11 で開始され、列 12 ～ 72 が文となります。

値 TERMINAL を指定した場合、列 1 ～ 6 は削除され、列 7 が左端の列になります。

Pro*FORTRAN プリコンパイラ

注意： Linux、Mac OS X または Solaris x86 では、Pro*FORTRAN はサポートされていません。

Pro*FORTRAN プリコンパイラを使用する場合は、事前に適切なバージョンのコンパイラがインストールされていることを確認してください。

関連項目： 各プラットフォームでサポートされるコンパイラのバージョンの詳細は、『Oracle Database インストレーション・ガイド』を参照してください。Pro*FORTRAN プリコンパイラおよびインタフェース機能の詳細は、『Pro*FORTRAN Supplement to the Oracle Precompiler Programmer's Guide』を参照してください。

Pro*FORTRAN のデモ・プログラム

デモ・プログラムは、Pro*FORTRAN プリコンパイラの機能を紹介するために用意されています。デモ・プログラムはすべて、\$ORACLE_HOME/precomp/demo/profor ディレクトリにあります。デフォルトでは、すべてのプログラムがクライアント共有ライブラリに動的にリンクされます。

プログラムを実行するには、\$ORACLE_HOME/sqlplus/demo/demobld.sql スクリプトで作成したデモンストレーション表が SCOTT スキーマにあり、そのパスワードが TIGER であることが必要です。

注意： デモンストレーションを作成する前に、SCOTT アカウントのロックを解除してパスワードを設定する必要があります。

デモ・プログラムを作成するには、\$ORACLE_HOME/precomp/demo/profor ディレクトリにある Make ファイル demo_profor.mk を使用します。たとえば、sample1 のデモ・プログラムをプリコンパイル、コンパイルおよびリンクするには、次のように入力します。

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

Pro*FORTRAN のデモ・プログラムをすべて作成するには、次のように入力します。

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

一部のデモ・プログラムでは、\$ORACLE_HOME/precomp/demo/sql ディレクトリ内にある SQL スクリプトを実行する必要があります。このスクリプトを実行しないと、実行を要求するメッセージが表示されます。デモ・プログラムを作成し、それに対応する SQL スクリプトを実行するには、「make マクロ引数 RUNSQL=run」をコマンドラインに追加します。たとえば、sample11 デモ・プログラムを作成し、必要な \$ORACLE_HOME/precomp/demo/sql/sample11.sql スクリプトを実行するには、次のように入力します。

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

Pro*FORTRAN デモ・プログラムをすべて作成し、必要な SQL スクリプトをすべて実行するには、次のように入力します。

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

Pro*FORTRAN のユーザー・プログラム

Make ファイル \$ORACLE_HOME/precomp/demo/profor/demo_profor.mk を使用して、ユーザー・プログラムを作成できます。この Make ファイルは、使用しているプラットフォームに応じて、32 ビットまたは 64 ビットのユーザー・プログラムを作成します。一部のプラットフォームでは、Make ファイル demo_profor_32.mk を使用して、32 ビットのユーザー・プログラムを作成することもできます。次の表に、各プラットフォームで Pro*FORTRAN により 32 ビットおよび 64 ビットのユーザー・プログラムを作成するための Make ファイルを示します。

プラットフォーム	64 ビット Make ファイル	32 ビット Make ファイル
AIX、HP-UX および Solaris SPARC	demo_procob.mk	demo_procob_32.mk
Tru64 UNIX	demo_procob.mk	該当なし

注意： ユーザー・プログラムの作成方法は、その Make ファイルを参照してください。

Make ファイル demo_proc.mk を使用してプログラムを作成する場合は、次のようなコマンドを入力します。

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

各項目の意味は次のとおりです。

- `target` は、使用する Make ファイルのターゲットです。
- `forfilen` は、プログラムの FORTRAN ソース・ファイルです。
- `exename` は、実行可能プログラムです。

たとえば、Pro*FORTRAN ソース・ファイル `myprog.pfo` からプログラム `myprog` を作成する場合は、作成する実行可能ファイルのタイプに応じて、次のいずれかのコマンドを入力します。

- 実行可能ファイルの場合、クライアント共有ライブラリに動的にリンクさせるには、次のように入力します。

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

- 実行可能ファイルの場合、静的にリンクさせるには、次のように入力します。

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

AIX の場合のみ : SQL*Module for Ada

SQL*Module for Ada を使用する場合は、事前に適切なバージョンのコンパイラがインストールされていることを確認してください。

関連項目： 必要なコンパイラのバージョンの詳細は、『Oracle Database インストール・ガイド for UNIX Systems』を参照してください。

SQL*Module for Ada の詳細は、『SQL*Module for Ada Programmer's Guide』を参照してください。

SQL*Module for Ada のデモ・プログラム

デモ・プログラムは、SQL*Module for Ada の機能を紹介するために用意されています。デモ・プログラムはすべて、`$ORACLE_HOME/precomp/demo/modada` ディレクトリにあります。デフォルトでは、すべてのプログラムがクライアント共有ライブラリに動的にリンクされます。

`ch1_drv` デモ・プログラムを実行するには、`$ORACLE_HOME/sqlplus/demo/demobld.sql` スクリプトで作成したデモンストレーション表が `SCOTT` スキーマにあり、そのパスワードが `TIGER` であることが必要です。

注意： デモンストレーションを作成する前に、`SCOTT` アカウントのロックを解除してパスワードを設定する必要があります。

`demcalsp` および `demohost` の各デモ・プログラムでは、大学のサンプル・データベースが `MODTEST` スキーマに存在している必要があります。適切な `make` コマンドを使用して、`MODTEST` スキーマを作成し、大学のサンプル・データベースをロードできます。

SQL*Module for Ada デモ・プログラムをすべて作成し、`MODTEST` ユーザーの作成に必要な SQL スクリプトを実行し、大学のサンプル・データベースを作成するには、次のように入力します。

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

デモ・プログラム (`demohost`) を 1 つのみ作成し、`MODTEST` ユーザーの作成に必要な SQL スクリプトを実行し、大学のサンプル・データベースを作成するには、次のように入力します。

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

SQL*Module for Ada デモ・プログラムをすべて作成し、大学のサンプル・データベースを再作成しない場合は、次のように入力します。

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

デモ・プログラム (`demohost`) を 1 つのみ作成し、大学のサンプル・データベースを再作成しない場合は、次のように入力します。

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

すべてのプログラムでは、Oracle Net の接続文字列または別名 `INST1_ALIAS` が定義されていて、適切な表が存在しているデータベースに接続できるものとします。

SQL*Module for Ada のユーザー・プログラム

Make ファイル `$ORACLE_HOME/precomp/demo/modada/demo_modada.mk` を使用して、ユーザー・プログラムを作成できます。Make ファイル `demo_modada.mk` を使用してユーザー・プログラムを作成する場合は、次のようなコマンドを入力します。

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

各項目の意味は次のとおりです。

- `modulen` は、コンパイル済の Ada オブジェクトです。
- `exename` は、実行可能プログラムです。
- `SQL*Module_arguments` は、SQL*Module に渡されるコマンドライン引数です。

関連項目： SQL*Module for Ada のユーザー・プログラムの詳細は、『SQL*Module for Ada Programmers Guide』を参照してください。

Oracle Call Interface と Oracle C++ Call Interface

Oracle Call Interface (OCI) または Oracle C++ Call Interface (OCCI) を使用する前に、適切なバージョンの C または C++ がインストール済であることを確認してください。

関連項目： 各プラットフォームでサポートされるコンパイラのバージョンの詳細は、『Oracle Database インストール・ガイド』を参照してください。OCI および OCCI の詳細は、『Oracle Call Interface プログラマーズ・ガイド』および『Oracle C++ Call Interface プログラマーズ・ガイド』を参照してください。

OCI と OCCI のデモ・プログラム

デモ・プログラムは、OCI および OCCI の機能を紹介するために用意されています。デモ・プログラムには、C および C++ プログラムの 2 種類があります。デモ・プログラムはすべて、`$ORACLE_HOME/rdbms/demo` ディレクトリにあります。デフォルトでは、すべてのプログラムがクライアント共有ライブラリに動的にリンクされます。

デモ・プログラムを実行するには、`$ORACLE_HOME/sqlplus/demo/demobld.sql` スクリプトで作成したデモンストレーション表が SCOTT スキーマにあり、そのパスワードが TIGER である必要があります。

注意： デモンストレーションを作成する前に、SCOTT アカウントのロックを解除してパスワードを設定する必要があります。

デモ・プログラムを作成するには、`$ORACLE_HOME/rdbms/demo` ディレクトリにある Make ファイル `demo_rdbms.mk` を使用します。たとえば、`cdemo1` のデモ・プログラムをプリコンパイル、コンパイルおよびリンクするには、次のコマンドを入力します。

```
$ make -f demo_rdbms.mk cdemo1
```

OCI の C デモ・プログラムをすべて作成するには、次のように入力します。

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

OCCI の C++ デモ・プログラムをすべて作成するには、次のように入力します。

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

OCI と OCCI のユーザー・プログラム

Make ファイル `$ORACLE_HOME/rdbms/demo/demo_rdbms.mk` を使用して、ユーザー・プログラムを作成できます。この Make ファイルは、使用しているプラットフォームに応じて、32 ビットまたは 64 ビットのユーザー・プログラムを作成します。一部のプラットフォームでは、Make ファイル `demo_rdbms32.mk` を使用して、32 ビットのユーザー・プログラムを作成することもできます。次の表に、各プラットフォームで Pro*FORTRAN により 32 ビットおよび 64 ビットのユーザー・プログラムを作成するための Make ファイルを示します。

プラットフォーム	64 ビット Make ファイル	32 ビット Make ファイル
AIX、HP-UX および Solaris SPARC	<code>demo_rdbms.mk</code>	<code>demo_rdbms32.mk</code>
Linux x86、Mac OS X および Solaris x86	該当なし	<code>demo_rdbms.mk</code>
Linux Itanium および Tru64 UNIX	<code>demo_rdbms.mk</code>	該当なし
zSeries Linux	<code>demo_rdbms.mk</code>	<code>demo_rdbms32.mk</code>

注意：この Make ファイルは 31 ビットのユーザー・プログラムを作成します。

注意：ユーザー・プログラムの作成方法は、その Make ファイルを参照してください。

Make ファイル `demo_rdbms.mk` を使用してプログラムを作成する場合は、次のようなコマンドを入力します。

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

この例の各項目の意味は、次のとおりです。

- `target` は、使用する Make ファイルのターゲットです。
- `objfilen` は、プログラムをリンクするためのオブジェクト・ファイルです。
- `exename` は、実行可能プログラムです。

たとえば、C ソース・ファイル `myprog.c` からプログラム `myprog` を作成する場合は、作成する実行可能ファイルのタイプに応じて、次のいずれかのコマンドを入力します。

- C ソースの場合、クライアント共有ライブラリに動的にリンクさせるには、次のように入力します。

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

- C ソースの場合、静的にリンクさせるには、次のように入力します。

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

たとえば、C++ ソース・ファイル `myprog.cpp` からプログラム `myprog` を作成する場合は、作成する実行可能ファイルのタイプに応じて、次のいずれかのコマンドを入力します。

- C++ ソースの場合、クライアント共有ライブラリに動的にリンクさせるには、次のように入力します。

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

- C++ ソースの場合、静的にリンクさせるには、次のように入力します。

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

64 ビット・ドライバを使用する Oracle JDBC/OCI プログラム (AIX、HP-UX、Solaris SPARC および zSeries Linux のみ)

注意： この項で説明する指示と Make ファイルを使用して、64 ビット・ドライバを使用する独自の JDBC/OCI ユーザー・プログラムを作成することもできます。

64 ビット・ドライバを使用する JDBC/OCI デモ・プログラムを実行するには、次の手順に従ってください。

1. 次の各ファイルについて、CLASSPATH 環境変数の先頭に \$ORACLE_HOME/jdbc/lib/ojdbc14.jar を追加します。


```

jdbc/demo/samples/jdbcoci/Makefile
jdbc/demo/samples/generic/Inheritance/Inheritance1/Makefile
jdbc/demo/samples/generic/Inheritance/Inheritance2/Makefile
jdbc/demo/samples/generic/Inheritance/Inheritance3/Makefile
jdbc/demo/samples/generic/JavaObject1/Makefile
jdbc/demo/samples/generic/NestedCollection/Makefile
      
```
2. \$ORACLE_HOME/jdbc/demo/samples/generic/Makefile ファイル内で、JAVA 変数と JAVAC 変数を変更し、プラットフォームに応じて JDK の位置を次のように指定します。
 - Solaris SPARC および HP-UX

JAVA と JAVAC を次のように変更し、-d64 フラグを指定します。

```

JAVA=${ORACLE_HOME}/java/bin/java -d64
JAVAC=${ORACLE_HOME}/java/bin/javac -d64
          
```
 - AIX および zSeries Linux

JAVA と JAVAC を次のように変更します。JDK14_HOME は、64 ビットの JDK がインストールされているディレクトリです。

```

JAVA=$(JDK14_HOME)/bin/java
JAVAC=$(JDK14_HOME)/bin/javac
          
```
3. jdbc/demo/samples/generic/Makefile ファイル内で、JAVA と JAVAC が定義されている箇所を除き、JDK14_HOME/bin/javac の出現箇所をすべて JAVAC で置き換え、JDK14_HOME/bin/java の出現箇所をすべて JAVA で置き換えます。
4. プラットフォームの共有ライブラリ・パスの環境変数を設定し、\$ORACLE_HOME/lib ディレクトリを含めます。

プラットフォーム	環境変数
AIX	LIBPATH
HP-UX および zSeries Linux	LD_LIBRARY_PATH
Solaris SPARC	LD_LIBRARY_PATH_64

カスタム Make ファイル

この章の製品別の項で説明したように、ユーザー・プログラムを作成する場合は、オラクル社が提供する Make ファイル `demo_product.mk` を使用してください。この Make ファイルを変更する場合、またはカスタム Make ファイルを使用する場合は、次の制限事項があります。

- Oracle ライブラリの順番は変更しないでください。リンク中にすべてのシンボルが解決されるように、Oracle ライブラリはリンク・ラインに 2 回以上追加されます。
Oracle ライブラリの順番が重要である理由は、次のとおりです。
 - Oracle ライブラリは相互に参照し合います。ライブラリ A の関数はライブラリ B の関数をコールし、ライブラリ B の関数はライブラリ A の関数をコールします。
 - HP-UX、Mac OS X および Tru64 UNIX の各リンカーは、1 パス・リンカーです。AIX、Linux および Solaris の各リンカーは、2 パス・リンカーです。
- リンク・ラインに独自のライブラリを追加する場合は、リンク・ラインの最初または最後に追加します。Oracle ライブラリ間にユーザー・ライブラリを入れないでください。
- `nmake` または `GNU make` などの `make` ユーティリティを使用する場合は、マクロおよび接頭辞の処理について、プラットフォームで提供されている `make` ユーティリティとの違いに注意してください。Oracle の Make ファイルは、プラットフォームの `make` ユーティリティによってテストおよびサポートされます。
- Oracle ライブラリの名前および内容は、リリース間で変更されることがあります。必要なライブラリを判断するには、現行のリリースで提供されている Make ファイル `demo_product.mk` を必ず使用してください。

未定義シンボルの修正

オラクル社が提供している `symfind` ユーティリティを使用すると、シンボルが定義されているライブラリまたはオブジェクト・ファイルの場所を確認する際に役立ちます。プログラムのリンク時に、未定義シンボルは一般的なエラーの 1 つとみなされ、次のようなエラー・メッセージが生成されます。

```
$ 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
```

このエラーは、参照するシンボルの定義をリンカーが検出できなかった場合に発生します。このエラー・メッセージが表示された場合は、シンボルが定義されているライブラリまたはオブジェクト・ファイルがリンク・ラインにあるかどうか、およびリンカーが検索しているファイルのディレクトリが正しいかどうかを確認します。

次の例は、`symfind` ユーティリティの出力です。このユーティリティを `sqlcex` シンボルの検索に使用しています。

```
$ symfind sqlcex

SymFind - Find Symbol <sqlcex> in <*>.*a, .o, .so
-----
Command:      /u01/app/oracle/product/10.1.0/bin/symfind sqlcex
Local Directory: /u01/app/oracle/product/10.1.0
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
```

マルチスレッド・アプリケーション

このリリースで提供されている Oracle ライブラリは、スレッド・セーフで、マルチスレッド・アプリケーションをサポートできます。

シグナル・ハンドラの使用

Oracle Database では、2 タスク通信にいくつかのシグナルが使用されます。シグナルはデータベースに接続するとユーザー・プロセスにインストールされ、切断すると削除されます。

表 6-5 に、Oracle Database で 2 タスク通信に使用されるシグナルを示します。

表 6-5 2 タスク通信に使用されるシグナル

シグナル	説明
SIGCLD	Oracle プロセスが終了すると、パイプ・ドライバは SIGCLD (SIGCHLD ともいいます) を使用します。SIGCLD シグナルは、UNIX カーネルによってユーザー・プロセスに送信されます。シグナル・ハンドラは、wait() ルーチンを使用して、サーバー・プロセスが異常終了しているかどうかを調べます。Oracle プロセスではなく、ユーザー・プロセスが SIGCLD を受け取ります。
SIGCONT	パイプ 2 タスク・ドライバが、バンド外ブレークをユーザー・プロセスから Oracle プロセスに送信する場合に、SIGCONT を使用します。
SIGINT	2 タスク・ドライバが、ユーザーの割り込み要求を検出する場合に、SIGINT を使用します。Oracle プロセスではなく、ユーザー・プロセスが SIGINT を受け取ります。
SIGIO	Oracle Net プロトコルが、ネットワーク・イベントの着信を示す場合に、SIGIO を使用します。
SIGPIPE	パイプ・ドライバが、通信チャネルのファイルの終わりを検出する場合に、SIGPIPE を使用します。パイプへの書き込み時に、読取りプロセスが存在していない場合、SIGPIPE シグナルが書き込みプロセスに送信されます。Oracle プロセスとユーザー・プロセスの両方が、SIGPIPE を受け取ります。SIGCLD は、SIGPIPE に似ていますが、ユーザー・プロセスのみに適用され、Oracle プロセスには適用されません。
SIGTERM	パイプ・ドライバが、ユーザー側から Oracle プロセスに割り込みシグナルを送る場合に、SIGTERM を使用します。ユーザーが割り込みキー ([Ctrl]+[C]) を押すと、このシグナルが送信されます。ユーザー・プロセスではなく、Oracle プロセスが SIGTERM を受け取ります。
SIGURG	Oracle Net TCP/IP ドライバが、バンド外ブレークをユーザー・プロセスから Oracle プロセスに送信する場合に、SIGURG を使用します。

表に記載されているシグナルは、すべてのプリコンパイラ・アプリケーションに影響します。Oracle プロセスへの接続時に、SIGCLD (または SIGCHLD) および SIGPIPE にシグナル・ハンドラを 1 つのみインストールできます。osnsui() ルーチンをコールして設定すると、複数のシグナル・ハンドラを SIGINT 用にインストールできます。SIGINT の場合、osnsui() および osncui() を使用して、シグナル受取りルーチンを登録および削除します。

また、必要に応じて、他のシグナルにもシグナル・ハンドラをインストールできます。Oracle プロセスに接続していない場合は、複数のシグナル・ハンドラをインストールできます。

例 6-1 に、シグナル・ルーチンおよび受取りルーチンの設定方法を示します。

例 6-1 シグナル・ルーチンおよび受取りルーチン

```
/* user side interrupt set */
word osnsui( /*_ word *handlep, void (*astp), char * ctx, _*/)
/*
** osnsui: Operating System dependent Network Set User-side Interrupt. Add an
** interrupt handling procedure astp. Whenever a user interrupt (such as a ^C)
** occurs, call astp with argument ctx. Put in *handlep handle for this
** handler so that it may be cleared with osncui. Note that there may be many
** handlers; each should be cleared using osncui. An error code is returned if
** an error occurs.
*/

/* user side interrupt clear */
word osncui( /*_ word handle _/ );
/*
** osncui: Operating System dependent Clear User-side Interrupt. Clear the
** specified handler. The argument is the handle obtained from osnsui. An error
** code is returned if an error occurs.
*/
```

例 6-2 に、アプリケーション・プログラムでの osnsui() および osncui() ルーチンの使用方法を示します。

例 6-2 osnsui() および osncui() ルーチンのテンプレート

```
/*
** 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 機能

Oracle XA は、X/Open Distributed Transaction Processing (DTP) XA インタフェースの Oracle 実装です。XA 標準には、トランザクション内の共有リソースへのアクセスを制御するリソース・マネージャ (Oracle など) 間や、トランザクションを監視および解決するトランザクション・サービス間の双方向インタフェースが規定されています。

Oracle Call Interface には、XA 機能があります。TP モニター XA アプリケーションを作成するときは、TP モニター・ライブラリ (シンボル `ax_reg` および `ax_unreg` を定義するライブラリ) が、リンク・ラインで Oracle Client 共有ライブラリより前に設定されていることを確認してください。このリンク制限は、XA の動的登録 (Oracle XA スイッチ `xaoswd`) を使用する場合にのみ必要です。

Oracle Database の XA コールは、クライアント共有ライブラリ (プラットフォームに応じて `libclntsh.a`、`libclntsh.sl`、`libclntsh.so` または `libclntsh.dylib`) およびクライアント静的ライブラリ (`libclntst10.a`) の両方で定義されています。これらのライブラリは、`$ORACLE_HOME/lib` ディレクトリにあります。

SQL*Loader および PL/SQL のデモ

この章では、Oracle Database 10g とともに使用できる SQL*Loader と PL/SQL の各デモ・プログラムを作成および実行する方法について説明します。次の項目について説明します。

- [SQL*Loader のデモ](#)
- [PL/SQL のデモ](#)
- [PL/SQL からの 32 ビット外部プロシージャのコール](#)

注意： この章で説明するデモンストレーションを使用するには、Oracle Database 10g Companion CD に収録されている Oracle Database Examples をインストールする必要があります。

また、デモンストレーションを作成する前に、SCOTT アカountのロックを解除してパスワードを設定する必要があります。

SQL*Loader のデモ

SQL*Loader のデモをすべて実行する場合は、`ulcase.sh` ファイルを実行します。デモを個別に実行する場合は、ファイル内に含まれている情報を読んで、実行方法を確認してください。

PL/SQL のデモ

PL/SQL には、ロードできるデモ・プログラムが多数含まれています。デモ・プログラムを操作するには、Oracle Database 10g をオープン状態にしてマウントする必要があります。

これらのプログラムを使用する前に、データベース・オブジェクトを作成し、サンプル・データをロードする必要があります。オブジェクトを作成してサンプル・データをロードするには、次の手順を実行します。

1. ディレクトリを PL/SQL デモ・ディレクトリに変更します。

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

2. SQL*Plus を起動し、SCOTT/TIGER として接続します。

```
$ sqlplus SCOTT/TIGER
```

3. 次のコマンドを入力し、オブジェクトを作成してサンプル・データをロードします。

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

```
SQL> @exemplod.sql
```

注意： デモは、十分な権限を持つ Oracle ユーザーとして作成してください。
デモは、作成時と同じ Oracle ユーザーで実行してください。

PL/SQL カーネル・デモ

次の PL/SQL カーネル・デモを使用できます。

- `examp1.sql` ～ `examp8.sql`
- `examp11.sql` ～ `examp14.sql`
- `sample1.sql` ～ `sample4.sql`
- `extproc.sql`

PL/SQL カーネル・デモ `exampn.sql` または `samp1en.sql` をコンパイルして実行するには、次の手順に従ってください。

1. SQL*Plus を起動し、SCOTT/TIGER として接続します。

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

```
$ sqlplus SCOTT/TIGER
```

2. 次のようなコマンドを入力してデモを実行します。`demo_name` はデモ名です。

```
SQL> @demo_name
```

extproc.sql デモを実行するには、次の手順に従ってください。

1. 必要に応じて、次のように、外部プロシージャのエントリを tnsnames.ora ファイルに追加します。

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

2. 必要な場合は、オペレーティング・システムに応じて、listener.ora ファイルに外部プロシージャ用の次のようなエントリを追加します。

注意： listener.ora ファイルの SID_NAME に指定する値と、tnsnames.ora ファイルの SID に指定する値は、一致している必要があります。

■ HP-UX、Linux、Solaris および Tru64 UNIX

```
SID_LIST_LISTENER =
  (SID_LIST =
    (SID_DESC=
      (SID_NAME=PLSExtProc)
      (ORACLE_HOME=oracle_home_path)
      (ENVS=EXTPROC_DLLS=oracle_home_path/plsql/demo/extproc.so,
        LD_LIBRARY_PATH=oracle_home_path/plsql/demo)
      (PROGRAM=extproc)
    )
  )
```

■ AIX

```
SID_LIST_LISTENER =
  (SID_LIST =
    (SID_DESC=
      (SID_NAME=PLSExtProc)
      (ORACLE_HOME=oracle_home_path)
      (ENVS=EXTPROC_DLLS=oracle_home_path/plsql/demo/extproc.so,
        LIBPATH=oracle_home_path/plsql/demo)
      (PROGRAM=extproc)
    )
  )
```

■ Mac OS X:

```
SID_LIST_LISTENER =
  (SID_LIST =
    (SID_DESC=
      (SID_NAME=PLSExtProc)
      (ORACLE_HOME=oracle_home_path)
      (ENVS=EXTPROC_DLLS=oracle_home_path/plsql/demo/extproc.dylib,
        DYLD_LIBRARY_PATH=oracle_home_path/plsql/demo)
      (PROGRAM=extproc)
    )
  )
```

3. ディレクトリを \$ORACLE_HOME/plsql/demo に変更します。
4. 次のコマンドを入力して、extproc.so 共有ライブラリを作成し、必要なデータベース・オブジェクトを構築して、サンプル・データをロードします。

注意： MacOS X システムでは、次の各例において共有ライブラリ名 extproc.dylib を使用してください。

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

データベース・オブジェクトが構築済で、サンプル・データがロードされている場合は、次のコマンドを入力します。

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

5. SQL*Plus から、次のコマンドを入力します。

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

6. デモを実行するには、次のコマンドを入力します。

```
SQL> @extproc
```

PL/SQL プリコンパイラ・デモ

注意： この項で示す make コマンドは、必要なデータベース・オブジェクトを作成し、サンプル・データを SCOTT スキーマにロードします。

次のプリコンパイラ・デモを利用できます。

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

PL/SQL プリコンパイラ・デモをすべて作成するには、\$ORACLE_HOME/lib ディレクトリを含むようにライブラリ・パス環境変数を設定し、次のコマンドを入力します。

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

デモを 1 つのみ作成する場合は、make コマンドにそのデモ名を引数として指定します。たとえば、examp9 デモを作成するには、次のように入力します。

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

examp9 デモを実行するには、次のコマンドを入力します。

```
$ ./examp9
```

PL/SQL からの 32 ビット外部プロシージャのコール

注意： この項の内容は、AIX、HP-UX、Solaris SPARC および zSeries Linux にのみ適用されます。

64 ビットの外部プロシージャ実行可能ファイル (extproc) および 32 ビットの外部プロシージャ実行可能ファイル (extproc32) は、\$ORACLE_HOME/bin ディレクトリにインストールされています。デフォルトでは、実行可能ファイル extproc を使用して、AIX、HP-UX、Solaris SPARC および zSeries Linux で、64 ビット外部プロシージャを実行できます。32 ビットの外部プロシージャを使用可能にするには、次の手順を実行します。

1. listener.ora ファイルに PROGRAM パラメータの値を次のように設定します。

(PROGRAM=extproc32)

2. 使用しているプラットフォームに応じて、次のいずれかの環境変数に \$ORACLE_HOME/lib32 ディレクトリを指定します。

プラットフォーム	環境変数
AIX	LIBPATH
HP-UX	SHLIB_PATH
Solaris SPARC および zSeries Linux	LD_LIBRARY_PATH

3. リスナーを再起動します。

注意： 32 ビットまたは 64 ビットのいずれかの外部プロシージャを実行できるようにリスナーを構成できますが、同時に両方のプロシージャを実行することはできません。ただし、32 ビットと 64 ビットの外部プロシージャを両方ともサポートする必要がある場合は、2 つのリスナーを構成できます。

UNIX での Oracle Database のチューニング

この章では、UNIX システムで Oracle Database をチューニングする方法について説明します。
次の項目について説明します。

- [チューニングの重要性](#)
- [オペレーティング・システムのツール](#)
- [メモリー管理のチューニング](#)
- [ディスク I/O のチューニング](#)
- [ディスク・パフォーマンスの監視](#)
- [システム・グローバル領域](#)
- [オペレーティング・システムのバッファ・キャッシュのチューニング](#)

チューニングの重要性

Oracle Database は、高度に最適化できるソフトウェア製品です。チューニングを頻繁に行うことで、システム・パフォーマンスが最適化され、データのボトルネックの発生を防ぐことができます。この章の説明は単一ノード・システムを対象に書かれていますが、記載されているパフォーマンスのチューニングに関するヒントの多くは、Oracle Real Application Clusters のインストールに適用できます。

システムのチューニングを始める前に、8-2 ページの「オペレーティング・システムのツール」で説明するツールを使用して、通常の動作を監視します。

オペレーティング・システムのツール

データベースのパフォーマンスを評価し、データベース要件を決定するのに役立つオペレーティング・システム・ツールがいくつかあります。これらのツールは、Oracle プロセスの統計に加えて、システム全体の CPU 使用率、割込み、スワッピング、ページング、コンテキストのスイッチング、I/O についての統計情報も提供します。

共通のツール

次の各項では、次に示す共通のツールについて説明します。

- [vmstat](#)
- [sar](#)
- [iostat](#)
- [swap](#)、[swapinfo](#)、[swapon](#)、[lsps](#)

関連項目： これらのツールの詳細は、オペレーティング・システムのドキュメントおよび UNIX の [man](#) ページを参照してください。

vmstat

注意： MacOS X では、`vm_stat` コマンドを使用すると、仮想メモリ情報が表示されます。このコマンドの使用方法は、`vm_stat` の [man](#) ページを参照してください。

プロセス、仮想メモリ、ディスク、トラップおよび CPU アクティビティを表示するときは、`vmstat` コマンドを使用します。表示内容は、コマンドで切り替えます。CPU アクティビティのサマリーを 5 秒間隔で 6 回表示する場合は、次のいずれかのコマンドを入力します。

- HP-UX および Solaris


```
$ vmstat -S 5 6
```
- AIX、Linux および Tru64 UNIX


```
$ vmstat 5 6
```

次の例は、Solaris SPARC でこのコマンドを実行した場合の出力です。

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

procs 列の下にある w 列は、スワップ・アウトされて、ディスクに書き込まれたプロセスの数を示します。値が 0（ゼロ）以外の場合は、スワッピングが発生し、システムがメモリー不足になっています。

page 列の下にある si 列および so 列は、それぞれ 1 秒当たりのスワップ・インとスワップ・アウトの回数を示します。スワップ・インとスワップ・アウトは、常に 0（ゼロ）にしてください。

page 列の下にある sr 列は、スキャン率を示します。利用できるメモリーが不足すると、スキャン率が高くなります。

page 列の下にある pi 列および po 列は、それぞれ 1 秒当たりのページインとページアウトの回数を示します。ページインおよびページアウトの回数は通常、増加します。利用できるメモリーが十分にあるシステムでも、常に多少のページングは行われます。

注意： vmstat コマンドの出力は、プラットフォーム間で異なります。使用しているプラットフォームでの出力の解釈については、man ページを参照してください。

sar

オペレーティング・システムのアクティビティ・カウンタの累計を表示するときは、sar (system activity reporter) コマンドを使用します。表示内容はコマンドで切り替えます。Solaris システムでは、次のコマンドは、I/O アクティビティのサマリーを 10 秒間隔で 10 回表示します。

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

次の例は、Solaris でこのコマンドを実行した場合の出力です。

```
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
```

注意： Tru64 UNIX システムでは、sar コマンドは、UNIX SVID2 互換サブセットの OSF5VID で利用できます。

sar 出力は、ある時点におけるシステムの I/O アクティビティのスナップショットを提供します。複数のオプションを使用して時間隔を指定すると、出力の読み取りができなくなることがあります。時間隔を 4 以下に指定すると、sar アクティビティ自体が出力に影響を与えることがあります。sar の詳細は、man ページを参照してください。

iostat

端末およびディスクのアクティビティを表示するときは、`iostat` コマンドを使用します。表示内容は、コマンドで切り替えます。`iostat` コマンドの出力には、ディスク要求キューは表示されず、ビジー状態のディスクが表示されます。この情報は、I/O 負荷のバランスを調整する場合に役立ちます。

次のコマンドは、端末およびディスク・アクティビティを 5 秒間隔で 5 回表示します。

```
$ iostat 5 5
```

次の例は、Solaris でこのコマンドを実行した場合の出力です。

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

大きなディスク要求キューを調べるときは、`iostat` コマンドを使用します。要求キューは、特定のディスク・デバイスに対する I/O 要求が実行されるまでにかかる時間を示します。要求キューが発生する原因は、特定のディスクに対する I/O 要求のボリュームが大きいこと、または I/O の平均シーク時間が長いことです。ディスク要求キューは、0（ゼロ）またはそれに近い値であることが理想的です。

swap、swapinfo、swapon、lsps

注意： Mac OS X システムのスワップ領域の詳細は、D-2 ページの「[使用可能および使用済みのスワップ領域の決定](#)」の項を参照してください。

スワップ領域の使用量に関する情報を表示するときは、`swap`、`swapinfo`、`swapon` または `lsps` の各コマンドを使用します。スワップ領域が不足すると、プロセスの応答が停止し、メモリ不足エラーでプロセスが生成できなくなることがあります。次の表は、プラットフォームごとの適切なコマンドの一覧です。

プラットフォーム	コマンド
AIX	<code>lsps -a</code>
HP-UX	<code>swapinfo -m</code>
Linux	<code>swapon -s</code>
Solaris	<code>swap -l</code> および <code>swap -s</code>
Tru64 UNIX	<code>swapon -s</code>

次の例は、Solaris で `swap -l` コマンドを実行した場合のサンプル出力です。

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

AIX のツール

次の各項では、AIX システムで利用できるツールについて説明します。

関連項目： これらのツールの詳細は、AIX オペレーティング・システムのドキュメントおよび man ページを参照してください。

AIX System Management Interface Tool

AIX System Management Interface Tool (SMIT) は、様々なシステム管理およびパフォーマンス・ツールに対して、メニュー方式のインタフェースを提供します。SMIT を使用すると、実行するジョブを中心に、様々なツールにナビゲートできます。

Base Operation System ツール

AIX の Base Operation System (BOS) は、UNIX システムに由来組み込まれていたパフォーマンス・ツールや、AIX の実装固有の機能を管理するパフォーマンス・ツールで構成されています。次の表は、最も重要な BOS ツールの一覧です。

ツール	説明
lsattr	デバイスの属性を表示します。
lslv	論理ボリューム、つまり物理ボリュームに対する論理ボリュームの割当てに関する情報を表示します。
netstat	ネットワーク関連のデータ構造の内容を表示します。
nfsstat	ネットワーク・ファイル・システム (NFS) とリモート・プロシージャ・コール (RPC) のアクティビティに関する統計を表示します。
nice	プロセスの初期優先順位を変更します。
no	ネットワーク・オプションを表示または設定します。
ps	1 つまたは複数のプロセスのステータスを表示します。
reorgvg	ボリューム・グループ内の物理パーティション割当てを再編成します。
time	経過した実行時間、ユーザーの CPU 処理時間およびシステムの CPU 処理時間を表示します。
trace	選択したシステム・イベントを記録および報告します。
vmo	Virtual Memory Manager のチューニング可能なパラメータを管理します。

AIX Performance Toolbox

AIX Performance Toolbox (PTX) には、システム・アクティビティをローカルおよびリモートで監視およびチューニングするためのツールがいくつか含まれています。PTX は、PTX Manager と PTX Agent という 2 つのコンポーネントで主に構成されています。PTX Manager は、xmperf ユーティリティを使用して、構成内の様々なシステムからデータを収集し、表示します。PTX Agent は、xmserd デーモンを使用して、データを収集し PTX Manager に転送します。PTX Agent は、Performance Aide for AIX と呼ばれる製品として個別に利用することもできます。

PTX と Performance Aide の両方には、次の監視およびチューニング・ツールがあります。

ツール	説明
fdpr	特定の作業負荷にあわせて実行可能プログラムを最適化します。
filemon	トレース機能を使用して、ファイル・システムのアクティビティを監視および報告します。
fileplace	論理ボリュームまたは物理ボリューム内の特定のファイルについて、ブロックの配置を表示します。

ツール	説明
lockstat	カーネル・ロックの競合に関する統計を表示します。
lvedit	ボリューム・グループ内の論理ボリュームを対話方式で配置します。
netpmon	トレース機能を使用して、ネットワーク I/O およびネットワーク関連の CPU 使用率を報告します。
rmss	様々なサイズのメモリーでシステムをシミュレートし、パフォーマンスをテストします。
svmon	仮想メモリーの使用量に関する情報を取得し、分析します。
syscalls	システム・コールを記録し、カウントします。
tprof	トレース機能を使用して、モジュールとソース・コード文の各レベルで CPU 使用率を報告します。
BigFoot	プロセスのメモリー・アクセス・パターンを報告します。
stem	サブルーチン・レベルのエントリを許可し、既存の実行可能ファイルのインストルメント処理を終了します。

関連項目： 各ツールの詳細は、『Performance Toolbox for AIX Guide and Reference』を参照してください。一部のツールの構文は、『AIX 5L Performance Management Guide』を参照してください。

HP-UX のツール

次の各項では、HP-UX システムで利用できるツールについて説明します。

パフォーマンス・チューニング・ツール

次の表は、HP-UX でパフォーマンス・チューニングの追加に使用できるツールの一覧です。

関連項目： これらのツールの詳細は、HP-UX オペレーティング・システムのドキュメントおよび man ページを参照してください。

ツール	説明
caliper (Itanium のみ)	高速の動的インストルメント処理とともに、キャッシュ・ミス、Translation Look-aside Buffer (TLB)、命令サイクルなどのシステム分析タスク用にランタイム・アプリケーション・データを収集します。C、C++、Fortran およびアセンブリ・アプリケーション用の動的パフォーマンス測定ツールです。
gprof	プログラムの実行プロファイルを作成します。
monitor	プログラム・カウンタを監視し、特定の関数をコールします。
netfmt	ネットワークを監視します。
netstat	ネットワーク・パフォーマンスに関する統計を報告します。
nfsstat	ネットワーク・ファイル・システム (NFS) とリモート・プロシージャ・コール (RPC) のアクティビティに関する統計を表示します。
nettl	ロギングとトレースによって、ネットワーク・イベントまたはパケットを取得します。
prof	C プログラムの実行プロファイルを作成してプログラムのパフォーマンス統計を表示し、そのプログラムが実行時間の大半を消費している箇所を示します。
profil	プログラム・カウンタ情報をバッファにコピーします。
top	システムの上位プロセスを表示し、その情報を定期的に更新します。

HP-UX パフォーマンス分析ツール

次の HP-UX パフォーマンス分析ツールは、HP-UX システムでも使用できます。

- [GlancePlus/UX](#)
- [HP PAK](#)

GlancePlus/UX

この HP-UX ユーティリティは、システムのアクティビティを測定するためのオンライン診断ツールです。GlancePlus は、システム・リソースの使用状況を表示します。つまり、システムの I/O、CPU およびメモリー使用量に関する動的な情報を一連の画面に表示します。このユーティリティを使用して、プロセスごとのリソースの使用状況を監視することもできます。

HP PAK

HP Programmer's Analysis Kit (HP PAK) は現在、Puma および Thread Trace Visualizer (TTV) という 2 つのツールで構成されています。

- Puma は、プログラムの実行時にパフォーマンス統計を収集します。収集した統計をグラフィカルに表示し、分析できるようにします。
- TTV は、インストルメント処理スレッド・ライブラリ `libpthreads_tr.sl` によって作成されるトレース・ファイルをグラフィカルに表示します。このツールを使用すると、スレッドの相互作用を表示したり、スレッドがブロックされてリソースを待機している箇所を調べることができます。

HP PAK は、HP Fortran 77、HP Fortran 90、HP C、HP C++、HP ANSI C++ および HP Pascal の各コンパイラに組み込まれています。

Linux のツール

Linux システムでは、`top`、`free` および `cat /proc/meminfo` コマンドを使用して、スワップ領域、メモリーおよびバッファの使用量を表示します。

Mac OS X のツール

Mac OS X システムでは、次のパフォーマンス・チューニング・ツールを追加で使用できます。

- `top` コマンドを使用して、実行中のプロセスおよびメモリー使用量を表示します。
- Shark や BigTop などの Apple Computer Hardware Understanding Developer (CHUD) ツールを使用して、システム・アクティビティの監視およびアプリケーションのチューニングをします。

CHUD ツールの詳細は、次の Web サイトを参照してください。

<http://developer.apple.com/documentation/Performance/Conceptual/PerformanceFundamentals/index.html>

Solaris のツール

Solaris システムでは、`mpstat` コマンドを使用して、マルチプロセッサ・システムのプロセッサごとの統計を表示します。表の各行は、1 プロセッサのアクティビティを示します。1 行目は、起動してからのすべてのアクティビティをまとめて表示します。2 行目以降の各行は、時間隔でのアクティビティをまとめて表示します。特に指定しないかぎり、すべての値は 1 秒当たりのイベント数です。引数は、統計および反復回数の時間隔を示します。

次の例は、Solaris で `mpstat` コマンドを実行した場合のサンプル出力です。

```
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
```

メモリー管理のチューニング

メモリー・チューニング・プロセスでは、最初にページングおよびスワッピング領域を測定して、使用可能なメモリー量を確認します。システムのメモリー使用量の確認後、Oracle バッファ・キャッシュをチューニングします。

Oracle バッファ・マネージャによって、アクセス頻度の高いデータをキャッシュに長く保存できます。バッファ・マネージャを監視し、バッファ・キャッシュをチューニングすると、Oracle Database のパフォーマンスが大幅に向上することがあります。各システムの Oracle Database バッファ・サイズの最適値は、システム全体の負荷や他のアプリケーションと比較した場合の Oracle の優先順位によって異なります。

十分なスワップ領域の割当て

スワッピングは、UNIX のオーバーヘッドに大きく影響するため、最小限に抑える必要があります。スワッピングが行われているかどうかを調べるには、`sar` コマンドまたは `vmstat` コマンドを使用します。この 2 つのコマンドで使用するオプションについては、`man` ページを参照してください。

システムでスワッピングが行われている場合は、メモリーを節約するために、次の処理を行います。

- 必要以上にシステム・デーモン・プロセスまたはアプリケーション・プロセスを実行しないようにします。
- データベース・バッファの数を減らし、一部のメモリーを解放します。
- UNIX ファイル・バッファの数を減らします（特に RAW デバイスを使用する場合）。

注意： Mac OS X システムでは、スワップ領域は動的に割り当てられます。オペレーティング・システムでスワップ領域の追加が必要な場合は、`/private/var/vm` ディレクトリに追加スワップ・ファイルが作成されます。このディレクトリが含まれるファイル・システムに、追加スワップ・ファイルに対応できるだけの十分な空きディスク領域があることを確認してください。

スワップ領域の使用量を確認するには、使用しているプラットフォームに応じて、次のいずれかのコマンドを入力します。

プラットフォーム	コマンド
AIX	<code>lspv -a</code>
HP-UX	<code>swapinfo -m</code>
Linux	<code>swapon -s</code>
Solaris	<code>swap -l</code> および <code>swap -s</code>
Tru64 UNIX	<code>swapon -s</code>

スワップ領域をシステムに追加するには、使用しているプラットフォームに応じて、次のいずれかのコマンドを入力します。

プラットフォーム	コマンド
AIX	<code>chps</code> または <code>mkps</code>
HP-UX	<code>swapon</code>
Linux	<code>swapon -a</code>
Solaris	<code>swap -a</code>
Tru64 UNIX	<code>swapon -a</code>

スワップ領域は、システムの物理メモリーの2～4倍に設定してください。スワップ領域の使用量を監視し、必要に応じて値を大きくしてください。

関連項目： これらのコマンドの詳細は、使用しているオペレーティング・システムのドキュメントを参照してください。

ページングの制御

プログラムを実行するためにプログラム全体をメモリーに格納しておく必要はないため、ページングはスワッピングほど深刻な問題ではありません。少量のページアウトでは、システムのパフォーマンスにほとんど影響はありません。

大量のページングを検出するには、高速応答時またはアイドル時の測定値と、低速応答時の測定値を比較します。

ページングを監視するには、vmstat (Mac OS X では vm_stat) または sar コマンドを使用します。プラットフォームの監視結果の解釈については、man ページまたはオペレーティング・システムのドキュメントを参照してください。Solaris では、これらのコマンドの出力から得られる次の列が重要です。

列	説明
vflt/s	アドレス変換ページ・フォルトの数を示します。アドレス変換フォルトは、プロセスが、メモリー内にない有効ページを参照したときに発生します。
rc1m/s	ページアウト・アクティビティによって再生され、空きリストに追加された有効ページ数を示します。この値は0（ゼロ）である必要があります。

システムで大量のページアウト・アクティビティが常に発生している場合は、次の方法で解決してください。

- メモリーを増設します。
- 一部の作業を別のシステムに移します。
- SGA で使用するメモリーを少なく設定します。

Oracle ブロック・サイズの調整

UNIX システムでは、オペレーティング・システムのブロック全体をディスクから読み取ります。データベースのブロック・サイズが、UNIX ファイル・システムのブロック・サイズより小さい場合は、I/O 帯域幅の効率が悪くなります。Oracle データベースのブロック・サイズをファイル・システムのブロック・サイズの倍数になるように設定すると、パフォーマンスを最大5パーセント向上させることができます。

データベースのブロック・サイズは、DB_BLOCK_SIZE 初期化パラメータで設定します。ただし、このパラメータの値を変更するには、データベースを再作成する必要があります。

DB_BLOCK_SIZE パラメータの現在の設定値を調べるには、SQL*Plus の SHOW PARAMETER DB_BLOCK_SIZE コマンドを入力します。

ディスク I/O のチューニング

使用可能なディスク全体で I/O を均等に分散して、ディスクへのアクセス時間が短くなるようにしてください。小規模なデータベースや RAID を使用しないデータベースでは、それぞれのデータファイルと表領域を使用可能なディスク間に分散してください。

Automatic Storage Management の使用

データベース記憶域に Automatic Storage Management を使用すると、すべてのデータベース I/O が、ASM ディスク・グループ内の使用可能なすべてのディスク・デバイス間に分散されます。ASM では、RAW デバイスを管理する煩わしさがなく、RAW デバイス I/O のパフォーマンスが向上します。

ASM を使用することで、ディスク I/O を手動でチューニングする必要がなくなります。

適切なファイル・システム・タイプの選択

使用するオペレーティング・システムに応じて、いくつかのファイル・システム・タイプから選択できます。ファイル・システム・タイプごとにそれぞれ特性が異なり、それらがデータベースのパフォーマンスに大きな影響を与えます。次の表に、UNIX プラットフォームで利用できる一般的なファイル・システムを示します。

ファイル・システム	プラットフォーム	説明
S5	AIX、HP-UX、Solaris	UNIX System V ファイル・システム
UFS	AIX、HP-UX、Mac OS X、Solaris、Tru64 UNIX	Unified ファイル・システム (BSD UNIX から派生) 注意: Mac OS X システムでは、ソフトウェアおよびデータベース・ファイルのいずれにも UFS ファイルの使用はお勧めしません。
VxFS	AIX、Solaris、HP-UX	VERITAS ファイル・システム
なし	すべて	RAW デバイス (ファイル・システムなし)
ext2/ext3	Linux	Linux 用拡張ファイル・システム
AdvFS	Tru64 UNIX	Advanced ファイル・システム
CFS	Tru64 UNIX	クラスタ・ファイル・システム
JFS/JFS2	AIX	ジャーナル・ファイル・システム
HFS Plus、HFSX	Mac OS X	HFS Plus は、Mac OS X により使用される標準の階層ファイル・システムです。HFSX は HFS Plus の拡張で、大 / 小文字を区別したファイル名を使用できます。
GPFS	AIX	一般的なパラレル・ファイル・システム
OCFS	Linux	Oracle Cluster ファイル・システム

ファイル・システムとアプリケーションには、必ずしも互換性があるとはかぎりません。たとえば、Unified ファイル・システムの各実装間でさえ、互換性の比較は容易ではありません。選択したファイル・システムによって、パフォーマンスに 0 ~ 20 パーセントの開きが出る場合があります。ファイル・システムを使用する場合は、次のことを行ってください。

- ハードディスクがクリーンで断片化されないように、ファイル・システムのパーティションを新しく作成します。
- データベース・ファイルに対してファイル・システムを使用する前に、パーティションでファイル・システム・チェックを行います。
- ディスク I/O をできるだけ均等に分散します。
- 論理ボリューム・マネージャまたは RAID デバイスを使用していない場合、データファイルと異なるファイル・システムにログ・ファイルを格納することを検討してください。

ディスク・パフォーマンスの監視

次の各項では、ディスク・パフォーマンスの監視方法について説明します。

Mac OS X でのディスク・パフォーマンスの監視

Mac OS X システムでディスク・パフォーマンスを監視するには、`iostat` および `sar` コマンドを使用します。これらのコマンドの使用方法は、`man` ページを参照してください。

その他のオペレーティング・システムでのディスク・パフォーマンスの監視

ディスク・パフォーマンスを監視するには、`sar -b` および `sar -u` コマンドを使用します。

表 8-1 に、`sar -b` コマンド出力の列をいくつか示します。これらの列は、ディスク・パフォーマンスの分析に重要です。

表 8-1 `sar -b` 出力の列

列	説明
<code>bread/s</code> , <code>bwrit/s</code>	1 秒ごとに読み取られるブロック数と書き込まれるブロック数 (ファイル・システム・データベースに重要)
<code>pread/s</code> , <code>pwrit/s</code>	1 秒ごとに読み取られるパーティション数と書き込まれるパーティション数 (RAW パーティション・データベース・システムに重要)

ディスク・パフォーマンスの分析に重要な `sar -u` 列の 1 つに、`%wio` があります。これによって、ブロックされた I/O で待機する CPU 時間の割合がわかります。

注意： 一部の Linux では、`sar -u` コマンドの出力結果に `%wio` 列が表示されません。詳細な I/O 統計を表示するには、`iostat -x` コマンドを使用できます。

キー・インジケータは次のとおりです。

- `bread`、`bwrit`、`pread` および `pwrit` 列の合計は、ディスク I/O サブシステムのアクティビティのレベルを示します。合計値が大きいほど、I/O サブシステムはビジーになります。物理ドライブの数が多いほど、合計のしきい値が高くなる可能性があります。デフォルトの最適値は、ドライブ 2 個の場合は 40 以下、ドライブ 4～8 個の場合は 60 以下です。
- `%rcache` 列の値は 91 以上、`%wcache` 列の値は 61 以上である必要があります。それ以外の場合は、システムがディスク I/O バウンドになる可能性があります。
- `%wio` 列の値が常に 21 以上である場合、システムは I/O バウンドになります。

システム・グローバル領域

システム・グローバル領域（SGA）とは、共有メモリーに格納されている Oracle 構造体のことです。この構造体には、静的データ構造体、ロックおよびデータ・バッファが含まれています。各 Oracle プロセスが SGA 全体をアドレス指定するためには、十分な共有メモリーが必要です。

1 つの共有メモリー・セグメントの最大サイズは、shmmax カーネル・パラメータ（Tru64 UNIX では shm_max）で指定します。次の表に、このパラメータの推奨値をプラットフォームごとに示します。

プラットフォーム	推奨値
AIX	該当なし。
HP-UX	システムに搭載された物理メモリーのサイズ。 関連項目： HP-UX の shmmax パラメータの詳細は、B-2 ページの「 Oracle インスタンス用の HP-UX 共有メモリー・セグメント 」を参照してください。
Linux	システムに搭載された物理メモリーの半分のサイズ。
Mac OS X	該当なし。Mac OS X での SGA の最大サイズは 1000 MB。
Solaris	4294967295 または 4GB マイナス 16MB。64 ビットのシステムでは 4GB より大きくすることも可能。
Tru64 UNIX	4294967295 または 4GB マイナス 16MB。 注意： Oracle インスタンスを起動する場合は、shm_max パラメータの値を 16MB 以上にする必要があります。システムで Oracle9i と Oracle Database 10g の両方のインスタンスが動作している場合は、このパラメータの値を 2GB マイナス 16MB に設定する必要があります。

SGA のサイズが共有メモリー・セグメントの最大サイズ（shmmax または shm_max）を超える場合、Oracle Database 10g では、要求された SGA サイズになるように、連続したセグメントが連結されます。shmseg カーネル・パラメータ（Tru64 UNIX では shm_seg）には、任意のプロセスで連結できるセグメントの最大数を指定します。SGA のサイズを制御するには、次の初期化パラメータを設定します。

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

または、Oracle で SGA サイズを自動的にチューニングできるように SGA_TARGET 初期化パラメータを設定します。

これらのパラメータの値は、十分注意して設定してください。値が大きすぎると、システムの物理メモリーに対する共有メモリーの割合が大きくなりすぎて、パフォーマンスが低下します。

共有サーバーで構成されている Oracle データベースでは、SHARED_POOL_SIZE 初期化パラメータの値を大きく設定するか、LARGE_POOL_SIZE 初期化パラメータを使用したカスタム構成が必要です。Oracle Universal Installer を使用してデータベースをインストールした場合、SHARED_POOL_SIZE パラメータの値は、Database Configuration Assistant によって自動的に設定されます。ただし、データベースを手動で作成した場合は、パラメータ・ファイルで SHARED_POOL_SIZE パラメータの値を同時ユーザーごとに 1KB ずつ増やしてください。

SGA サイズの確認

次のいずれかの方法で、SGA サイズを確認できます。

- 次の SQL*Plus コマンドを入力して、実行中のデータベースの SGA サイズを表示します。

```
SQL> SHOW SGA
```

結果はバイト単位で表示されます。

- データベース・インスタンスを起動するときに、SGA のサイズを確認します。SGA サイズは、ヘッダー「Total System Global Area」の横に表示されます。
- Mac OS X 以外のシステムでは、oracle ユーザーとして `ipcs` コマンドを入力します。

AIX での共有メモリー

AIX では、共有メモリーとしてプロセス間で共通の仮想メモリー・リソースが使用されます。各プロセスは、パフォーマンスを向上させるために、表やキャッシュ・エントリなどの共通の仮想メモリー変換リソース・セットを介して仮想メモリー・セグメントを共有します。

AIX 上の Oracle Database では、ページングを回避し、I/O のオーバーヘッドを減らすために、共有メモリーを確保できます。これを行うには、`LOCK_SGA` パラメータを `true` に設定します。AIX 5L の場合、基礎となるハードウェアでラージ・ページ機能がサポートされていれば、同じパラメータによってその機能がアクティブになります。

確保されたメモリーを AIX システムの Oracle Database で利用できるようにするには、次のコマンドを入力します。

```
$ /usr/sbin/vmo -r -o v_pinshm=1
```

確保されたメモリーに利用できる実メモリーの最大量を設定するには、次のようなコマンドを入力します。`percent_of_real_memory` は、設定する実メモリーの最大量（パーセント）です。

```
$ /usr/sbin/vmo -r -o maxpin%=percent_of_real_memory
```

`maxpin%` オプションを使用しているときは、確保されたメモリーの量が Oracle SGA のサイズに対して、システムの実メモリーの 3 パーセント以上であることが重要です。これにより、確保可能な空きメモリーをカーネルが使用できます。たとえば、物理メモリーが 2GB であり、SGA を 400MB（RAM の 20 パーセント）確保する場合は、次のコマンドを入力します。

```
$ /usr/sbin/vmo -r -o maxpin%=23
```

システムの操作時に確保されたメモリーの使用量を監視するには、`svmon` コマンドを使用します。Oracle Database は、`LOCK_SGA` パラメータが `true` に設定されている場合にのみメモリーを確保しようとします。

AIX POWER4-Based Systems でのラージ・ページ機能

POWER4 システム上でサイズがそれぞれ 16MB のラージ・ページを 10 件オンにして予約するには、次のコマンドを入力します。

```
$ /usr/sbin/vmo -r -o lgpg_regions=10 -o lgpg_size=16777216
```

このコマンドは、`bosboot` を提案し、変更を有効にするにはリブートする必要があることを示す警告を表示します。

SGA 全体を格納できるラージ・ページを指定することをお勧めします。`LOCK_SGA` パラメータが `true` に設定されている場合、Oracle インスタンスではラージ・ページを割り当てようとします。SGA のサイズが確保可能なメモリー・サイズまたはラージ・ページのサイズを超える場合は、これらのサイズを超える SGA 部分が通常の共有メモリーに割り当てられます。

関連項目： 確保されたメモリーやラージ・ページの有効化およびチューニングについては、AIX のドキュメントを参照してください。

オペレーティング・システムのバッファ・キャッシュのチューニング

RAW デバイスを最大限に活用するには、Oracle Database バッファ・キャッシュのサイズを調整し、メモリーに制限がある場合は、オペレーティング・システムのバッファ・キャッシュのサイズも調整します。

オペレーティング・システムのバッファ・キャッシュには、メモリーからディスクまたはディスクからメモリーへの転送中に、メモリー内のデータ・ブロックが保持されます。

Oracle Database バッファ・キャッシュは、Oracle データベース・バッファを格納するためのメモリー内の領域です。Oracle Database では、RAW デバイスを使用できるため、オペレーティング・システムのバッファ・キャッシュを使用する必要はありません。

RAW デバイスを使用する場合は、Oracle Database バッファ・キャッシュのサイズを大きくします。システムで使用できるメモリーに制限がある場合は、それに応じてオペレーティング・システムのバッファ・キャッシュのサイズを小さくします。

調整するバッファ・キャッシュを判断するには、sar コマンドを使用します。sar コマンドの詳細は、man ページを参照してください。

注意： Tru64 UNIX の場合は、オペレーティング・システムのバッファ・キャッシュを小さくしないでください。これは、オペレーティング・システムでは、ファイル・システム I/O のバッファリングに必要なメモリー量が自動的にサイズ変更されるためです。オペレーティング・システムのバッファ・キャッシュを制限すると、パフォーマンスの問題が発生する可能性があります。

AIX システムでの Oracle Database の管理

この付録では、AIX システムで Oracle Database を管理する方法について説明します。次の項目について説明します。

- [メモリーとページング](#)
- [ディスク I/O の問題](#)
- [CPU のスケジューリングおよびプロセスの優先順位](#)
- [Oracle Real Application Clusters に関する情報](#)
- [AIXTHREAD_SCOPE 環境変数の設定](#)

メモリーとページング

メモリーの競合は、プロセスに必要なメモリー量が、利用できる容量よりも大きくなったときに発生します。このようなメモリー不足に対処するために、メモリーとディスクとの間でプログラムやデータのページングが行われます。

バッファ・キャッシュのページング・アクティビティの制御

ページング・アクティビティが過剰に行われると、パフォーマンスが大幅に低下します。ジャーナル・ファイル・システム（JFS および JFS2）上に作成されたデータベース・ファイルでは、このページングが問題になることがあります。この状況では、大量の SGA データ・バッファ内に、参照頻度の最も高いデータを含む、似たようなジャーナル・ファイル・システムのバッファが存在する可能性があります。AIX ファイル・バッファ・キャッシュ・マネージャの動作は、パフォーマンスに大きく影響します。この動作によっては、I/O ボトルネックが発生し、システム全体のスループットが低下する可能性があります。

AIX では、バッファ・キャッシュ・ページング・アクティビティをチューニングできますが、回数を絞って十分注意して行う必要があります。/usr/samples/kernel/vmtune コマンドを使用して、次の AIX システム・パラメータをチューニングします。

パラメータ	説明
minfree	空きリスト・サイズの最小値。バッファ内の空きリストの容量がこのサイズを下回ると、ページ・スティーリングによって、空きリストが補充されます。
maxfree	空きリスト・サイズの最大値。バッファ内の空きリストの容量がこのサイズを上回ると、ページ・スティーリングによる空きリストの補充が中止されます。
minperm	ファイル I/O の永続バッファ・ページの最小数。
maxperm	ファイル I/O の永続バッファ・ページの最大数。

関連項目： AIX システム・パラメータの詳細は、『AIX 5L Performance Management Guide』を参照してください。

AIX ファイル・バッファ・キャッシュのチューニング

AIX ファイル・バッファ・キャッシュの目的は、ジャーナル・ファイル・システム使用時のディスクへのアクセス頻度を減少させることです。このキャッシュが小さすぎると、ディスク使用率が増加して、1 つ以上のディスクが一杯になることがあります。このキャッシュが大きすぎると、メモリーが無駄になります。

関連項目： AIX ファイル・バッファ・キャッシュの増加に伴う影響については、A-2 ページの「[バッファ・キャッシュのページング・アクティビティの制御](#)」を参照してください。

AIX ファイル・バッファ・キャッシュは、minperm および maxperm パラメータの調整によって構成できます。一般に、sar -b コマンドで指定するバッファ・ヒット率が低い（90 パーセント未満）場合は、minperm パラメータ値を増やします。バッファ・ヒット率を高くする必要がない場合は、minperm パラメータ値を減らして、利用できる物理メモリーを増やします。AIX ファイル・バッファ・キャッシュのサイズの増加については、AIX のドキュメントを参照してください。

パフォーマンスの向上は、マルチプログラミングの量や作業負荷の I/O 特性によって変化するため、簡単には数値化できません。

minperm および maxperm パラメータのチューニング

AIX では、ファイルに使用されるページ・フレームと、計算可能（作業用またはプログラム・テキスト）セグメントに使用されるページ・フレームとの比率を、大まかに制御できます。次のガイドラインに従って、minperm および maxperm の値を調整してください。

- 実メモリー内でのファイル・ページの占有率が minperm 値を下回る場合は、再ページ率に関係なく、Virtual Memory Manager (VMM) のページ置換アルゴリズムによって、ファイル・ページおよび計算可能ページが奪われます。
- 実メモリー内でのファイル・ページの占有率が maxperm 値を上回る場合は、Virtual Memory Manager のページ置換アルゴリズムによって、ファイル・ページおよび計算可能ページが奪われます。
- 実メモリー内でのファイル・ページの占有率がパラメータ値 minperm と maxperm の間にある場合は、Virtual Memory Manager によって、通常はファイル・ページのみが奪われます。ただし、ファイル・ページの再ページ率が計算可能ページの再ページ率よりも大きい場合は、計算可能ページも同様に奪われます。

デフォルト値を計算するには、次のアルゴリズムを使用します。

- $\text{minperm (ページ数)} = ((\text{ページ・フレームの数}) - 1024) \times 0.2$
- $\text{maxperm (ページ数)} = ((\text{ページ・フレームの数}) - 1024) \times 0.8$

minperm パラメータの値を総ページ・フレーム数の 5 パーセントに変更し、maxperm パラメータの値を総ページ・フレーム数の 20 パーセントに変更するには、次のコマンドを使用します。

```
# /usr/samples/kernel/vmtune -p 5 -P 20
```

デフォルト値は、それぞれ 20 パーセントと 80 パーセントです。

新しいデータベース接続をオープンする際の即時応答を最適化するには、minfree パラメータを調節して、システムが空きリストにページを追加せずにアプリケーションをメモリーにロードできるように、十分な空きページを確保します。プロセスの実メモリー・サイズ（常駐セット・サイズ、作業用セット）を判断するには、次のコマンドを使用します。

```
$ ps v process_id
```

この値または 8 フレームのいずれか大きい値を minfree パラメータに設定します。

データベース・ファイルが RAW デバイス上にある場合、またはダイレクト I/O を使用している場合は、minperm および maxperm パラメータをそれぞれ 5 パーセントと 20 パーセントなど、低い値に設定できます。これは、RAW デバイスまたはダイレクト I/O では、AIX ファイル・バッファ・キャッシュが使用されないためです。メモリーは、Oracle システム・グローバル領域など、他の目的に有効に利用できます。

十分なページング領域の割当て（スワップ領域）

ページング領域（スワップ領域）が十分に割り当てられていないと、システムが停止したり、応答時間が非常に遅くなります。AIX では、RAW ディスク・パーティションにページング領域を動的に追加できます。設定するページング領域の大きさは、実装されている物理メモリーの量およびアプリケーションのページング領域要件によって異なります。ページング領域の使用量を監視するには、lsps コマンドを使用します。システムのページング・アクティビティを監視するには、vmstat コマンドを使用します。ページング領域を増やすには、smit pgsp コマンドを使用します。

ページング領域が事前に割り当てられるプラットフォームでは、ページング領域を RAM の量よりも大きい値に設定することをお勧めします。AIX では、ページング領域は必要になるまで割り当てられません。システムでは、実メモリーが不足した場合にのみスワップ領域が使用されます。メモリーのサイズを正しく設定した場合は、ページングが行われないため、ページング領域を小さくできます。要求されるページ数が増減しない場合は、ページング領域が小さくてもシステムは適切に動作します。ページングが大幅に増加する可能性がある場合は、その最大ページ数を処理できるページング領域が必要になります。

一般に、ページング領域の初期設定は、32GB を上限として、RAM の半分のサイズに 4GB を加えた値です。lsps -a コマンドを使用してページング領域を監視し、その結果に従ってページング領域のサイズを増減します。lsps -a コマンドによって出力される使用率 (%Used) は、適正なシステムの場合は通常 25 パーセント未満になります。メモリー・サイズが適切に割り当てられている場合、ページング領域はほとんど必要ありません。また、スワッピングが過剰に発生する場合は、システムに対して RAM サイズが小さすぎる可能性があります。

注意： ページング領域のサイズは小さくしないでください。サイズを小さくしたことによって領域が不足すると、アクティブなプロセスが終了する可能性があります。一方、ページング領域が大きすぎたとしてもマイナスの影響はほとんどありません。

ページングの制御

過剰なページングが頻繁に発生する場合は、実メモリーが不足していることを意味します。通常は、次のように対処してください。

- ページングが頻繁に発生しないようにします。または、システムに高速の拡張記憶域を装備し、メモリーと拡張記憶域間のページングが、SGA とディスク間のデータの読取り / 書き込み速度よりも十分に速くなるようにします。
- 制限されたメモリー・リソースを、システム・パフォーマンスが最も向上する場所に割り当てます。場合によっては、メモリー・リソース要件とその影響のバランスを調整するために、この処理を繰り返し行う必要があります。
- メモリーが不足している場合は、メモリーを必要とするシステム内のプロセスと要素を優先順に並べたリストを作成します。次に、パフォーマンスが最も向上する場所にメモリーを割り当てます。優先順リストの例を次に示します。
 1. OS および RDBMS のカーネル
 2. ユーザー・プロセスおよびアプリケーション・プロセス
 3. REDO ログ・バッファ
 4. PGA および共有プール
 5. データベース・ブロック・バッファ・キャッシュ

たとえば、Oracle の動的パフォーマンス表およびビューを問い合わせた結果、共有プールおよびデータベース・バッファ・キャッシュにメモリーを追加する必要があるとします。この場合、制限された予備メモリーは、データベース・ブロック・バッファ・キャッシュではなく、共有プールに割り当てたほうがパフォーマンスが向上します。

次の AIX コマンドを実行すると、ページングの状況および統計が表示されます。

- `vmstat -s`
- `vmstat interval [repeats]`
- `sar -r interval [repeats]`

データベース・ブロック・サイズの設定

Oracle データベース・ブロック・サイズを設定することによって、I/O スループットを改善できます。AIX では、DB_BLOCK_SIZE 初期化パラメータの値を 2 ～ 32KB に設定できます。デフォルト値は 4KB です。Oracle Database がジャーナル・ファイル・システムにインストールされている場合は、そのブロック・サイズをファイル・システムのブロック・サイズ (JFS では 4KB、GPFS では 16KB ～ 1MB) の倍数にする必要があります。データベースが RAW パーティション上にある場合は、Oracle データベース・ブロック・サイズをオペレーティング・システムの物理ブロック・サイズ (AIX では 512 バイト) の倍数にします。

Oracle データベース・ブロック・サイズは、オンライン・トランザクション処理 (OLTP) または複合作業負荷の環境では小さめ (2 または 4KB) に設定し、意思決定支援システム (DSS) 作業負荷環境では大きめ (8、16 または 32KB) に設定することをお勧めします。

ログ・アーカイブ・バッファのチューニング

トランザクションが長い場合やトランザクションの数が多い場合は特に、LOG_BUFFER サイズを増やすことで、データベースのアーカイブ速度を向上させることができます。ログ・ファイル I/O アクティビティおよびシステム・スループットを監視して、最適な LOG_BUFFER サイズを判断します。LOG_BUFFER パラメータをチューニングするときは、通常のデータベース・アクティビティの全体的なパフォーマンスが低下しないように注意してください。

注意： LOG_ARCHIVE_BUFFER_SIZE パラメータは、Oracle8i で廃止されました。

I/O バッファおよび SQL*Loader

SQL*Loader ダイレクト・パス・オプションを使用しながらデータを並行してロードする場合など、データを高速にロードするときは、CPU 時間の大半が I/O 完了の待ち時間として使用されます。通常は、バッファの数を増やすことにより、CPU 使用率が上がり、スループット全体も向上します。

選択するバッファの数（SQL*Loader の BUFFERS パラメータで設定）によって、使用可能なメモリー量や CPU 使用率の上昇程度は異なります。BUFFERS パラメータのファイル処理オプション文字列の調整については、『Oracle Database ユーティリティ』を参照してください。

パフォーマンスの向上は、CPU 使用率やデータのロード時に使用する並列度によって変化します。

関連項目： SQL*Loader ユーティリティの概要は、『Oracle Database ユーティリティ』を参照してください。

インポート・ユーティリティ用の BUFFER パラメータ

インポート・ユーティリティ用の BUFFER パラメータには、高速ネットワークのパフォーマンスを最適化するために、大きい値を設定する必要があります。たとえば、IBM RS/6000 Scalable POWERparallel Systems (SP) スイッチを使用する場合は、BUFFER パラメータの値を 1MB 以上に設定する必要があります。

ディスク I/O の問題

ディスク I/O の競合は、メモリー管理が良好でない場合（その結果として発生するページングおよびスワッピングを含む）や、ディスク間の表領域とファイルの配分が適切でない場合に発生します。

filemon、sar、iostat などの AIX ユーティリティや、その他のパフォーマンス・ツールを使用して、I/O 負荷が高いディスクを識別し、すべてのディスク・ドライブに I/O 負荷を均等に分散します。

AIX 論理ボリューム・マネージャ

AIX 論理ボリューム・マネージャ (LVM) は、複数のディスクにデータをストライプ化して、ディスクの競合を軽減できます。ストライプ化の主な目的は、大容量の順次ファイルに対する読取り / 書込みのパフォーマンスを向上させることです。LVM のストライプ機能を効果的に使用すると、ディスク間に I/O を均等に分散できるため、全体的なパフォーマンスが向上します。

注意： Automatic Storage Management (ASM) ディスク・グループには論理ボリュームを追加しないでください。ASM は、ディスク・グループに RAW ディスク・デバイスが追加された場合に最も効果を発揮します。ASM を使用している場合、LVM によるストライプ化は行わないでください。Automatic Storage Management は、ストライプ化およびミラー化を実装しません。

ストライプ化された論理ボリュームの設計

ストライプ化された論理ボリュームを定義するときは、次の項目を指定する必要があります。

項目	推奨される設定値
ドライブ	2つ以上の物理ドライブ。パフォーマンスが重視される順次 I/O を実行するときは、2つ以上の物理ドライブのアクティビティを最小にする必要があります。場合によっては、複数のアダプタ間で論理ボリュームをストライプ化する必要があります。
ストライプ・ユニット・サイズ	ストライプ・ユニット・サイズには、2 ～ 128KB の範囲で 2 の累乗を指定できます。ただし、ほとんどの作業負荷には、32KB と 64KB のストライプ・サイズで十分です。Oracle データベース・ファイルでは、ストライプ・サイズをデータベース・ブロック・サイズの倍数にする必要があります。
サイズ	論理ボリュームに割り当てる物理パーティションの数は、使用するディスク・ドライブ数の倍数にする必要があります。
属性	ミラー化することはできません。copies 属性の値を 1 に設定します。

その他の考慮事項

LVM を使用したときのパフォーマンスの向上度は、使用する LVM や作業負荷の特性によって大きく異なります。DSS 作業負荷では、パフォーマンスが大幅に向上します。OLTP タイプまたは複合作業負荷の場合も、かなりのパフォーマンスの向上を期待できます。

ジャーナル・ファイル・システムを使用した場合と RAW 論理ボリュームを使用した場合の相違

ジャーナル・ファイル・システムと RAW 論理ボリュームのどちらを使用するかを決定する場合は、次のことを考慮してください。

- ファイル・システムは、実装の多様化に伴い、継続的に改善されています。場合によっては、RAW デバイスよりもファイル・システムを使用したほうが I/O パフォーマンスが向上することがあります。
- ファイル・システムでは追加構成（AIX の minservers および maxservers パラメータ）が必要であり、ファイル・システムの非同期 I/O がカーネルの外部で実行されるため、CPU オーバーヘッドが少し増加します。
- 様々なベンダーが、各ディスクの特長を生かすために、様々な方法でファイル・システム・レイヤーを実装しています。この結果、プラットフォーム間でのファイル・システムの比較が難しくなっています。
- 強力な LVM インタフェースを導入すると、RAW 論理ボリュームに基づく論理ディスクの設定やバックアップ作業が大幅に減少します。
- AIX 5L に組み込まれているダイレクト I/O およびコンカレント I/O 機能によって、ファイル・システムのパフォーマンスは、RAW 論理ボリュームに匹敵するレベルまで向上します。

ジャーナル・ファイル・システムを使用した場合は、RAW デバイスを使用した場合と比較して、データベース・ファイルの管理や保守が容易になります。以前のバージョンの AIX では、ファイル・システムはバッファに対する読取り / 書込みのみサポートしており、inode ロックが不完全なために余計な競合が発生していました。この 2 つの問題は、JFS2 コンカレント I/O 機能および GPFS ダイレクト I/O 機能によって解決されており、最高のパフォーマンスが必要な場合でも、RAW デバイスのかわりにファイル・システムを使用できます。

注意： Oracle Real Application Clusters オプションを使用するには、ASM ディスク・グループ内の RAW デバイスまたは GPFS ファイル・システムにデータファイルを配置する必要があります。JFS または JFS2 は使用できません。GPFS を使用すると、ダイレクト I/O が暗黙的に有効になります。

ファイル・システム・オプション

AIX 5L では、ダイレクト I/O およびコンカレント I/O がサポートされています。ダイレクト I/O およびコンカレント I/O のサポートによって、データベース・ファイルがファイル・システム上に存在できるようになります。これは、Oracle Database が提供する機能を使用して、オペレーティング・システムのバッファ・キャッシュを回避し、冗長な inode ロック操作を排除することによって実現されます。

Oracle のデータファイルが含まれるファイル・システムでは、コンカレント I/O またはダイレクト I/O をできるかぎり有効にすることをお勧めします。次の表に、AIX で使用可能なファイル・システムとその推奨設定を示します。

ファイル・システム	オプション	説明
JFS	dio	JFS ではコンカレント I/O は使用できません。ダイレクト I/O (dio) を使用できます。ただし、コンカレント I/O を使用した JFS2 に比べてパフォーマンスは劣ります。
JFS ラージ・ファイル	なし	128KB の位置合せ制約によってダイレクト I/O の使用が回避されるため、サイズの大きい JFS ファイルを Oracle Database に使用することはお勧めしません。
JFS2	cio	コンカレント I/O は同一のファイルに対して複数のコンカレント・リーダー / ライターをサポートしているため、JFS2 では、ダイレクト I/O (dio) よりもコンカレント I/O (cio) を設定するほうが有効です。
GPFS	N/A	Oracle Database は、最適なパフォーマンスとなるように GPFS に対してダイレクト I/O を暗黙的に有効にします。GPFS のダイレクト I/O は、すでに複数のノード上の複数のリーダー / ライターをサポートしています。したがって、GPFS の場合、ダイレクト I/O とコンカレント I/O は同じです。

JFS および JFS2 の考慮事項

JFS2 ファイル・システムに Oracle Database ログを置いている場合、agblksize=512 オプションを使用してファイル・システムを作成し、cio オプションでこれをマウントすると最適な構成となります。これによって、ロギング・パフォーマンスが RAW デバイスのパフォーマンスの数パーセント以内に抑えられます。

Oracle Database 10g より前のバージョンでは、JFS/JFS2 においてファイル・レベルでダイレクト I/O またはコンカレント I/O (あるいはその両方) を有効にできませんでした。したがって、最適なパフォーマンスを得るために、Oracle ホーム・ディレクトリおよびデータファイルを独立したファイル・システムに配置し、ファイル・システムに配置された Oracle ホーム・ディレクトリをデフォルト・オプションでマウントし、ファイル・システム上のデータファイルおよびログを dio または cio オプションを使用してマウントする必要がありました。

Oracle Database 10g では、JFS/JFS2 において、ダイレクト I/O またはコンカレント I/O (あるいはその両方) を個々のファイル・レベルで有効にできます。そのためには、サーバー・パラメータ・ファイルの FILESYSTEMIO_OPTIONS パラメータを、setall (デフォルト) または directIO に設定します。これにより、すべてのデータファイル I/O について、JFS2 でのコンカレント I/O と JFS でのダイレクト I/O が有効になります。その結果、Oracle ホーム・ディレクトリと同じ JFS/JFS2 ファイル・システムにデータファイルを配置できます。前述のように、最適なパフォーマンスを得るには、Oracle Database ログを独立した JFS2 ファイル・システムに配置する必要があります。

GPFS の考慮事項

GPFS を使用している場合は、Oracle ホーム・ディレクトリ、データファイルおよびログなどすべての目的に同じファイル・システムを使用できます。最適なパフォーマンスを得るためには、大規模な GPFS ブロック・サイズ (通常は 512KB 以上) を使用してください。GPFS はスケーラビリティを確保するように設計されており、データ量が単一の GPFS ファイル・システムに収まるかぎり、複数の GPFS ファイル・システムを作成する必要はありません。

ジャーナル・ファイル・システムから RAW 論理ボリュームへの移動

すべてのデータを手動で再ロードせずに、ジャーナル・ファイル・システムから RAW デバイスにデータを移動するには、次の操作を root ユーザーで実行します。

1. 新規 RAW 論理ボリューム・デバイス・タイプ (-T O) を使用して、RAW デバイス (BigVG を推奨) を作成します。これにより、最初の Oracle ブロックをオフセット 0 (ゼロ) にでき、最適なパフォーマンスを得ることができます。

```
# mklv -T O -y new_raw_device VolumeGroup NumberOfPartitions
```

注意： この新規 RAW デバイスは、既存のファイルより大きくする必要があります。新規 RAW デバイスのサイズが領域を無駄にしないよう、注意してください。

2. RAW デバイスに対する権限を設定します。
3. 次のように、dd を使用して JFS ファイルの内容を変換し、新規 RAW デバイスにコピーします。

```
# dd if=old_JFS_file of=new_raw_device bs=1m
```

4. データファイルの名前を変更します。

RAW 論理ボリュームからジャーナル・ファイル・システムへの移動

RAW 論理ボリュームの最初の Oracle ブロックが必ずしもオフセット 0 (ゼロ) ではないのに対し、ファイル・システム上の最初の Oracle ブロックは常にオフセット 0 (ゼロ) です。オフセットを決定し、RAW 論理ボリュームの最初のブロックを検索するには、\$ORACLE_HOME/bin/offset コマンドを使用します。オフセットは、AIX 論理ボリュームでは 4096 バイトまたは 128KB となり、mklv -T O オプションにより作成された AIX 論理ボリュームでは 0 (ゼロ) となります。

オフセットの決定後、dd コマンドを使用し、オフセットをスキップして、RAW 論理ボリュームからファイル・システムにデータをコピーできます。次の例では、オフセットを 4096 バイトと仮定しています。

```
# dd if=old_raw_device bs=4k skip=1|dd of=new_file bs=256
```

RAW 論理ボリュームの最大容量よりも小さいブロック数を使用するように、Oracle Database に指示できます。このように指示する場合、count 句を追加して、必ず Oracle ブロックが含まれるデータのみをコピーする必要があります。次の例では、オフセットが 4096 バイト、Oracle ブロック・サイズが 8KB、ブロック数が 150000 であると仮定しています。

```
# dd if=old_raw_device bs=4k skip=1|dd bs=8k count=150000|dd of=new_file bs=256k
```

非同期 I/O の使用

Oracle Database では、AIX が提供する非同期 I/O（AIO）を最大限に利用して、データベース・アクセスの高速化を図っています。

AIX 5L は、ファイル・システム・パーティションおよび RAW デバイスで作成されたデータベース・ファイルに対して、非同期 I/O（AIO）をサポートしています。RAW デバイスに対する AIO は、AIX カーネル内に完全に実装されるため、サーバーのプロセスで AIO 要求を処理する必要はありません。ファイル・システムに対して AIO を使用するときは、要求がキューから取り出されてから処理が完了するまで、カーネル・サーバー・プロセス（aioserver）が各要求を制御します。カーネル・サーバー・プロセスは、FastPath が無効になっている仮想共有ディスク（VSD）や HSD による I/O にも使用されます。デフォルトでは、FastPath は有効になっています。aioserver サーバーの数によってシステムで同時に実行できる AIO 要求の数が決まるため、Oracle Database のデータファイルの格納にファイル・システムを使用する場合には、aioserver プロセスの数をチューニングすることが重要です。

注意： AIO FastPath が有効になっている（デフォルト）VSD や HSD で AIO を使用する場合は、最大バディ・バッファ・サイズを 128KB 以上に設定する必要があります。

サーバー数を設定するには、次のいずれかのコマンドを使用します。このコマンドは、RAW デバイスではなくファイル・システムで非同期 I/O を使用している場合にのみ適用されます。

- `smit aio`
- `chdev -l aio0 -a maxservers='m' -a minservers='n'`

関連項目： システム管理インタフェース・ツール（SMIT）の詳細は、SMIT のオンライン・ヘルプを参照してください。smit aio および chdev の各コマンドの詳細は、man ページを参照してください。

注意： AIX 5L リリース 5.2 以降は、2 つの AIO サブシステムを使用できます。Oracle Database 10g では、Oracle プリインストール・スクリプトにより LEGACY AIO（aio0）および POSIX AIO（posix_aio0）が有効になる場合でも、LEGACY AIO（aio0）が使用されます。どちらの AIO サブシステムの場合もパフォーマンス特性は同じです。

最小値には、システム・ブート時に起動するサーバーの数を指定します。最大値には、多数の同時要求に応答するために起動できるサーバーの数を指定します。これらのパラメータは、ファイル・システムにのみ適用され、RAW デバイスには適用されません。

サーバーの最小数のデフォルト値は 1 です。最大数のデフォルト値は 10 です。カーネル化された AIO を使用していない場合、CPU を 4 個以上搭載する大規模なシステムで Oracle Database を実行するには、通常、これらの値では低すぎます。次の表に示されている値に設定することをお勧めします。

パラメータ	値
minservers	初期値としては、システム上の CPU の数または 10 のうち、いずれか小さい値を設定することをお薦めします。
maxservers	AIX 5L リリース 5.2 からは、このパラメータで CPU ごとの AIO サーバーの最大数がカウントされるようになりましたが、以前のバージョンの AIX では、システム全体の値がカウントされていました。GPFS を使用している場合は、maxservers を、CPU の数で割った worker1threads に設定します。これが最適な設定です。maxservers の値を大きくしても、I/O のパフォーマンスは向上しません。JFS または JFS2 を使用している場合、初期値を (10 × 論理ディスク数 ÷ CPU 数) に設定し、pstat または ps コマンドを使用して、典型的な作業負荷で起動された aioservers の実際の数を監視します。実際にアクティブな AIO サーバー数が maxservers と等しい場合は、maxservers の値を増加します。
maxreqs	初期値を (4 × 論理ディスク数 × キューの深さ) に設定します。キューの深さは次のコマンドを実行することによって判断できます。通常は 3 です。 \$ lsattr -E -l hdiskxx

maxservers または maxreqs パラメータの値が低すぎると、次の警告メッセージが繰り返し表示される場合があります。

```
Warning: lio_listio returned EAGAIN
Performance degradation may be seen.
```

これらのエラーが表示されないようにするには、maxservers パラメータの値を大きくします。稼動している AIO サーバーの数を表示するには、次のコマンドを root ユーザーで入力します。

```
# pstat -a | grep -c aios
# ps -k | grep aioserver
```

アクティブな AIO サーバーの数を定期的にチェックし、必要に応じて minservers および maxservers パラメータの値を変更してください。パラメータの変更は、システムが再起動されるときに有効になります。

I/O スレーブ

I/O スレーブは、I/O のみを実行する特別な Oracle プロセスです。非同期 I/O がデフォルトであり、AIX に対する I/O 操作は Oracle での実行が推奨されている方法であるため、I/O スレーブは AIX ではほとんど使用されません。I/O スレーブは、共有メモリー・バッファから割り当てられます。I/O スレーブには、次の表に示す一連の初期化パラメータを使用します。

パラメータ	許容値	デフォルト値
DISK_ASYNC_IO	true/false	true
TAPE_ASYNC_IO	true/false	true
BACKUP_TAPE_IO_SLAVES	true/false	false
DBWR_IO_SLAVES	0 ~ 999	0
DB_WRITER_PROCESSES	1 ~ 20	1

通常、この表のパラメータを調整する必要はありません。ただし、作業負荷が大きい場合に、データベース・ライターがボトルネックになることがあります。その場合は、DB_WRITER_PROCESSES を大きくします。このデータベース・ライター・プロセスの数は、システムまたはパーティション内で CPU2 つにつき 1 つですが、原則として、この数は増やさないでください。

デバッグのためにオラクル社カスタマ・サポート・センターにより指示された場合など、非同期 I/O の無効化が必要な場合があります。DISK_ASYNC_IO および TAPE_ASYNC_IO の各パラメータを使用して、ディスクまたはテープ・デバイスに対する非同期 I/O を無効にできます。各プロセス・タイプの I/O スレーブ数のデフォルトは 0（ゼロ）であるため、デフォルトでは I/O スレーブは割り当てられません。

DBWR_IO_SLAVES パラメータは、DISK_ASYNC_IO または TAPE_ASYNC_IO パラメータが false の場合にのみ、0（ゼロ）より大きい値に設定します。設定しないと、データベース・ライター・プロセス（DBWR）がボトルネックになります。この場合、AIX での DBWR_IO_SLAVES パラメータの最適値は 4 です。

DB_FILE_MULTIBLOCK_READ_COUNT パラメータの使用

Oracle Database 10g では、デフォルトでダイレクト I/O またはコンカレント I/O が使用されるため（使用可能な場合）、ファイル・システムでは順次スキャンでの先読みは実行されません。DB_FILE_MULTIBLOCK_READ_COUNT 初期化パラメータの指定どおり、Oracle Database により先読みが実行されます。

DB_FILE_MULTIBLOCK_READ_COUNT 初期化パラメータの値を大きくすると、通常は順次スキャンでの I/O スループットが向上します。AIX では、このパラメータの範囲は 1 ～ 512 ですが、16 を超える値を使用しても、通常はそれ以上のパフォーマンス効果は得られません。

このパラメータの値は、DB_BLOCK_SIZE パラメータの値との積が LVM ストライプ・サイズよりも大きくなるように設定します。このように設定することによって、使用できるディスク数が増加します。

後書きの使用

後書き機能を使用すると、オペレーティング・システムで書き込み I/O をパーティションのサイズまでグループ化できるようになります。これによって、I/O 処理数が少なくなるため、パフォーマンスが向上します。ファイル・システムでは、各ファイルが複数の 16KB パーティションに分割されるため、書き込みパフォーマンスが向上し、メモリー内の使用済ページ数が制限され、ディスクの断片化が最小限に抑えられます。特定のパーティションのページは、プログラムによって次の 16KB パーティションの 1 バイト目書き込まれたときに、ディスクに書き込まれます。後書き用バッファのサイズを 8 個の 16KB パーティションに設定するには、次のコマンドを入力します。

```
# /usr/samples/kernel/vmtune -c 8
```

後書き機能を無効にするには、次のコマンドを入力します。

```
# /usr/samples/kernel/vmtune -c 0
```

順次先読みのチューニング

この項の内容は、ダイレクト I/O もコンカレント I/O も使用していないファイル・システムにのみ適用されます。

Virtual Memory Manager (VMM) は、順次ファイルのページの必要性を予測します。また、プロセスがファイルにアクセスするパターンを監視しています。プロセスがファイルの 2 つのページに連続してアクセスすると、VMM はプログラムがファイルへの順次アクセスを続行すると予測し、ファイルに対する後続の順次読み込みをスケジュールします。この結果、プログラム処理がオーバーラップし、プログラムはすぐにデータを利用できます。次の 2 つの VMM しい値がカーネル・パラメータとして実装されており、先読みするページ数の決定に使用されます。

- minpgahead
VMM が最初に順次アクセス・パターンを検出したときに先読みするページ数。
- maxpgahead
順次ファイルから VMM が先読みする最大ページ数。

minpgahead および maxpgahead パラメータは、アプリケーションに適した値に設定してください。デフォルト値は、それぞれ 2 および 8 です。これらの値を変更するには、`/usr/samples/kernel/vmtune` コマンドを使用します。ストライプ化論理ボリュームの順次パフォーマンスを重視する場合は、maxpgahead パラメータの値を高めに設定します。minpgahead パラメータを 32 ページに、maxpgahead パラメータを 64 ページに設定するには、次のコマンドを root ユーザーで入力します。

```
# /usr/samples/kernel/vmtune -r 32 -R 64
```

minpgahead および maxpgahead パラメータは、2 の累乗に設定します。たとえば、2、4、8、...512、1024... などのように設定します。

ディスク I/O の歩調合せのチューニング

ディスク I/O の歩調合せとは、システム管理者がファイルに対して保留される I/O 要求の数を制限するための AIX のメカニズムです。このメカニズムによって、ディスク I/O が頻繁に発生するプロセスのために CPU が飽和状態に陥ることを回避できます。この結果、対話型プロセスや CPU 使用量の多いプロセスの応答時間に遅延が発生しません。

ディスク I/O の歩調合せを行うには、最高水位標および最低水位標という 2 つのシステム・パラメータを調整します。保留中の I/O 要求が最高水位標に達しているファイルに対してプロセスが書き込みを実行すると、プロセスはスリープ状態になります。未処理の I/O 要求の数が最低水位標以下になると、プロセスはスリープ状態から解放されます。

最高水位標および最低水位標を変更するには、`smit` コマンドを使用します。試行錯誤を重ねて適切な水位標を決定します。水位標はパフォーマンスに影響を与えるため、設定するときには注意が必要です。ディスク I/O が 4KB を超える場合は、最高水位標および最低水位標をチューニングしてもほとんど効果はありません。

ディスク I/O の飽和状態を判断するには、`iostat` の結果、特に `iowait` および `tm_act` の割合を分析します。特定のディスクの `iowait` の割合と `tm_act` の割合が大きい場合、ディスクが飽和状態であることを表します。`iowait` の割合のみが大きい場合は、必ずしも I/O ボトルネックを表すものではないことに注意してください。

リモート I/O 操作の最小化

SP アーキテクチャで動作する Oracle Real Application Clusters は、様々なノードのすべてのインスタンスからアクセスできる共通記憶域として、VSD または HSD を使用します。I/O 要求が VSD に対して発行されたときに、VSD がノードのローカル論理ボリューム上にある場合は、ローカル I/O が実行されます。ローカルでない VSD に対して発行された I/O 通信量は、ネットワーク通信レイヤーを介して処理されます。

パフォーマンスを向上させるには、リモート I/O をできるかぎり最小限に抑えることが重要です。各インスタンスの REDO ログは、ローカル論理ボリュームにある VSD に配置する必要があります。更新および挿入が頻繁に発生する場合は、ローカル論理ボリュームに配置された VSD 上に各インスタンスの UNDO セグメントを割り当てる必要があります。

各セッションでは、一時表領域を 1 ユーザーに 1 つ割り当てることができます。各一時表領域には、各ノードのローカル・データファイルを 1 つ以上配置する必要があります。

リモート I/O が最小限に抑えられるように、アプリケーションおよびデータベースの設計（アプリケーションとデータベースのパーティション化など）には注意が必要です。

Oracle Database でのミラー復元

RAW 論理ボリューム (LV) に割り当てられた Oracle データファイルに対してミラー書き込み整合性 (MWC) を無効にすると、Oracle Database のクラッシュ・リカバリ・プロセスによるシステム・クラッシュ後のリカバリで、ミラーの復元が行われます。このミラー復元プロセスを実行することによって、データベースの不整合や破損を防止できます。

クラッシュ・リカバリ時に、論理ボリューム上のデータファイルに複数のコピーが割り当てられている場合、ミラー復元プロセスでは、それらのすべてのコピーのデータ・ブロックに対してチェックサムを実行します。その後、次のいずれかの処理を実行します。

- コピー内のデータ・ブロックのチェックサムが有効である場合、ミラー復元プロセスではそのコピーを使用して、チェックサムが無効なコピーを更新します。
- すべてのコピーについてブロックのチェックサムが無効である場合は、REDO ログ・ファイルの情報をを使用してブロックを再構築します。次に、そのデータファイルを論理ボリュームに書き込み、すべてのコピーを更新します。

AIX の場合、ミラー復元プロセスは、RAW 論理ボリュームに割り当てられたデータファイルのうち、MWC が無効になっているデータファイルに対してのみ有効です。ミラー化論理ボリューム上のデータファイルのうち、MWC が有効になっているデータファイルには、MWC によってすべてのコピーの同期が保証されているため、ミラー復元は必要ありません。

以前のリリースの Oracle Database のアップグレード中にシステムがクラッシュし、論理ボリューム上のデータファイルの MWC が無効になっていた場合は、`syncvg` コマンドを入力してミラー化 LV を同期化してから、Oracle Database を起動してください。ミラー化 LV を同期せずにデータベースを起動すると、LV コピーからデータが正しく読み込めないことがあります。

注意： ディスク・ドライブに障害が発生した場合、ミラー復元は行われません。その場合は、`syncvg` コマンドを入力してから、LV を再度アクティブにする必要があります。

注意： ミラー復元は、データファイルに対してのみサポートされています。このため、REDO ログ・ファイルの MWC は無効にしないでください。

RAW デバイスのバックアップ

RAW デバイスをバックアップするには RMAN の使用をお勧めします。dd コマンドを使用して RAW デバイスのバックアップを実行する場合は、次の注意に従います。

RAW デバイスの最初の Oracle ブロックのオフセットは、デバイス・タイプによって 0 (ゼロ)、4K または 128K になります。offset コマンドを使用して、適切なオフセットを判断できます。

論理ボリュームを作成する場合、オフセット 0 (ゼロ) を使用することをお勧めします。これは、`-t 0` オプションを使用した場合に可能です。ただし、以前のバージョンの Oracle Database で作成された既存の RAW 論理ボリュームでは、通常は 0 (ゼロ) 以外のオフセットです。次の例では、最初の Oracle ブロックのオフセットが 4K である RAW デバイスのバックアップおよびリストア方法を示しています。

```
$ dd if=/dev/raw_device of=/dev/rmt0.1 bs=256k
```

RAW デバイスをテープからリストアするには、次のようなコマンドを入力します。

```
$ 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
```

CPU のスケジューリングおよびプロセスの優先順位

CPU も、プロセスの競合が発生する可能性のあるシステム・コンポーネントです。AIX カーネルによってほとんどの CPU 時間は効果的に割り当てられますが、プロセスの多くは CPU サイクルをめぐって競合します。複数の CPU (SMP) が搭載されている場合は、各 CPU で様々なレベルの競合が発生する可能性があります。

プロセスのランタイム・スライスの変更

AIX RR ディスパッチャのランタイム・スライスのデフォルト値は 10 ミリ秒です。タイム・スライスを変更するには、`schedtune` コマンドを使用します。ただし、このコマンドを使用する場合は注意が必要です。タイム・スライスを長くすると、アプリケーションの自発的な切替率の平均が低い場合に、コンテキスト切替率も低くなります。その結果、プロセスのコンテキスト切替率に消費される CPU サイクルが少なくなるため、システムのスループットが向上します。

ただし、ランタイム・スライスが長いと、応答時間が遅くなる可能性があります (特に単一プロセッサ・システムの場合)。デフォルトのランタイム・スライスは通常、ほとんどのアプリケーションに使用できます。実行キューが大きく、多くのアプリケーションと Oracle シャドウ・プロセスがかなり長時間にわたって実行できる場合は、次のコマンドを入力して、タイム・スライスを長くできます。

```
# /usr/samples/kernel/schedtune -t n
```

前のコマンドで、*n* の値に 0 (ゼロ) を設定すると、タイム・スライスは 10 ミリ秒 (ms) になり、1 を設定すると 20ms、2 を設定すると 30ms になります。

SMP システムでのプロセッサ・バインディングの使用

SMP システムでは、複数のプロセスを 1 台のプロセッサにバインドすると、パフォーマンスが大幅に向上する場合があります。プロセッサ・バインディングは、AIX 5L ですべての機能を利用できます。

ただし、AIX 5L リリース 5.2 以降は、AIX スケジューラの特異な改良により、プロセッサ・バインディングの必要性もなく、Oracle Database プロセスを最適にスケジュール設定できます。したがって、AIX 5L リリース 5.2 以降で実行する場合は、プロセッサへのプロセスのバインドは、特にお薦めしません。

Oracle Real Application Clusters に関する情報

次の各項では、Oracle Real Application Clusters に関する情報を提供します。

UDP のチューニング

Oracle Real Application Clusters では、AIX でのプロセス間通信にユーザー・データグラム・プロトコル (UDP) が使用されます。UDP カーネルの設定をチューニングすることによって、Oracle のパフォーマンスを向上させることができます。AIX でのカーネルの UDP バッファリングを変更するには、`udp_sendspace` および `udp_recvspace` パラメータを変更します。`udp_sendspace` の値は、Oracle Database の `DB_BLOCK_SIZE` 初期化パラメータの値よりも常に大きくする必要があります。このように設定しないと、1 つ以上の Oracle Real Application Clusters インスタンスの起動で障害が発生します。これらのパラメータをチューニングするときは、次のガイドラインに従ってください。

- `udp_sendspace` パラメータの値は、`DB_BLOCK_SIZE` と `DB_FILE_MULTIBLOCK_READ_COUNT` を掛け合せて 4KB をプラスした値に設定します。したがって、たとえば、ブロック・サイズが 16KB で、`DB_FILE_MULTIBLOCK_READ_COUNT` が 16 の場合は、`udp_sendspace` を 260KB (266240) に設定します。
- `udp_recvspace` パラメータの値は、`udp_sendspace` パラメータの値の 10 倍以上に設定します。
- `udp_recvspace` パラメータの値は、`sb_max` パラメータの値より小さい値にする必要があります。

udp_recvspace パラメータの設定が適切かどうかを監視するには、次のコマンドを入力します。

```
$ netstat -p udp | grep "socket buffer overflows"
```

オーバーフローの数が 0（ゼロ）でない場合は、udp_recvspace パラメータの値を大きくします。次のコマンドを使用してエラー・カウンタをリセットしてから、このパラメータ設定の妥当性を監視します。

```
$ netstat -Zs -p udp
```

関連項目： これらのパラメータの設定方法は、『Oracle Real Application Clusters インストレーションおよび構成』を参照してください。AIX チューニング・パラメータの詳細は、『AIX 5L Performance Management Guide』を参照してください。

透過的アプリケーション・フェイルオーバーのためのネットワーク・チューニング

透過的アプリケーション・フェイルオーバーの時間が 10 分を超える場合は、rto_length、rto_low および rto_high の各ネットワーク・パラメータをチューニングし、フェイルオーバー時間を短縮してください。

透過的アプリケーション・フェイルオーバーの時間は、クライアントがクラッシュしたノードに接続し、障害が発生しているインスタンスから応答を受け取らない場合の TCP タイムアウトと再送信の問題に起因して長くなります。結果的に、クライアントは指数バックオフ・アルゴリズムを使用して、同じパケットの再送信を繰り返し続行します（詳細は TCP/IP のドキュメントを参照してください）。

AIX では、デフォルトのタイムアウト値は、約 9 分に設定されています。このパラメータは、no コマンドでロード時間属性 rto_length、rto_low および rto_high を使用してチューニングできます。これらのパラメータを使用すると、クライアントが同じパケットの再送信を試みる頻度と回数を制御できます。rto_low パラメータ（デフォルトは 1 秒）および rto_high パラメータ（デフォルトは 64 秒）は、パケットを送信する頻度を制御します。rto_length パラメータ（デフォルトは 13）は、パケットの送信回数を制御します。

たとえば、AIX のデフォルト値を使用すると、指数バックオフ・アルゴリズムによってタイムアウト値は約 9.3 分に設定されます。ただし、rto_length を 7 に設定すると、同じアルゴリズムによってタイムアウト値は 2.5 分に減少します。

注意： これらのパラメータを設定する前に、ネットワークの通信状況をチェックしてください。netstat コマンドを使用して、ネットワークの通信状況をチェックできます。ネットワークの通信状況がよい場合は、長めのタイムアウト値が必要です。

Oracle Real Application Clusters および HACMP または PSSP

Oracle Database 10g の場合、Real Application Clusters (RAC) では、AIX 5L RSCT Peer Domains (RPD) により提供されたグループ・サービスが使用されます。RAC は、HACMP または PSSP により提供される特定のサービスには依存しなくなりました。特に、情報リポジトリで PGSD_SUBSYS 変数を設定する必要はありません。

RAC は引き続き HACMP および PSSP と互換性があります。共有論理 RAW ボリュームが GPFS ファイル・システムのかわりに使用されている場合、通常は HACMP が存在します。SP Switch または SP Switch 2 がインターコネクトとして使用されている場合、PSSP が存在します。

IP ベースのインターコネクト（ギガビット・イーサネット、IEEE 802.3ad、EtherChannel、IP over SP Switch など）を使用している場合、サーバー・パラメータ・ファイルの CLUSTER_INTERCONNECTS パラメータの指定どおり、RAC によって使用するインタフェース名が決定されます。

Oracle Real Application Clusters とフォルト・トレラント IPC

Real Application Clusters 10g で使用されるインターコネクト（IPC）が Internet Protocol（IP）に基づいている場合、RAC では、IEEE 802.3ad Link Aggregation または EtherChannel テクノロジ（あるいはその両方）により、AIX 5L で作成されるフォルト・トレランスおよび Link Aggregation が利用されます。これは、以前のバージョンの Real Application Clusters で使用されていたフォルト・トレラント IPC 機能（FT-IPC）に代わるものです。

802.3ad を使用した Link Aggregation では、同じレベルのフォルト・トレランスが提供され、帯域幅集合体のサポートが追加されます。また、Real Application Clusters の構成が簡略化されます。

RAC は、サーバー・パラメータ・ファイルで CLUSTER_INTERCONNECTS パラメータを検索することで、使用する IP インタフェースを決定します。通常、このパラメータには、IEEE 802.3ad Link Aggregation または EtherChannel により作成された IP インタフェースの名前のみが含まれます。詳細は、AIX システムの管理ガイドのコミュニケーションおよびネットワークでのイーサチャンネルおよび IEEE 802.3ad Link Aggregation に関する説明を参照してください。

AIXTHREAD_SCOPE 環境変数の設定

AIX のスレッドは、プロセス全体の競合範囲（M:N）またはシステム全体の競合範囲（1:1）により動作できます。AIXTHREAD_SCOPE 環境変数は、使用する競合範囲を制御します。

AIXTHREAD_SCOPE 環境変数のデフォルト値は、P（プロセス全体の競合範囲）です。プロセス全体の競合範囲を使用する場合、Oracle スレッドはカーネル・スレッドのプールにマップされます。Oracle がイベントで待機中に Oracle スレッドがスワップ・アウトされると、スレッド ID が異なる別のカーネル・スレッドに戻る場合があります。Oracle では待機中のプロセスのポストにスレッド ID を使用するため、スレッド ID が同じままであることが重要です。システム全体の競合範囲を使用する場合、Oracle スレッドはカーネル・スレッドに 1 対 1 で静的にマップされます。このため、システム全体の競合を使用することをお勧めします。システム全体の競合の使用は、特に Oracle Real Application Clusters（RAC）インスタンスには重要です。

さらに、AIX 5L リリース 5.2 以上では、システム全体の競合範囲を設定した場合、各 Oracle プロセスに割り当てられるメモリー量がかなり少なくなります。

Oracle データベース・インスタンスまたは Oracle Net Listener プロセスにおいて、ORACLE_HOME または ORACLE_SID 環境変数の設定に使用する環境スクリプトで、次のように AIXTHREAD_SCOPE 環境変数の値を S に設定することをお勧めします。

- Bourne、Bash または Korn シェルの場合


```
~/.profile または /usr/local/bin/oraenv スクリプトに次の行を追加します。

AIXTHREAD_SCOPE=S; export AIXTHREAD_SCOPE
```
- C シェルの場合


```
~/.login または /usr/local/bin/coraenv スクリプトに次の行を追加します。

setenv AIXTHREAD_SCOPE S
```

これにより、すべての Oracle プロセスの実行に対してシステム全体のスレッド範囲が有効になります。

HP-UX システムでの Oracle Database の管理

この付録では、HP-UX システムで Oracle Database を管理する方法について説明します。次の項目について説明します。

- Oracle インスタンス用の HP-UX 共有メモリー・セグメント
- HP-UX SCHED_NOAGE スケジューリング・ポリシー
- 軽量タイマーの実装
- 非同期 I/O
- 大規模メモリーの割当てと Oracle Database のチューニング
- HP-UX での Oracle Real Application Clusters
- CPU_COUNT 初期化パラメータおよび HP-UX 動的プロセッサ再構成

Oracle インスタンス用の HP-UX 共有メモリー・セグメント

Oracle Database は、インスタンスの起動時に、Oracle 共有グローバル領域（SGA）の作成用に割り当てられた共有メモリーを HP-UX `shmmax` カーネル・パラメータの値で除算して、メモリー・セグメントを作成します。たとえば、1 つの Oracle インスタンスに割り当てられた共有メモリーが 64GB で、`shmmax` パラメータの値が 1GB の場合、Oracle Database はそのインスタンスに対して 64 個の共有メモリー・セグメントを作成します。

1 つの Oracle インスタンスに対して複数の共有メモリー・セグメントが作成されると、パフォーマンスが低下する可能性があります。これは、Oracle Database でインスタンスが作成されるときに、各共有メモリー・セグメントが一意的なプロテクション・キーを受け取るためです。使用できるプロテクション・キーの数は、次のように、システム・アーキテクチャによって異なります。

アーキテクチャ	プロテクション・キーの数
PA-RISC	6
Itanium	14

Oracle インスタンスが作成する共有メモリー・セグメントがプロテクション・キーの数より多い場合、HP-UX オペレーティング・システムは、プロテクション・キー・フォルトを表示します。

`shmmax` パラメータの値は、システムで使用できる物理メモリーの量に設定することをお勧めします。この設定によって、1 つの Oracle インスタンスの共有メモリー全体が 1 つの共有メモリー・セグメントに割り当てられ、インスタンスに必要なプロテクション・キーが 1 つで済みます。

システムのアクティブなメモリー・セグメントのリストを表示するには、次のコマンドを入力します。

```
$ ipcs -m
```

Oracle Database がそのインスタンスに対してプロテクション・キーの数よりも多いセグメントを作成する場合は、`shmmax` カーネル・パラメータの値を大きくします。

関連項目： 推奨されるカーネル・パラメータの最小値については、『Oracle Database インストレーション・ガイド for UNIX Systems』を参照してください。

HP-UX SCHED_NOAGE スケジューリング・ポリシー

HP-UX システムのほとんどのプロセスは、タイム・シェアリング・スケジューリング・ポリシーを使用します。タイム・シェアリングが適用されると、重要な処理（ラッチの保持など）が実行されるときに Oracle プロセスがスケジュールから除外されるため、パフォーマンスが低下することがあります。HP-UX の `SCHED_NOAGE` は、特にこの問題に対処した、修正済のスケジューリング・ポリシーです。通常のタイム・シェアリング・ポリシーとは異なり、`SCHED_NOAGE` によってスケジュールが設定されたプロセスは、優先順位が上下したり、優先使用されることがありません。

この機能は、オンライン・トランザクション処理（OLTP）環境に適しています。これは、OLTP 環境では、重要なリソースを対象に競合が発生することがあるためです。Oracle Database の OLTP 環境で `SCHED_NOAGE` ポリシーを使用した場合、パフォーマンスが 10 パーセント以上も向上する可能性があります。

`SCHED_NOAGE` ポリシーを意思決定支援（DSS）環境で使用しても、パフォーマンス上の効果はほとんどありません。これは、リソースの競合がほとんど発生しないためです。アプリケーションとサーバーの環境はそれぞれ異なるため、使用している環境に `SCHED_NOAGE` ポリシーが適切であるかどうかをテストおよび検証する必要があります。`SCHED_NOAGE` を使用する場合、Oracle プロセスに最も高い優先順位を割り当てることには注意が必要です。`SCHED_NOAGE` の最も高い優先順位を Oracle プロセスに割り当てると、システムの CPU リソースを使い果たし、他のユーザー・プロセスが停止する可能性があります。

Oracle Database での SCHED_NOAGE の有効化

Oracle Database で SCHED_NOAGE スケジューリング・ポリシーを使用するには、OSDBA グループ（通常は dba グループ）が RTSCHED および RTPRIO 権限を取得し、スケジューリング・ポリシーを変更したり、Oracle プロセスの優先順位レベルを設定する必要があります。dba グループにこれらの権限を付与するには、次の手順を実行します。

1. root ユーザーでログインします。
2. テキスト・エディタを使用して /etc/privgroup ファイルを開くか、必要な場合は作成します。
3. OSDBA グループの名前で始まる次の行を追加または編集し、システムがリブートするたびにこのグループに付与する RTPRIO 権限と RTSCHED 権限を指定します。

```
dba RTPRIO RTSCHED
```

4. ファイルを保存してテキスト・エディタを終了します。
5. 次のコマンドを入力して、OSDBA グループに権限を付与します。

```
# /usr/sbin/setprivgrp -f /etc/privgroup
```

6. 次のコマンドを入力して、権限が正しく設定されていることを確認します。

```
# /usr/sbin/getprivgrp dba
```

HP-UX SCHED_NOAGE 初期化パラメータを各インスタンスのパラメータ・ファイルに追加し、このパラメータに対してプロセスの優先順位レベル（整数値）を設定します。サポートされる値の範囲は 178 ～ 255 です。値が小さくなるほど優先順位が高くなります。パラメータの設定が範囲外にある場合、Oracle Database は自動的にそのパラメータを許容値に設定し、新しい値が設定された SCHED_NOAGE ポリシーを使用して処理を続行します。また、新しい設定に関するメッセージを alert_sid.log ファイルに生成します。ユーザーまたは自動再調整によって、Oracle プロセスに対して最高レベルの優先順位が割り当てられた場合、Oracle Database は、システムの CPU リソースが使い果たされる可能性があることを警告するメッセージを alert_sid.log ファイルに生成します。このパラメータには、Oracle プロセスに必要な優先順位レベルを割り当てておくことをお勧めします。

関連項目： 優先順位ポリシーおよび優先順位の範囲の詳細は、HP-UX システムのドキュメント、および rtsched(1) と rtsched(2) の各 man ページを参照してください。

軽量タイマーの実装

Oracle Database 10g では、動的初期化パラメータ STATISTICS_LEVEL が TYPICAL（デフォルト）または ALL に設定されている場合、いつでもランタイム統計を収集できます。このパラメータ設定は、TIMED_STATISTICS 初期化パラメータを暗黙的に true に設定します。HP-UX システム上の Oracle Database では、gethrtime() システム・ライブラリ・コールを使用して、統計の収集中に経過時間を計算します。この軽量システム・ライブラリ・コールを使用することによって、パフォーマンスに影響を与えることなく、Oracle インスタンスを実行しながら、いつでもランタイム統計を収集できます。

TIMED_STATISTICS 初期化パラメータが明示的に true に設定されているときに、gethrtime() システム・ライブラリ・コールを使用すると、使用しない場合に比べて Oracle のパフォーマンスを最大 10 パーセント改善できます。また、ランタイム統計の収集に gethrtime() システム・ライブラリ・コールを使用しても、Oracle Database の OLTP 環境のパフォーマンスが低下することはありません。

非同期 I/O

非同期 I/O 擬似ドライバを HP-UX システム上で使用すると、Oracle Database が RAW ディスク・パーティションへの I/O を非同期方式で実行できるようになり、I/O のオーバーヘッドが減少し、スループットが向上します。非同期 I/O 擬似ドライバは、HP-UX のサーバーとワークステーションの両方で使用できます。

MLOCK 権限

Oracle Database で非同期 I/O 操作を実行するには、OSDBA グループ (dba) に MLOCK 権限を付与する必要があります。dba グループに MLOCK 権限を付与するには、次の手順を実行します。

1. root ユーザーでログインします。
2. テキスト・エディタを使用して /etc/privgroup ファイルを開くか、必要な場合は作成します。
3. OSDBA グループの名前で始まる次の行を追加または編集し、MLOCK 権限を指定します。

注意： このファイルでは、特定のグループに対する権限指定に 1 行のみを使用する必要があります。このファイルに dba グループに関する行がすでに含まれている場合は、その行に MLOCK 権限を追加してください。

```
dba RTPRIO RTSCHED MLOCK
```

4. ファイルを保存してテキスト・エディタを終了します。
5. 次のコマンドを入力して、OSDBA グループに権限を付与します。

```
# /usr/sbin/setprivgrp -f /etc/privgroup
```
6. 次のコマンドを入力して、権限が正しく設定されていることを確認します。

```
# /usr/sbin/getprivgrp dba
```

非同期 I/O の実装

HP-UX システムで非同期 I/O を使用する場合は、データベース・ファイルに対して次の記憶域オプションを使用する必要があります。

- RAW デバイス (パーティションまたは論理ボリューム)
- RAW パーティションを使用する Automatic Storage Management (ASM) ディスク・グループ

関連項目： HP-UX システムで ASM および RAW 論理ボリュームを構成する方法は、『Oracle Database インストレーション・ガイド for UNIX Systems』を参照してください。

いずれかの記憶域オプションを使用して非同期 I/O を実装する前に、System Administrator Management (SAM) ユーティリティを使用して、非同期ディスク・ドライバを HP-UX カーネルに設定する必要があります。

SAM ユーティリティを使用して非同期ディスク・ドライバを追加し、カーネルを設定するには、次の手順を実行します。

1. root ユーザーで次のコマンドを入力します。

```
# sam
```
2. 「Kernel Configuration」領域を選択します。
3. 「Drivers」領域を選択します。

4. 非同期ディスク・ドライバ (asyncdsk) を選択します。
5. 「Actions」 → 「Add Driver to Kernel」を選択します。
6. 「List」 → 「Configurable Parameters」を選択します。
7. MAX_ASYNC_PORTS パラメータを選択します。
8. 「Action」 → 「Modify Configurable Parameter」を選択します。
9. 次のガイドラインを使用して、パラメータに新しい値を指定し、「OK」を選択します。

MAX_ASYNC_PORTS パラメータは、設定可能な HP-UX カーネル・パラメータの 1 つで、`/dev/async` ファイルを同時にオープンできる最大プロセス数を制御します。

最大数のプロセスが `/dev/async` ファイルをオープンした後で、別のプロセスがそのファイルをオープンしようとする、エラーが表示されます。多数のシャドウ・プロセスやパラレル問合せスレーブが非同期 I/O を実行しているシステムで、このエラーが発生すると、システムのパフォーマンスが低下することがあります。このエラーは記録されません。このエラーを回避するには、`/dev/async` ファイルにアクセスする最大プロセス数を予測し、MAX_ASYNC_PORTS パラメータにその値を設定します。

10. 「Actions」 → 「Process a New Kernel」を選択します。
11. 次のオプションのいずれかを選択し、「OK」を選択します。
 - 「Move Kernel Into Place and Shutdown System/Reboot Now」
 - 「Do Not Move Kernel Into Place: Do Not Shutdown/Reboot Now」

2 番目のオプションを選択した場合は、新しいカーネル `vmunix_test` とその作成に使用される `system.SAM` 構成ファイルが、`/stand/build` ディレクトリに作成されます。

新しいカーネルを使用するには、次の手順を実行します。

1. 次のコマンドを入力して、新しいカーネルを所定の場所に移動します。

```
# /usr/sbin/kmupdate
```

2. 次のコマンドを入力して、システムをリブートします。

```
# /sbin/shutdown -r now
```

HP-UX 非同期デバイス・ドライバを使用して非同期 I/O 操作を可能にするには、次の手順を実行します。

1. root ユーザーでログインします。
2. 次のコマンドを入力して、新しいデバイス・ファイルを作成します。

```
# /sbin/mknod /dev/async c 101 0x0
```

3. 次のコマンドを入力して、メジャー番号が 101 の `/dev/async` デバイス・ファイルが存在することを確認します。

```
# ls -l /dev/async
```

このコマンドの出力は、次のようになります。

```
crw----- 1 oracle dba 101 0x000000 Oct 28 10:32 /dev/async
```

4. 必要に応じて、このデバイス・ファイルに対し、Oracle ソフトウェア所有者および OSDBA グループとの整合性のある UNIX 所有者およびアクセス権を設定します。

Oracle ソフトウェア所有者が `oracle` で、OSDBA グループが `dba` の場合は、次のコマンドを入力します。

```
# /usr/bin/chown oracle:dba /dev/async
# /usr/bin/chmod 660 /dev/async
```

非同期 I/O の検証

非同期 I/O を検証するには、最初に、HP-UX 非同期ドライバが Oracle Database に対して設定されていることを検証します。次に、Oracle Database が HP-UX デバイス・ドライバを介して非同期 I/O を実行していることを検証します。

HP-UX 非同期ドライバが Oracle Database に対して設定されているかどうかの検証

HP-UX 非同期ドライバが Oracle Database に対して適切に設定されていることを検証するには、次の手順を実行します。

1. 少ない数のパラレル問合せスレーブ・プロセスを持つ Oracle Database を起動します。
2. 次のコマンドを入力して、GlancePlus/UX ユーティリティを起動します。
\$ gpm
3. メイン・ウィンドウで、「Reports」→「Process List」を選択します。
4. 「Process List」ウィンドウで、パラレル問合せスレーブ・プロセスを 1 つ選択し、「Reports」→「Process Open Files」を選択します。
パラレル問合せスレーブ・プロセスによって現在オープンされているファイルのリストが表示されます。
5. オープン・ファイルのリストから、/dev/async ファイルまたはモード 101 0x000000 を検索します。

/dev/async ファイルまたはモード 101 0x000000 のいずれかがリストに含まれている場合は、パラレル問合せスレーブ・プロセスによって /dev/async ファイルがオープンされています。つまり、HP-UX 非同期デバイス・ドライバは、Oracle プロセスが非同期 I/O を実行できるように正しく設定されています。/dev/async ファイルのファイル記述子番号をメモしておいてください。

Oracle Database が非同期 I/O を使用しているかどうかの検証

Oracle Database が HP-UX 非同期デバイス・ドライバを介して非同期 I/O を使用しているかどうかを検証するには、次の手順を実行します。

1. HP-UX システムの tusc ユーティリティを、前述の手順で GlancePlus ユーティリティを使用して選択した Oracle パラレル問合せスレーブに接続します。
2. 使用している環境で I/O バウンド問合せを実行します。
3. tusc 出力で read/write コールのパターンをチェックします。

たとえば、次のコマンドを入力します。pid は、非同期 I/O を実行する予定のパラレル問合せスレーブのプロセス ID です。

```
$ tusc -p pid > tusc.output
```

4. 問合せの実行後、[Ctrl] キーを押しながら [C] キーを押してプロセスから切断し、tusc.output ファイルを開きます。

次に、tusc.output ファイルのサンプルを示します。

```
( Attached to process 2052: "ora_p000_tpch" [ 64-bit ] )
.....
.....
[2052] read(9, "80\0\001\013 \b\0\0\0\0\0\0\0"..., 388) .. = 28
[2052] write(9, "\0\0\00e\0\0\0\080\0\001\013\0"..., 48) .. = 48
[2052] read(9, "80\0\001\013\0 18\0\0\0\0\0\0\0"..., 388) .. = 28
[2052] write(9, "\0\0\00e\0\0\0\080\0\001\01bd4\0"..., 48) .. = 48
```

DISK_ASYNC_IO 初期化パラメータが明示的に false に設定されていない場合（デフォルトの true になっている場合）、tusc.output ファイルには、同じファイル記述子（前述の例では 9）の非同期 read/write コールのパターンが連続して表示されます。

tusc.output ファイルのファイル記述子番号を GlancePlus の /dev/async ファイルで使用されている番号にマップします。この番号は、特定のパラレル問合せスレーブ・プロセスについて一致しています。これによって、HP-UX 非同期デバイス・ドライバを介した I/O が非同期であることが検証されます。同期 I/O の場合または DISK_ASYNC_IO 初期化パラメータが明示的に false に設定されている場合、前述の非同期 read/write パターンは表示されません。かわりに、lseek または pread/pwrite のコールが表示されます。また、1 つのファイル記述子のみでなく、多数の異なるファイル記述子（read/write の最初の引数）が表示されます。

SGA の非同期フラグ

HP-UX システム上の Oracle Database では、HP-UX 非同期ドライバが提供する非ブロック・ポーリング機能を使用して、I/O 操作の状態がチェックされます。非同期ドライバは、送信された I/O 操作の状態に基づいてフラグを更新します。このポーリングではこのフラグがチェックされます。HP-UX では、このフラグが共有メモリーに読み込まれている必要があります。

Oracle Database では、非同期フラグを各 Oracle プロセスの SGA 内に設定します。HP-UX システム上の Oracle Database は、真の非同期 I/O メカニズムを備えています。つまり、以前に発行された I/O 操作の一部が完了していなくても、I/O 要求を発行できます。このメカニズムにより、パフォーマンスが向上し、パラレル I/O プロセスのスケラビリティが保証されます。

Oracle8i リリース 8.1.7 より前の Oracle Database では、I/O 操作は、HP-UX 非同期ドライバを使用して共有メモリーからのみ実行できました。新しい HP-UX 非同期ドライバでは、Oracle Database 10g は共有メモリーとプロセス専用領域の両方から I/O 操作を実行できます。ただし、非同期ドライバを介した I/O 操作は、本質的には非同期ではありません。Oracle Database では、非同期ドライバに送信された I/O 操作の状態をチェックするために、ブロック待機が必要です。このため、データベース・ライター・プロセスなど一部の Oracle プロセスでは、本質的には同期 I/O が実行されています。

大規模メモリーの割当てと Oracle Database のチューニング

Oracle Database 10g 上で実行するアプリケーションは、以前のリリースの Oracle で実行するアプリケーションと比べて非常に大きいメモリーを使用する場合があります。これには、次の 2 つ理由があります。

- CURSOR_SPACE_FOR_TIME 初期化パラメータのデフォルト値が、false から true に変更されたこと。
- HP-UX システム上の Oracle Database 10g で、仮想メモリー・データ・ページのデフォルト設定値が D (4KB) から L (1GB) に変更されたこと。

永続的な専用 SQL 領域とメモリー

ユーザーが SQL 文を発行すると、Oracle Database は、次のメモリー割当てステップを自動的に実行します。

1. Oracle SGA の共有プールをチェックし、同一文に対する共有 SQL 領域がすでに存在しているかどうかを確認します。共有 SQL 領域が存在する場合は、その領域を使用して、文の以降の新しいインスタンスを実行します。共有 SQL 領域が存在しない場合は、共有プール内の新しい共有 SQL 領域を SQL 文に割り当てます。
2. ユーザー・セッションのために、専用 SQL 領域を割り当てます。

専用 SQL 領域には、処理対象 SQL 文のバインド情報やランタイム・メモリー構造などのデータが格納されます。また、解析対象文および情報を処理するその他の文も格納されます。

同じ SQL 文を実行するすべてのユーザーには、単一の共有 SQL 領域を使用するカーソルが割り当てられます。このようにして、多数の専用 SQL 領域を同じ共有 SQL 領域に関連付けることができます。ユーザー・セッションが専用サーバーを介して接続された場合、専用 SQL 領域

はサーバー・プロセスの PGA 内に設定されます。ただし、ユーザー・セッションが共有サーバーを介して接続された場合は、専用 SQL 領域の一部が SGA 内に保持されます。

`CURSOR_SPACE_FOR_TIME` 初期化パラメータは、SQL カーソルの割当てをライブラリ・キャッシュから解除して、新規 SQL 文のために領域を確保できるかどうかを指定します。このパラメータを `true` に設定すると、共有 SQL 領域の割当てが Oracle ライブラリ・キャッシュから解除されるのは、SQL 文に関連付けられたすべてのアプリケーション・カーソルがクローズしている場合のみになります。また、このパラメータが `true` の場合、オープンしているカーソルに関連付けられた専用 SQL 領域の割当て解除も防止されるため、ユーザーの専用 SQL 領域を永続的にすることができます。

Oracle Database で `CURSOR_SPACE_FOR_TIME` 初期化パラメータを `true` に設定した場合、以前のリリースの Oracle と比べて次のメリットがあります。

- アクティブな各カーソルの SQL 領域がメモリーに存在してエージ・アウトしないため、SQL の実行コールが高速になります。
- 共有 SQL 領域がライブラリ・キャッシュに存在することを Oracle Database が検証する必要がないため、アプリケーションのパフォーマンスが向上します。次の SQL 文の実行まで専用 SQL 領域が保持されるため、カーソルの割当てと初期化の時間が節約されます。

Oracle Database で `CURSOR_SPACE_FOR_TIME` 初期化パラメータを `true` に設定した場合、以前のリリースの Oracle と比べて次の短所があります。

- 永続的な専用 SQL 領域用のメモリー割当てが増加するため、ユーザー・プロセスのメモリー要件が増加します。
- カーソルのメモリーが大幅に増加するため、Oracle Database のシャドウ・プロセス用のメモリー割当てが大きくなります。

`CURSOR_SPACE_FOR_TIME` パラメータを `false` に設定すると、SQL 全体の実行速度が遅くなり、パフォーマンスが低下することがあります。このパラメータを `false` に設定すると、共有 SQL 領域の割当てがライブラリ・キャッシュから早期に解除される可能性があります。

デフォルトの大規模仮想メモリー・ページ・サイズ

デフォルトでは、Oracle Database は、プロセス専用メモリーの割当てに、HP-UX システムの使用可能な最大の仮想メモリー・ページ・サイズ設定を使用します。このサイズは、値 **L** (最大) で定義されます。HP-UX 11i では、現在 1GB です。この値は、Oracle 実行可能ファイルのリンク時に、`LARGE_PAGE_FLAGS` オプションの 1 つとして設定されます。

仮想メモリー・ページ・サイズが **L** に設定されていると、HP-UX は、使用可能なプロセス専用メモリーを、1GB 制限まで、または割り当てられたメモリー量の合計に達するまで、1MB、4MB、16MB などのサイズのページに割り当てます。Oracle PGA に十分なメモリーが割り当てられていて、大きなデータ・ページ・サイズ単位でのメモリー割当てが可能な場合は、オペレーティング・システムは、一度に最大のページ・サイズを割り当てます。たとえば、Oracle PGA に 48MB を割り当てている場合、システムでは、16MB を 3 ページ設定するか、またはより小さい倍数の単位サイズのページを組み合わせることができます。たとえば、1MB を 4 ページ、4MB を 3 ページおよび 16MB を 2 ページとするなどです。PGA に 64MB を割り当てた場合、データ・ページ単位サイズと使用可能なメモリー量とが一致するため、1 ページ (64MB) がオペレーティング・システムによって割り当てられます。

一般的には、大規模メモリー・ページによってアプリケーションのパフォーマンスは向上します。これは、オペレーティング・システムが処理する必要のある仮想メモリー変換時のエラー数が減り、より多くの CPU リソースをアプリケーションに解放できるためです。また、プロセス専用メモリーの割当てに必要なデータ・ページ数の合計も減ります。これによって、プロセス・レベルでの Translation Lookaside Buffer (TLB) ミスの可能性が減ります。

ただし、アプリケーションにメモリーの制約があり、非常に多くのプロセスを実行する傾向がある場合は、この大幅なページ・サイズの増加によって、プロセスに大規模メモリーの割当てが指示されるため、メモリー不足エラーが発生する可能性があります。メモリー不足エラーが発生する場合は、ページ・サイズの値を小さくして、**D** (デフォルト) サイズの 4KB と **L** (最大) サイズの 1GB の間に設定する必要があります。

最小ページ・サイズ設定値（4KB）を使用すると、最大ページ・サイズ設定値を使用した場合よりも CPU 使用率が 20 パーセント以上高くなります。最大設定値 L を使用すると、4MB の設定値を使用した場合よりもメモリー使用率が 50 パーセント高くなります。システムにメモリー制約がある場合は、使用できるメモリー・リソースの制約内で、ページ・サイズを特定のアプリケーションの要件と一致するように設定することをお勧めします。

たとえば、設定値 L では問題が発生するアプリケーションで、設定値 4MB の仮想メモリー・ページを使用すると、適切なパフォーマンスが得られる場合があります。

チューニングに関する推奨事項

永続的な専用 SQL 領域および大規模仮想メモリー・ページ・サイズへのメモリー割当ての増加に対応してチューニングを行うには、次の推奨事項があります。

- CURSOR_SPACE_FOR_TIME パラメータの値は、true にしておきます。ただし、この設定で、アプリケーションの実行時にライブラリ・キャッシュ・ミスが発生する場合があります。この場合、共有プールが、同時にオープンしているすべてのカーソル用の SQL 領域を保持できる最小限のサイズになっている可能性があります。
- 必要に応じて、Oracle Database の仮想メモリー・データ・ページ・サイズを減らします。次のコマンドを使用して、ページ・サイズ設定を変更します。

```
# /usr/bin/chatr +pd newsize $ORACLE_HOME/bin/oracle
```

この例の *newsize* は仮想メモリー・ページ・サイズの新しい値を表します。

chatr コマンドを次のように使用して、新しい設定を表示します。

```
# /usr/bin/chatr $ORACLE_HOME/bin/oracle
```

HP-UX での Oracle Real Application Clusters

次の各項では、HP-UX での Oracle Real Application Clusters に関する情報を提供します。

PA-RISC の場合のみ : Hyper Messaging Protocol パラメータのチューニング

HP-UX 11i (11.11) PA-RISC システムの Oracle Real Application Clusters には、クラスタ・インターコネクト・プロトコルとして Hyper Messaging Protocol (HMP) を使用できます。HMP の詳細は、HP のマニュアルを参照してください。

Oracle で HMP を使用可能にするには、次のように、HMP を使用して oracle バイナリを再リンクする必要があります。

1. すべてのノードで環境を設定します。
2. データベースをシャットダウンします。
3. すべてのノードで次のコマンドを入力します。

```
$ cd $ORACLE_HOME/rdbms/lib
$ make -f ins_rdbms.mk ipc_hms ioracle
```

HMP を使用する場合は、/opt/clic/lib/skgxp/skclic.conf ファイルにリストされている HMP パラメータを検討およびチューニングすることをお勧めします。

CPU_COUNT 初期化パラメータおよび HP-UX 動的プロセッサ再構成

HP-UX 11i では、プロセッサ・セット (Psets) の動的ランタイム再構成、および有効なユーザーによるプロセッサ・セット間の作業負荷の動的再割当てをサポートしています。

PA-RISC システムでは、HP-UX Virtual Partitions (vPars) を使用すると、ユーザーは、各自のシステムを複数の論理パーティションで構成し、各パーティションに独自のプロセッサ、メモリーおよび I/O リソースのセットを割り当て、HP-UX オペレーティング・システムの個別のインスタンスを実行できます。vPars に組み込まれている HP-UX Processor Sets を使用すると、仮想パーティションをリブートせずに、仮想パーティション間で動的プロセッサを移行できます。これによって、アプリケーション間でリソースのパーティション化を効率的に行うことができるため、HP-UX サーバー上で実行する各アプリケーションへのインタフェースおよび保証を必要とするリソースの割当てが最小限で済みます。

Oracle Database の CPU_COUNT 初期化パラメータは、Oracle Database で使用可能な CPU 数を指定します。HP-UX 11i システムの Oracle Database 10g では、オペレーティング・システムを定期的に問い合わせることによって、CPU ホスト構成の変更を動的に検出できます。システムの CPU 数が増えられた場合、Oracle は CPU_COUNT パラメータを適切な値に調整して、その内部リソースを再割当てします。これによって、新しい作業負荷は新しく追加されたプロセッサを利用できるため、高い CPU 使用率が原因でボトルネックが発生している場合は、DBA による変更を必要とせずに、データベースのパフォーマンスを改善できます。

初期化パラメータの一部の値は、システムの起動時に CPU_COUNT 値に基づいて計算されます。システムの起動後に CPU 数が増えられた場合、これらの初期化パラメータは動的に更新されず、新しい CPU 数は考慮されません。このため、新しい CPU 数が元の CPU 数と大きく異なる場合は、最適でないデータベース構成となる場合があります。システムで CPU 数が大幅に増加すると、データベースは追加された処理能力を利用できない可能性があります。

システムで増加した CPU 数が小さい場合（たとえば、2 から 4 に変更した場合）、必要な処理はありません。

システムで増加した CPU 数が大きい場合（たとえば、2 から 32 に変更した場合）は、次の手順に従ってください。

1. 次のどちらかの方法で、CPU_COUNT 初期化パラメータを新しい値に設定します。
 - データベースでサーバー・パラメータ・ファイル (spfiledbname.ora) を使用している場合は、次の SQL*Plus コマンドを SYS ユーザーで入力し、新しいパラメータ値を指定します。

```
SQL> ALTER SYSTEM SET CPU_COUNT=32 SCOPE=SPFILE
```
 - データベースで初期化パラメータ・ファイル (initSID.ora) を使用している場合は、このファイルを編集して新しいパラメータ値を指定します。
2. データベースを再起動します。

Linux システムでの Oracle Database の管理

この付録では、Linux システムで Oracle Database を管理する方法について説明します。次の項目について説明します。

- x86 の場合のみ : 拡張バッファ・キャッシュのサポート
- x86 の場合のみ : Red Hat Enterprise Linux AS 2.1 および SUSE Linux Enterprise Server 8 でのラージ・ページの有効化
- Red Hat Enterprise Linux AS 2.1 (Itanium) または SUSE Linux Enterprise Server 9 での hugetlbfs の使用
- Red Hat Enterprise Linux AS 3 での hugetlbfs の使用
- Linux x86 の場合のみ : SGA アドレス空間の増加
- 非同期 I/O サポート
- ダイレクト I/O サポート
- semtimedop のサポート
- x86 の場合のみ : 高速ネットワークのサポート

x86 の場合のみ：拡張バッファ・キャッシュのサポート

Oracle Database では、データベース・バッファ・キャッシュとして、4GB を超えるメモリを割り当てて使用できます。この項では、x86 システムの拡張バッファ・キャッシュ機能の制限事項と要件について説明します。

関連項目： 拡張バッファ・キャッシュ機能の詳細は、『Oracle Database 概要』を参照してください。

メモリー内ファイル・システム

拡張バッファ・キャッシュ機能を使用するには、データベース・バッファ・キャッシュに使用するメモリー量以上のサイズのメモリー内ファイル・システムを `/dev/shm` マウント・ポイントに作成します。たとえば、8GB のファイル・システムを `/dev/shm` マウント・ポイントに作成するには、次の手順を実行します。

1. root ユーザーで次のコマンドを入力します。

```
# mount -t tmpfs shmfs -o size=8g /dev/shm
```

2. `/etc/fstab` ファイルに次のようなエントリを追加し、システムのリブート時にメモリー内ファイル・システムが確実にマウントされるようにします。

```
shmfs /dev/shm tmpfs size=8g 0 0
```

拡張バッファ・キャッシュ機能を有効にして Oracle Database を起動すると、Oracle バッファ・キャッシュに対応するファイルが `/dev/shm` ディレクトリに作成されます。

注意： メモリー内ファイル・システムが `/dev/shm` マウント・ポイントにすでにマウント済の場合は、その大きさがデータベース・バッファ・キャッシュに使用するメモリー量以上であることを確認します。

USE_INDIRECT_DATA_BUFFERS 初期化パラメータ

拡張バッファ・キャッシュ機能を有効にするには、パラメータ・ファイルで `USE_INDIRECT_DATA_BUFFERS` 初期化パラメータを `true` に設定します。これによって、Oracle Database は大きいバッファ・キャッシュを指定できるようになります。

動的キャッシュ・パラメータ

拡張キャッシュ機能が有効になっている場合は、`DB_BLOCK_BUFFERS` パラメータを使用してデータベース・キャッシュ・サイズを指定する必要があります。

拡張バッファ・キャッシュ機能が有効になっている間は、次の動的キャッシュ・パラメータを使用しないでください。

- `DB_CACHE_SIZE`
- `DB_2K_CACHE_SIZE`
- `DB_4K_CACHE_SIZE`
- `DB_8K_CACHE_SIZE`
- `DB_16K_CACHE_SIZE`

制限事項

拡張バッファ・キャッシュ機能には、次の制限事項が適用されます。

- 作成または使用できる表領域はデフォルト・ブロック・サイズの表領域のみです。`DB_BLOCK_SIZE` パラメータで指定されたブロック・サイズのみを使用した表領域を作成できます。
- インスタンスの実行中はバッファ・キャッシュのサイズを変更できません。

関連項目： CREATE TABLESPACE コマンドで使用するデフォルト・ブロック・サイズの詳細は、『Oracle Database SQL リファレンス』を参照してください。

注意： デフォルトの VLM ウィンドウ・サイズは 512MB です。このメモリー・サイズは、プロセスのアドレス空間に割り当てられます。この値を増減させるには、VLM_WINDOW_SIZE 環境変数をバイト単位の新しいサイズに設定します。たとえば、VLM_WINDOW_SIZE を 256MB に設定するには、次のように入力します。

```
$ export VLM_WINDOW_SIZE=268435456
```

VLM_WINDOW_SIZE 環境変数に指定する値は、64KB の倍数である必要があります。

Red Hat Enterprise Linux 3 の場合のみ : VLM ウィンドウ・サイズ

VLM ウィンドウ・サイズに対応するには、プロセスごとのロック・メモリーのデフォルトの最大サイズを増やす必要があります。このサイズを増やすには、`/etc/security/limits.conf` ファイルに次の行を追加します。この場合、oracle はデータベースを管理するユーザーです。

oracle	soft	memlock	3145728
oracle	hard	memlock	3145728

ssh を使用してシステムにログインする場合、`/etc/ssh/ssh_config` ファイルに次の要素を追加して、ssh セッションの開始時にデフォルト値を使用できるようにします。

```
UsePrivilegeSeparation no
```

x86 の場合のみ : Red Hat Enterprise Linux AS 2.1 および SUSE Linux Enterprise Server 8 でのラージ・ページの有効化

ラージ・ページを有効化すると、カーネルにおける仮想メモリー・オーバーヘッドが減少し、スケーラビリティが向上します。ただし、ラージ・ページ・プールのメモリー・リソースはスワッピングできなくなるため、システムのメモリーが十分であることを確認する必要があります。

注意： bigpages パラメータの使用は、Red Hat Enterprise Linux AS 2.1 および SUSE Linux Enterprise Server 8 でのみサポートされます。Red Hat Enterprise Linux AS 3 または SUSE Linux Enterprise Server 9 では使用不可であるか、サポートされていません。

ラージ・ページ機能を使用するには、bigpages パラメータの値を決定する必要があります。この値を決定するには、インスタンスの SGA のサイズを MB に変換し、4MB 単位で端数を切り上げます。次に例を示します。

注意： SGA のサイズを決定するには、SQL*Plus で SHOW SGA コマンドを使用する方法と、ipcs コマンドを使用する方法があります。

SGA が 2.7GB と仮定します。

1. `bigpages` の値を次のように決定します。

$$\text{bigpages} = 2.7 \times 1024 = 2764.8$$
2. この値を 4 単位で端数を切り上げて 2768 にします。
3. 使用するブート・ローダーに応じて、次のいずれかの手順を実行します。

LILO の場合

- a. `/etc/lilo.conf` ファイルの適切なイメージ・セクションに、`bigpages` オプションを追加します。

```
append = "bigpages=2768MB"
```

- b. `/sbin/lilo` を実行します。
- c. システムを再起動します。

GRUB の場合

- a. `/etc/grub.conf` ファイルの `kernel` コマンドに、`bigpages` オプションを追加します。次に例を示します。

```
kernel /vmlinuz-2.4.9 root=/dev/hda5 bigpages=2768MB
```

- b. システムを再起動します。
4. ラージ・ページ機能を有効化するには、拡張バッファ・キャッシュ機能を使用しているかどうかに応じて、`/etc/sysctl.conf` ファイルの `shm-use-bigpages` カーネル・パラメータの値を 1 または 2 に設定し、パラメータ・ファイル内で `USE_INDIRECT_DATA_BUFFERS` 初期化パラメータを `true` に設定します。

- 拡張バッファ・キャッシュ (VLM) を使用していない場合は、値を 1 に設定します。

```
kernel.shm-use-bigpages = 1
```

- 拡張バッファ・キャッシュ (VLM) を使用している場合は値を 2 に設定し、`/dev/shm` に割り当てられたメモリを含めるように `bigpages` メモリー・プールを設定します。

```
kernel.shm-use-bigpages = 2
```

5. 次のコマンドを入力してパラメータ値を設定します。

```
# sysctl -p /etc/sysctl.conf
```

6. Oracle Database を起動します。

Red Hat Enterprise Linux AS 2.1 (Itanium) または SUSE Linux Enterprise Server 9 での hugetlbfs の使用

Red Hat Enterprise Linux AS 2.1 (Itanium) または SUSE Linux Enterprise Server 9 において、Oracle Database でラージ・ページ (Huge ページとも呼ばれる) を使用可能にするには、`vm.nr_hugepages` カーネル・パラメータの値を設定し、予約するラージ・ページ数を指定します。データベース・インスタンスの SGA 全体を保持するだけの十分なラージ・ページ数を指定する必要があります。必要なパラメータ値を判断するには、インスタンスの SGA サイズをラージ・ページのサイズで除算してから、結果の端数を切り上げて最も近い整数にします。

デフォルトのラージ・ページ・サイズを判断するには、次のコマンドを入力します。

```
# grep Hugesize /proc/meminfo
```

たとえば、`/proc/meminfo` にラージ・ページのサイズが 2MB とリストされ、インスタンスの総 SGA サイズが 1.6GB の場合は、`vm.nr_hugepages` カーネル・パラメータの値を 820 ($1.6\text{GB} / 2\text{MB} = 819.2$) に設定します。

Red Hat Enterprise Linux AS 3 での hugetlbfs の使用

Red Hat Enterprise Linux AS 3 でラージ・ページを使用するには、次の手順を実行します。

1. ラージ・ページ・プールに必要なメモリーを決定します。

この値を決定するには、インスタンスの SGA のサイズを MB に変換し、4MB 単位で端数を切り上げます。たとえば、SGA が 2.7GB の場合、適切な値は 2768MB です。

2. 使用するブート・ローダーに応じて、次のいずれかの手順を実行します。

LILO の場合

- a. /etc/lilo.conf ファイルの適切なイメージ・セクションに、hugepages オプションを追加して、ページ数を指定します。

```
append = "hugepages=1024"
```

- b. /sbin/lilo を実行します。

- c. システムを再起動します。

GRUB の場合

- a. /etc/grub.conf ファイルの kernel コマンドに、hugepages オプションを追加して、ページ数を指定します。

```
kernel /vmlinuz-2.4.9 root=/dev/hda5 hugepages=1024
```

- b. システムを再起動します。

3. /etc/sysctl.conf ファイルに次のエントリを追加または編集して、ラージ・ページ・プールのサイズを MB で指定します。

```
vm.hugetlb_pool = 2768
```

4. 次のコマンドを入力してカーネル・パラメータの値を設定します。

```
# sysctl -p /etc/sysctl.conf
```

5. 次のように入力し、このメモリー量がラージ・ページ・プールに正常に移動したことを確認します。

```
# cat /proc/meminfo
```

最後に表示される行は、ラージ・ページ・プールに移動したメモリー・ページ数を示します。

6. データベースを起動します。

Linux x86 の場合のみ : SGA アドレス空間の増加

Linux のディストリビューションに応じて、次のいずれかの項の説明を参照して SGA アドレス空間を増やします。

- [Red Hat Enterprise Linux AS 2.1](#) および [SUSE Linux Enterprise Server 8](#) または [9](#)
- [Red Hat Enterprise Linux AS 3](#)

Red Hat Enterprise Linux AS 2.1 および SUSE Linux Enterprise Server 8 または 9

Red Hat Enterprise Linux AS 2.1 および SUSE Linux Enterprise Server 8 または 9 で SGA アドレス空間を増やすには、次の手順を実行します。

1. 必要に応じて、oracle ユーザーでログインします。
2. \$ORACLE_HOME/rdbms/lib ディレクトリで、次のコマンドを入力します。

```
$ genksms -s 0x15000000 > ksms.s
$ make -f ins_rdbms.mk ksms.o
$ make -f ins_rdbms.mk ioracle
```

注意： 前述の手順を完了しても Oracle Database が起動しない場合、またはランタイム・メモリー・エラーが存在する場合は、最初のコマンドで指定した 16 進の数値を大きくしてください。たとえば、0x15000000 の値で Oracle Database が起動しない場合は、0x20000000 の値を指定します。この値を小さくすると SGA アドレス空間が大きくなりますが、PGA アドレス空間は小さくなる場合があります。

3. 次のコマンドを入力し、oracle ユーザーのシェル・プロセスのプロセス ID を確認します。

```
$ echo $$
```

戻される数値はプロセス ID です。

4. ユーザーを root に切り替えます

```
$ su - root
```

5. 次のコマンドを入力し、oracle ユーザーのシェル・プロセスの mapped_base 設定を変更します。pid は、手順 3 で識別したプロセス ID です。

```
# echo 268435456 > /proc/pid/mapped_base
```

6. exit と入力して oracle ユーザーのシェル・プロセスに戻り、Oracle リスナーと Oracle Database を起動します。

注意： Oracle プロセスはすべて、この変更した mapped_base 値を取得する必要があります。mapped_base が変更されているシェルからリスナーを起動すると、クライアント接続を正しく接続できます。

Red Hat Enterprise Linux AS 3

Red Hat Enterprise Linux AS 3 で SGA アドレス空間を増やすには、次の手順を実行します。

1. 必要に応じて、oracle ユーザーでログインします。
2. \$ORACLE_HOME/rdbms/lib ディレクトリで、次のコマンドを入力します。

```
$ genksms -s 0x15000000 > ksms.s
$ make -f ins_rdbms.mk ksms.o
$ make -f ins_rdbms.mk ioracle
```

3. Oracle Database を起動します。

非同期 I/O サポート

注意： Linux では、デフォルトにより、Automatic Storage Manager (ASM) で非同期 I/O が使用されます。

NFS ファイル・システムに格納されたデータベース・ファイルでは、非同期 I/O はサポートされません。

Oracle Database では、カーネルの非同期 I/O がサポートされます。この機能は、デフォルトでは無効になっています。Oracle Database の稼働しているシステムがカーネルの非同期 I/O をサポートしていて、そのシステムでの非同期 I/O の使用をオラクル社が認定している場合は、次の手順に従って非同期 I/O サポートを有効にします。

1. oracle ユーザーとして、ディレクトリを \$ORACLE_HOME/rdbms/lib ディレクトリに変更します。
2. 次のコマンドを入力します。

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

注意： 「/usr/bin/ld: cannot find -laio」のエラーが表示された場合は、そのシステムではカーネルの非同期 I/O がサポートされていません。次のコマンドを入力して、Oracle インスタンスを使用可能な状態にリストアしてください。

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

デフォルトでは、パラメータ・ファイルの DISK_ASYNCH_IO 初期化パラメータは true に設定され、RAW デバイスに対する非同期 I/O が有効になっています。ファイル・システムのファイルに対して非同期 I/O を有効にするには、次の手順を実行します。

1. すべての Oracle データベース・ファイルが、非同期 I/O をサポートしているファイル・システム上にあることを確認します。
2. パラメータ・ファイル内の FILESYSTEMIO_OPTIONS 初期化パラメータを次のいずれかの値に設定します。

Linux ディストリビューション	推奨値
SUSE Linux Enterprise Server 9	setall
その他のディストリビューション	asynch

ダイレクト I/O サポート

ダイレクト I/O サポートは、Red Hat Enterprise Linux 2.1 および SUSE Linux Enterprise Server 8 では使用できず、サポートされていません。Red Hat Enterprise Linux 3 および SUSE Linux Enterprise Server 9 では使用可能であり、サポートされています。

注意： Red Hat Enterprise Linux 3 でダイレクト I/O を使用するには、使用するドライバが vary I/O をサポートしている必要があります。

ダイレクト I/O サポートを使用可能にするには、次の手順を実行します。

- FILESYSTEMIO_OPTIONS 初期化パラメータを directIO に設定します。
- 非同期 I/O オプションを使用している場合、FILESYSTEMIO_OPTIONS 初期化パラメータを setall に設定します。

semtimedop のサポート

注意： semtimedop 機能は Red Hat Enterprise Linux 2.1 では使用できません。

Red Hat Enterprise Linux 3 および SUSE Linux Enterprise Server では、Oracle Database 10g で semtimedop() システム・コール（時間制限付きセマフォ）がサポートされます。ただし、SUSE Linux Enterprise Server 8 では、この機能はデフォルトで無効になっています。この機能のサポートを有効にするには、\$ORACLE_HOME/rdbms/lib ディレクトリで oracle ユーザーとして次のコマンドを入力します。

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

semtimedop() のサポートを無効にするには、\$ORACLE_HOME/rdbms/lib ディレクトリで oracle ユーザーとして次のコマンドを入力します。

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

x86 の場合のみ：高速ネットワークのサポート

x86 システムの場合のみ、Oracle Database 10g リリース 1 (10.1) の Oracle Net では、Red Hat Enterprise Linux AS 2.1 および 3 で InfiniBand ネットワーク・アーキテクチャを介した Sockets Direct プロトコル (SDP) がサポートされます。このリリースでは、SDP のサポートは同期 I/O のみの限定です。SDP を介した非同期 I/O 使用のサポートの詳細は、次の Web サイトで最新情報を確認してください。

http://www.oracle.com/technology/products/oraclenet/files/Oracle_Net_High-Speed_Interconnect_Support.doc

注意： この Web サイトで特に示されていないかぎり、Oracle Net の NET_ASYNC_IO および SDP_ASYNC_IO 構成パラメータを設定しないでください。

Mac OS X での Oracle Database の管理

この付録では、Mac OS X で Oracle Database を管理する方法について説明します。

使用可能および使用済のスワップ領域の決定

Mac OS X では、/private/var/vm ディレクトリに必要な応じてスワップ・ファイルが動的に作成されます。Oracle Database を実行する場合、新規作成されたスワップ・ファイルに対応するために、ルート (/) ファイル・システムに使用可能なディスク領域が 1GB 以上あることを確認してください。

/private/var/vm ディレクトリで使用可能なディスク領域のサイズを判別するには、次のコマンドを入力します。

```
$ df -k /private/var/vm
```

現在使用中のスワップ領域のサイズを判別するには、次のコマンドを入力します。

```
$ du -sk /private/var/vm/swapfile*
```

Tru64 UNIX システムでの Oracle Database の 管理

この付録では、Tru64 UNIX システムで Oracle Database を管理する方法について説明します。
次の項目について説明します。

- Oracle Database 指定配置最適化の有効化
- 複合 CPU システムのサポート
- Tru64 UNIX でのデータベース統計の収集
- Tru64 UNIX での Oracle Real Application Clusters
- 非同期 I/O のチューニング
- ダイレクト I/O サポートおよびコンカレント・ダイレクト I/O サポート
- リアルタイム・クロックへのアクセスの有効化
- RAW デバイスの設定
- Spike 最適化ツール

Oracle Database 指定配置最適化の有効化

HP AlphaServer GS Series、ES47 および ES80 システムは、リソース・アフィニティ・ドメイン (RAD) と呼ばれる小さな構築ブロックで構成されています。RAD は、高速インターコネクトで接続された密結合 CPU、メモリー・モジュールおよび I/O コントローラの集合です。第 2 レベルのインターコネクトでは、各 RAD が接続されて、より大きな構成が形成されます。

CPU、メモリーおよび I/O コントローラ間の共有インターコネクトが 1 つのみであった以前のサーバーとは異なり、GS Series、ES47 および ES80 サーバーでは、特定の CPU によるその RAD 内のメモリーへのアクセス時またはローカル I/O コントローラ使用時のパフォーマンスとメモリー・アクセス時間が向上しています。インターコネクトが切り替わるため、特定の RAD 内の I/O アクティビティやメモリー・アクセスによって、別の RAD 内の I/O アクティビティやメモリー・アクセスが妨げられることはありません。ただし、RAD 境界にまたがって配置されている CPU とメモリー・モジュール間にメモリー・アクセスを行った場合は、2 つのレベルのインターコネクト階層を横断する必要があります。この場合、1 つの RAD 内でメモリーを参照する場合に比べ、多くの参照時間を必要とします。

メモリーとプロセスの指定配置サポートを使用することによって、高度なアプリケーション固有のプロセスおよびメモリーのレイアウトに関する要件をオペレーティング・システムに伝達できます。この機能によって、特定の RAD 内のメモリー参照を詳細に設定できるため、パフォーマンスが向上します。

Oracle Database では、GS Series、ES47、ES80 などの高性能サーバーの特別な機能に対するサポートが強化されています。指定配置最適化では、これらのサーバーで使用できる階層型インターコネクトを利用しています。以前のサーバーでは、共有インターコネクトが 1 つのみであったため、指定配置最適化の利点を直接受けることもなく、これらのサーバーのパフォーマンスが低下することもあります。したがって、Oracle Database では指定配置最適化がデフォルトで無効になっています。

指定配置最適化を実行するための要件

Oracle Database の指定配置最適化を使用するには、次のシステム要件を満たす必要があります。

- HP GS Series、ES47 または ES80 AlphaServer など、ローカル・パフォーマンスを重視したシステムを使用する必要があります。Oracle Database の最適化は、ローカル・パフォーマンスを重視したシステムにのみ効果があります。
- オペレーティング・システムは Tru64 UNIX V5.1B 以上を使用する必要があります。

Oracle 指定配置最適化の有効化

Oracle 指定配置最適化を有効にするには、次の手順を実行します。

1. Oracle インスタンスをシャットダウンします。
2. 次のコマンドを入力して、Oracle Database を再リンクします。

```
$ cd $ORACLE_HOME/rdbms/lib
$ make -f ins_rdbms.mk numa_on
$ make -f ins_rdbms.mk ioracle
```

互換性のあるバージョンの Tru64 UNIX を使用していない場合は、次のメッセージが表示されます。

```
Operating System Version Does not Support NUMA.
Disabling NUMA!
```

Oracle 指定配置最適化を有効にしてから、非互換バージョンの Tru64 UNIX に変更する場合は、次の項で説明する手順に従って、Oracle 指定配置最適化を無効にしてください。

Oracle 指定配置最適化の無効化

Oracle 指定配置最適化を無効にするには、次の手順を実行します。

1. Oracle インスタンスをシャットダウンします。
2. numa_off オプションを使用して、Oracle Database を再リンクします。

```
$ cd $ORACLE_HOME/rdbms/lib
$ make -f ins_rdbms.mk numa_off
$ make -f ins_rdbms.mk ioracle
```

Oracle 指定配置最適化の使用

Oracle 指定配置最適化は、等しくパーティション化された構成を想定しています。つまり、すべての RAD は同じ数の CPU と同じ量のメモリーで構成されていることを想定します。Oracle Database は、システム上のすべての RAD にわたって稼動することを想定します。

Oracle 初期化パラメータ

Oracle Database では、ローカル環境を最も効率的に使用するために、オペレーティング・システムから報告されたサーバー構成に応じて、いくつかの初期化パラメータが自動的に調整されます。この調整によって、これらの初期化パラメータ間のいくつかの依存関係を計算する際に発生する一般的なエラーが除去されます。

Tru64 UNIX サブシステム属性

NUMA システムの利点をすべて実現するには、次の表に示すサブシステム属性を設定する必要があります。

サブシステム	属性	設定
ipc	ssm_threshold	0
	shm_allocate_striped	1 (デフォルト)
vm	rad_gh_regions[0]、 rad_gh_regions[1]、 以下同様	rad_gh_regions[n] 属性を次のように設定します。 1. 次のコマンドを入力します。 \$ ipcs -b 2. 出力から SEGSZ 列の値を追加し、合計を 1048576 で除算して MB に変換します。 3. LOG_BUFFER 初期化パラメータの値が 8388608 (8MB) 未満の場合、合計から 8388608 を差し引きます。LOG_BUFFER の値が 8388608 ~ 16777216 (16MB) の間である場合、SEGSZ 出力の合計から 16777216 を差し引きます。 4. 手順 3 の結果をシステムの RAD 数で除算します。 5. rad_gh_regions[0] パラメータを、手順 4 の結果に 8MB または 16MB を加えた値に設定します。どちらを加えるかは、LOG_BUFFER 初期化パラメータおよびその他の有効な rad_gh_regions[n] 属性の値によって異なります。 これらの手順では、システム上のすべての RAD が Oracle に割り当てられていることを前提としています。 システムに存在する RAD 数のサブセットを使用するように Oracle が制限されている場合、E-4 ページの「システムに存在する RAD 数のサブセットへの Oracle Database の実行制限」の項を参照してください。 numa_config_sid.ora ファイルにリストされた最初の RAD については、前述の手順のかわりに、rad_gh_regions[0] 設定を rad_gh_regions[n] パラメータに適用する必要があります。

Tru64 UNIX V5.1B の `vm` サブシステムには、`rad_gh_regions` 属性が 63 個あります。システム上の RAD の合計数に応じて、これらの属性を設定してください。たとえば、システムに 4 つの RAD があり、これらがすべて Oracle で使用され、`SEGSZ` 列の合計が 10248MB で、`LOG_BUFFER` 初期化パラメータが 2097152 (2MB) に設定されている場合、`rad_gh_regions[0]` を 2568 に設定し、`rad_gh_regions[1]`、`rad_gh_regions[2]` および `rad_gh_regions[3]` を 2560 に設定します。インスタンスを正常に起動させるには、この値をわずかに (1 または 2) 高くする必要があります。

CPU とメモリーがオフラインの場合、Oracle Database は引き続き動作しますが、パフォーマンスは低下します。RAD が頻繁にオフラインになったり、等しくパーティション化できないような場合は、RAD ごとに 1 つのインスタンスを使用して、Oracle Real Application Clusters を実行することをお勧めします。Oracle Real Application Clusters を使用すると、実際の RAD 構成に応じて初期化パラメータを様々に組み合わせ、個々のインスタンスを構成できます。また、アプリケーションを全体的に使用可能にしたままで、特定のインスタンスを起動またはシャットダウンできます。

RAD に対するプロセスの親和性

特定の RAD 上でプロセスを実行するようにオペレーティング・システムに設定することによって、パフォーマンスを向上させることができます。Oracle リスナー・プロセスを介してデータベースに接続し、対応するネットワーク・インターコネクト・アダプタが RAD 上にある場合は、各 RAD 上でリスナーを実行できます。特定の RAD 上でリスナーを実行するには、次のコマンドを入力します。`rad_number` は RAD の数です。

```
$ runon -r rad_number lsnrctl start [listener_name]
```

すべての Oracle シャドウ・プロセスが、Oracle リスナーと同じ RAD 上に自動的に作成されます。

システムに存在する RAD 数のサブセットへの Oracle Database の実行制限

Oracle Database の実行を、システムに存在する RAD の数のサブセットに制限できます。インスタンスの起動時に、Oracle Database は `$ORACLE_HOME/dbs` ディレクトリの `numa_config_sid.ora` ファイルを検索します。ファイルがない場合、Oracle Database はシステム上のすべての RAD を使用します。ファイルが見つかった場合、Oracle Database はインスタンスに対して、ファイルに指定されている RAD のみを使用して、指定の RAD のみでプロセスを実行するように指示します。

`numa_config_sid.ora` ファイルは、次のような形式になっています。

```
number_RADs
group1
...
groupn
```

この例では、`number_RADs` は Oracle が使用するプロセッサ・グループまたは RAD の数、`group1 ~ groupn` は、Oracle が使用する RAD の番号です。

ファイルを作成するときは、次のガイドラインに従ってください。

- 空白の行は挿入しない
- 個々の行に番号を指定する
- 番号以外は行に一切指定しない

各 RAD には、特定のシステムに対する番号が 0 (ゼロ) から最大数まで付けられます。システムが完全に統合化されている AlphaServer GS320 の場合、RAD には、0、1、2、3、4、5、6、7 と番号が付きます。たとえば、完全に構成された GS320 の RAD のうち、Oracle Database が 1 と 3 を使用するように制限するには、`numa_config_sid.ora` ファイルを次のように作成します。

```
2
1
3
```

グループ番号は、連続したり昇順の番号にする必要はありません。

Bequeathed 接続のためのパラレル問合せスレーブとシャドウ・プロセスは、ファイルに指定されている RAD 上に作成されます。リモート TCP 接続の場合は、RAD にリスナーを明示的にバインドする必要があります。詳細は、E-4 ページの「[RAD に対するプロセスの親和性](#)」を参照してください。

複合 CPU システムのサポート

Tru64 UNIX V5.1B 以上を使用する Tru64 UNIX システムでは、異なる CPU 速度およびタイプを組み合わせて動作できます。1 つの RAD 内の CPU はすべて、速度およびキャッシュ・サイズが同じである必要があります。別の RAD には、速度およびキャッシュ・サイズの異なる CPU を指定できます。

複合 CPU システムのパフォーマンスは、低速 CPU と高速 CPU の比率によって変わります。また、システム上の Oracle プロセスの配置にも左右されます。高速トランザクションのオンライン・トランザクション処理 (OLTP) 環境では、データベース・ライター・プロセスおよびログ・ライター・プロセスを低速 CPU に配置すると、パフォーマンスが低下する可能性があります。一方、データ・ウェアハウスや意思決定支援環境では、データベース・ライター・プロセスおよびログ・ライター・プロセスを低速 CPU に配置しても、パフォーマンスはまったく変わらないことがあります。

CPU システムを組み合わせることによって、ハードウェア資産を保護できます。古い CPU を交換せずに、より高速で強力な CPU をシステムに追加できます。HP 社とオラクル社では、複合 CPU システムのテストを完了し、現在この機能をサポートしています。

注意： この複合 CPU システムのパフォーマンスは、最高速 CPU のみで構成された複合 CPU システムと同等のパフォーマンスにはなりません。ただし、最低速 CPU のみで構成された複合 CPU システムよりは高くなります。複合 CPU システムの詳しい規則と制限事項は、HP 社にお問い合わせください。

Tru64 UNIX でのデータベース統計の収集

Oracle Database 10g は、Tru64 UNIX V5.1B 以上でのみ動作します。これは、Tru64 UNIX の long double データ型のサイズが、V4.0x では 64 ビットでしたが、V5.x では 128 ビットに変更されたためです。この変更によって、いくつかの操作の精度が高くなりました。これらの操作を実行すると、表または索引が分析され、統計情報がデータ・ディクショナリに格納されます。

Oracle Database 内の問合せオブティマイザでは、データ・ディクショナリに格納されている統計情報を使用して、最適な問合せ方法を判断します。最適な問合せ方法を判断しているときに、データ・ディクショナリに格納されている統計と問合せオブティマイザによって算出された統計とが一致しない場合、問合せオブティマイザは誤った方法で問合せを実行することがあります。この場合、問合せが正しく実行されなかったり、失敗する可能性があります。

このため、Oracle8i リリース 8.1.7 以前から Oracle Database 10g にアップグレードした後は、各スキーマのオブジェクト統計をすべて分析する必要があります。Oracle9i リリース 1 (9.0.1) またはリリース 2 (9.0.2) から Oracle Database 10g にアップグレードした後は、スキーマを分析しなおす必要はありません。DBMS_STATS.GATHER_SCHEMA_STATS プロシージャを使用して、分析を実行し、スキーマごとに統計情報を収集できます。DBMS_STATS パッケージでは、新しい統計に問題が発生した場合に備えて、現在の表または索引の統計を表に保存しています。

関連項目： データベース統計の収集の詳細は、『PL/SQL パッケージ・プロシージャおよびタイプ・リファレンス』を参照してください。

Tru64 UNIX での Oracle Real Application Clusters

この項では、Tru64 UNIX での Oracle Real Application Clusters について説明します。

Reliable Data Gram

Reliable Data Gram (RDG) は、Tru64 UNIX TruCluster プラットフォームの IPC インフラストラクチャの 1 つです。RDG は、Tru64 UNIX 上の Oracle Database のデフォルトの IPC 方式で、Oracle Real Application Clusters 環境に最適化されています。

要件

RDG を使用するには、ノードがクラスタに所属し、メモリー・チャンネルを介して接続する必要があります。また、RDG を使用する場合は、表 E-1 に示すサブシステム属性をノード全体に設定することをお勧めします。

表 E-1 rdg サブシステム属性の設定

属性	設定
max_objs	ノード当たりの Oracle プロセス数の 5 倍以上で、10240 または Oracle プロセス数の 70 倍以下。
msg_size	データベースの DB_BLOCK_SIZE パラメータの最大値以上。 Oracle Database は各表領域の様々なブロック・サイズに対応しているため、推奨値は 32768 です。
max_async_req	100 またはオペレーティング・システムのデフォルト値のいずれか大きい値。 注意: 多くの場合、1000 以上の値を指定するとパフォーマンスが向上します。
max_sessions	170 (20 + Oracle の PROCESSES 初期化パラメータの値)。
rdg_max_auto_msg_wires	常に 0 (ゼロ)。

UDP IPC の有効化

Oracle Database 10g では、RDG が Tru64 UNIX のデフォルトの IPC 方式です。Oracle Real Application Clusters オプションが有効になっている場合は、グローバル・キャッシュ・サービス (GCS)、グローバル・エンキュー・サービス (GES)、インタープロセッサ・パラレル問合せ (IPQ) およびキャッシュ・フュージョンで RDG が使用されます。ユーザー・データグラム・プロトコル (UDP) IPC の実装も利用できますが、明示的に有効にする必要があります。

UDP IPC を有効にする場合は、事前に Oracle Real Application Clusters オプションを有効にしておく必要があります。Oracle Real Application Clusters オプションを有効にするには、Oracle Universal Installer を使用するか、次のコマンドを入力します。

```
$ cd $ORACLE_HOME/rdbms/lib
$ make -f ins_rdbms.mk rac_on
$ make -f ins_rdbms.mk ioracle
```

Oracle IPC ルーチンで UDP プロトコルを使用するには、oracle 実行可能ファイルを再リンクする必要があります。次の手順を実行する前に、クラスタ内のインスタンスをすべてシャットダウンしてください。

UDP IPC を有効にするには、次のコマンドを入力します。

```
$ cd $ORACLE_HOME/rdbms/lib
$ make -f ins_rdbms.mk ipc_udp
$ make -f ins_rdbms.mk ioracle
```

UDP IPC を無効にし、Oracle Real Application Clusters のデフォルトの実装 (RDG) に戻すには、次のコマンドを入力します。

```
$ cd $ORACLE_HOME/rdbms/lib
$ make -f ins_rdbms.mk rac_on
$ make -f ins_rdbms.mk ioracle
```

UDP および CLUSTER_INTERCONNECTS パラメータの設定の詳細は、1-8 ページの「[CLUSTER_INTERCONNECTS 初期化パラメータ](#)」を参照してください。

Oracle Real Application Clusters と NUMA の同時有効化

Oracle Real Application Clusters と指定配置 (NUMA) を同時に実行できます。AlphaServer GS Series、ES47 および ES80 サーバーのほとんどの作業負荷については、Oracle Database 10g リリース 1 (10.1) および Real Application Clusters を実行しているシステムで、NUMA オプションを有効にすることによって、パフォーマンスが大幅に向上します。

非同期 I/O のチューニング

Oracle Database for Tru64 UNIX システムでは、同期 I/O または非同期 I/O を実行できます。パフォーマンスを向上させるには、非同期 I/O を使用することをお勧めします。非同期 I/O を有効にするには、DISK_ASYNC_IO 初期化パラメータを true に設定します。

Oracle Database では、Automatic Storage Management ディスク・グループ内の AdvFS ファイル・システムまたはクラスタ・ファイル・システム (CFS)、あるいは RAW デバイスに格納されているデータファイルに対して、非同期 I/O を使用できます。非同期 I/O のパフォーマンスを最適化するには、いくつかのカーネル・サブシステム属性をチューニングする必要があります。

aio_task_max_num 属性

単一インスタンスの aio_task_max_num カーネル・サブシステム属性は、8193 に設定してください。

aio_task_max_num 属性の設定は、単一ノードで複数の Oracle Database インスタンスを使用する場合など、非同期 I/O を使用する他のすべてのアプリケーションを考慮して調整する必要があります。このパラメータの値は、アプリケーションが発行できる I/O 要求の最大数に設定してください。たとえば、3 つのアプリケーションが動作しているときに、アプリケーション 1 が最大 10 個の非同期 I/O 要求を同時に発行でき、アプリケーション 2 が 100 個の非同期 I/O 要求を同時に発行でき、アプリケーション 3 が 1000 個の非同期 I/O 要求を同時に発行できる場合は、aio_task_max_num パラメータを 1000 以上に設定します。

aio_task_max_num 属性を設定しないと、Oracle Database のパフォーマンスが低下し、I/O エラーが発生する可能性があります。これらのエラーは、アラート・ログ・ファイルおよびトレース・ファイルに記録されます。

ダイレクト I/O サポートおよびコンカレント・ダイレクト I/O サポート

この項では、ダイレクト I/O およびコンカレント I/O のサポートについて説明します。

単一インスタンスの要件

単一インスタンスをインストールする場合の Oracle Database の要件は、次のとおりです。

- Tru64 UNIX V5.1B 以上で、適切なパッチ・キットを適用している必要があります。

関連項目： Tru64 UNIX のパッチ・キットの詳細は、『Oracle Database インストール・ガイド for UNIX Systems』を参照してください。

- Oracle データファイルは、AdvFS ファイル・システムまたは ASM ディスク・グループに格納されている必要があります。
- AdvFS ファイル・システムを使用するディスクは、Oracle Database インスタンスを実行しているコンピュータに物理的に接続する必要があります。これには、ファイバー・チャネルによって接続されるディスクも含まれます。物理的な接続がなく、I/O 処理を別のノードで行う必要がある場合は除きます。

Tru64 UNIX V5.1B 以上のシステムが非クラスタ化システム環境で稼動している場合は、ファイル・システム・キャッシュが使用されないため、RAW デバイスのほとんどは AdvFS ファイル・システムとダイレクト I/O によって処理されます。さらに、このファイル・システムを使用すると、データベース・ファイルの管理が簡素化されます。

クラスタ化されたシステム

Tru64 UNIX V5.1B 以上のシステムは、クラスタ・ファイル・システム (CFS) に対応しています。CFS では、クラスタ内のすべてのノードで単一名前空間ファイル・システムが使用されています。クラスタにマウントされたすべてのファイル・システムは、同じクラスタ内のすべてのノードに自動的に公開されます。CFS ファイル・システムは、AdvFS ファイル・システムの最上位に位置するため、非クラスタ化システムの特徴をほぼ継承します。

Tru64 UNIX V5.1B クラスタ化システム

CFS ファイル・システムは、現在コンカレント・ダイレクト I/O モデルに対応しているため、Tru64 UNIX V5.1B 以上のシステムでのみサポートされています。ドライブに物理的に接続されているノードは、所有元のノードに問い合わせなくてもそのファイル・システムにデータ I/O を発行できます。

アクセスまたは変更日の延長、クローズ、変更など、ファイルのメタデータを変更すると、その変更はすべて所有ノードによって処理されるため、クラスタのインターコネクトが飽和状態になる可能性があります。インターコネクトが飽和状態になると、CFS ファイル・システムでの CREATE TABLESPACE、ALTER TABLESPACE、ADD DATAFILE、ALTER DATABASE DATAFILE または RESIZE コマンドのパフォーマンスが、RAW デバイスよりも低下することがあります。

複数インスタンスの要件（Oracle Real Application Clusters）

データファイルの格納に CFS ファイル・システムを使用する場合、Oracle Real Application Clusters では、システム上でダイレクト I/O が有効になっている必要があります。CFS ファイル・システムを使用するディスクは、Oracle インスタンスを実行しているすべてのコンピュータに物理的に接続する必要があります。これには、ファイバー・チャネルによって接続されるディスクも含まれます。物理的な接続の理由から、I/O の処理を別のノードで行う必要がある場合は、この対象外です。

データベースがアーカイブ・ログ・モードで動作し、アーカイブ・ログがディスクに書き込まれる場合は、書込み先の CFS ドメインを、REDO ログを保管しているインスタンスのノードにする必要があります。たとえば、nodea、nodeb および nodec の 3 つのノードから構成されるクラスターで、各ノードにインスタンスが 1 つずつ動作している場合、アーカイブ・ログが書き込まれる CFS ドメインは、arcnodea、arcnodeb および arcnodec の 3 つです。これらのドメイン名の場所は、nodea、nodeb および nodec として、各インスタンスの LOG_ARCHIVE_DEST 初期化パラメータに指定する必要があります。

ダイレクト I/O サポートの無効化

ダイレクト I/O サポートが有効になっている AdvFS ファイル・システム上で動作している Oracle Database は、RAW デバイスで動作している Oracle Database と同等のパフォーマンスを実現します。ほとんどの場合、ダイレクト I/O サポートが有効になっている AdvFS ボリュームに格納されている Oracle Database は、ダイレクト I/O サポートが無効になっている同等のデータベース以上のパフォーマンスを実現します。ただし、ダイレクト I/O サポートが有効になっている場合は、次の作業負荷条件によってパフォーマンスが低下する可能性があります。

- 読取り / 書込みの比率が高い
- 問合せでパラレル問合せスレーブが使用されるため、Oracle データ・ブロックが SGA にキャッシュされない
- UNIX バッファ・キャッシュ（UBC）のサイズが数 MB 以上ある
- 全表スキャンによって、同じ表セットが繰り返しスキャンされる
- スキャンされる表が UBC にキャッシュされる

ダイレクト I/O サポートが無効になっている場合、前述のほとんどの条件に当てはまる作業負荷は UBC に大きく依存します。スキャンされる表のすべてが UBC にキャッシュされるわけではありませんが、ほとんどの場合、パラレル問合せによって発行された I/O 要求は UBC から返されます。したがって、データのすべてをディスクから読み込む場合、つまりダイレクト I/O が有効になっている場合に比べ、問合せがより速く返されます。

ダイレクト I/O サポートが有効になっている場合、Oracle データ・ブロックは UBC にキャッシュされません。それらのデータ・ブロックは、プロセス専用メモリーに読み込まれます。つまり、以前にスキャンした表を読み込む問合せの場合も、ディスクへの I/O 要求を実行して、データを取得する必要があります。ディスク I/O 待機時間は、メモリー待機時間よりも大幅に長くなります。このため、問合せの実行速度が遅くなり、パフォーマンスが低下します。

作業負荷が前述のほとんどの条件に当てはまる場合は、ダイレクト I/O サポートを無効にすることによって、パフォーマンスが改善する可能性があります。ただし、多くの場合、様々な種類の問合せが同時に実行されています。データを単に読み込む問合せもある一方で、データを挿入、変更または削除する問合せもあります。これらの問合せの比率は環境ごとに異なります。一般に、OLTP 作業負荷の比率が高い環境では、ダイレクト I/O サポートを無効にしてもパフォーマンスは改善しません。

Oracle Database 10g では、ダイレクト I/O サポートがデフォルトで有効になっています。Oracle9i リリース 1 (9.0.1) でダイレクト I/O サポートを無効にするための `_TRU64_DIRECTIO_DISABLED` 初期化パラメータ（マニュアルには記載されていません）は、Oracle Database 10g では削除されました。かわりに、汎用の `FILESYSTEMIO_OPTIONS` 初期化パラメータが使用されます。次の表に、Tru64 UNIX で解釈される `FILESYSTEMIO_OPTIONS` 初期化パラメータの有効な値を示します。

値	説明
directIO	AdvFS ファイル・システムのファイルの I/O に対して、ダイレクト I/O サポートを有効にし、非同期 I/O サポートを無効にします。
asynch	none と等価です。AdvFS ファイルに対して、ダイレクト I/O サポートが有効である場合にのみ非同期 I/O サポートを有効にします。
setall	AdvFS ファイルに対して、ダイレクト I/O サポートおよび非同期 I/O サポートを有効にします。これはデフォルトのオプションです。
none	AdvFS ファイルに対して、ダイレクト I/O サポートおよび非同期 I/O サポートを無効にします。

関連項目： FILESYSTEMIO_OPTIONS 初期化パラメータの詳細は、『Oracle Database 概要』を参照してください。

DISK_ASYNC_IO 初期化パラメータは、ファイル・システムまたは RAW デバイスのいずれのファイルであっても、すべてのデータベース・ファイルの非同期 I/O 状態を制御します。したがって、DISK_ASYNC_IO 初期化パラメータが `false` に設定されている場合、ファイル・システムのファイルへの I/O 要求はすべて、FILESYSTEMIO_OPTIONS 初期化パラメータの値に関係なく同期化されます。DISK_ASYNC_IO 初期化パラメータは、デフォルトで `true` に設定されています。

リアルタイム・クロックへのアクセスの有効化

Oracle プロセスの多くは、特に TIMED_STATISTICS 初期化パラメータが `true` に設定されている場合は、各種処理時間が測定されます。これらのタイミング機能は、Tru64 UNIX カーネルを呼び出すため、Oracle Database のパフォーマンスに影響する可能性があります。Tru64 UNIX では、プロセスからリアルタイム・クロックに直接アクセスできるため、負荷の大きいシステムのパフォーマンスを改善できます。

リアルタイム・クロックへのアクセスを有効にするには、次の手順を実行します。

1. root ユーザーでログインします。
2. 次のコマンドを入力します。

```
# mknod /dev/timedev c 15 0
# chmod a+r /dev/timedev
```

Tru64 UNIX V5.1B 以上が稼動するクラスタ化システムの場合は、これらのコマンドを各クラスタ・ノードで入力します。

注意： 特殊ファイル `/dev/timedev` は、リブート後もシステムに残ります。

3. Oracle Database インスタンスを再起動します。

`/dev/timedev` ファイルが存在しているかどうかは、インスタンスの起動時にのみチェックされます。

この機能は、クラスタ内のすべてのインスタンス、つまりすべてのノードで有効にすることをお勧めします。

RAW デバイスの設定

注意： RAW デバイスの設定には、経験の豊富なシステム管理者の協力と、使用しているシステムに関する専門知識が必要です。

注意： データベース・ファイル管理作業を簡素化するために、ダイレクト I/O には RAW デバイスよりも ASM または AdvFS を使用することをお勧めします。

Tru64 UNIX システム上で RAW デバイスおよび RAW ボリュームを設定するには、次の手順を実行します。

1. Oracle Real Application Clusters を使用している場合、追加するパーティションは共有ディスク上に配置します。
2. 空きディスク・パーティションの名前を確認します。
 空きディスク・パーティションは、Tru64 UNIX ファイル・システムに使用されていないパーティションで、次の制限に準拠している必要があります。
 - /usr/sbin/mount コマンドの入力で、リストに表示されないこと。
 - スワップ・デバイスとして使用されていないこと。
 - スワップ・パーティションをオーバーラップしていないこと。
 - 他の Tru64 UNIX アプリケーション（Oracle Database の他のインスタンスなど）で使用されていないこと。
 - Tru64 UNIX ファイル・システムをオーバーラップしていないこと。
 - ファイル・システムによってすでに使用されている領域を使用しないこと。

パーティションが空いているかどうかを判断するには、デバイス上のパーティションの開始位置とサイズを示す詳細なマップを出力し、空き領域を確認します。パーティションの中には、現在マウントされずに /usr/sbin/mount の出力に表示されないファイル・システムもあります。

注意： パーティションがシリンダ 0（ゼロ）で開始していないことを確認してください。

3. Oracle Database で使用する RAW デバイスを設定します。
 ディスクがパーティション化されているかどうかを確認します。パーティション化されていない場合は、disklabel コマンドを使用してディスクをパーティション化します。
4. ls コマンドを入力して、デバイス・ファイルの所有者と権限を表示します。たとえば、次のように入力します。

```
$ ls -la
```
5. パーティションの所有者が Oracle ソフトウェア所有者であることを確認します。必要に応じて、chown コマンドを使用して、デバイスのブロック・ファイルおよびキャラクタ・ファイルの所有者を変更します。たとえば、次のように入力します。

```
# chown oracle:dba /dev/rdisk/dsk10c
```

- パーティションに正しい権限が付与されていることを確認します。必要に応じて、`chmod` コマンドを使用して、**Oracle** ソフトウェア所有者のみがそのパーティションにアクセスできるようにします。たとえば、次のように入力します。

```
# chmod 600 /dev/rdisk/dsk10c
```

- 目的の RAW デバイスへのシンボリック・リンクを作成します。たとえば、次のように入力します。

```
$ ln -s /dev/rdisk/dsk10c /oracle_data/datafile.dbf
```

シンボリック・リンクが作成されたことを確認するには、文字型特殊デバイス（ブロック型特殊デバイスではなく）を使用し、次のコマンドを入力します。

```
$ ls -l datafile
```

次の出力が表示されます。

```
crwxrwxrwx oracle dba datafile
```

注意： このシンボリック・リンクは、クラスタの各ノードに設定する必要があります。複数のシンボリック・リンクが同一の RAW デバイスを指定していないことを確認してください。

- 新しいパーティションを新しいデータベースに作成または追加します。

新しいパーティションを作成するには、**SQL*Plus** から次のコマンドを入力します。

注意： RAW パーティションに作成する Oracle データファイルのサイズは、RAW パーティションのサイズと比較して、64KB に Oracle ブロック・サイズ 1 つ分を加えた値以上小さくする必要があります。

```
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;
```

パーティションを既存の Oracle データベースの表領域に追加するには、次のコマンドを入力します。

```
SQL> ALTER TABLESPACE tablespace_name
2 ADD DATAFILE '/dev/rdisk/dsk10c' SIZE 10000K REUSE;
```

同じ手順を使用して、REDO ログ・ファイルの RAW デバイスを設定できます。

Spike 最適化ツール

Spike 最適化ツール (Spike) は、Tru64 UNIX バイナリのパフォーマンスを改善するパフォーマンス最適化ツールです。OLTP 作業負荷を伴ったテスト環境で Spike (フィードバック付き) を使用したところ、Oracle Database のパフォーマンスが最大 23 パーセント向上しました。

Spike の詳細は、Tru64 UNIX のドキュメントを参照するか、または次のいずれかのコマンドを入力してください。

- `man spike`
- `spike`

Oracle Database には、Spike V5.2 (510 USG、GEM: 48C5K、LIBMLD: 2.4、DATE: 2003 年 9 月 28 日) 以上が必要です。

注意: Spike のバージョンが V5.2 (510 USG、GEM: 48C5K、LIBMLD: 2.4、DATE: 2003 年 9 月 28 日) より前である場合、HP に連絡してパッチ・キットを入手してください。

Spike のバージョンを確認するには、次のコマンドを入力します。

```
$ spike -V
```

HP 社の Web サイトから最新バージョンの Spike をダウンロードできます。

注意: オラクル社では、`spike` コマンドを使用して最適化された Oracle 実行可能ファイルをサポートしていません。Spike を使用して最適化された Oracle Database バイナリに問題がある場合は、元の最適化されていないバイナリを使用してその問題を再現してください。

Spike の使用

この項では、Spike に必要なシステム・リソース、Spike 最適化フラグを使用する方法と理由、および Spike を実行するいくつかの方法について説明します。

システム・リソースの設定

表 E-2 に、Spike の実行に必要なシステム・リソースを示します。

表 E-2 Spike のシステム・リソース要件

リソース	最小値
物理メモリー	1024MB
max-per-proc-address-space サブシステム属性の値	1024MB
max-per-proc-data-space サブシステム属性の値	1024MB
vm-maxvas サブシステム属性の値	1024MB

/etc/sysconfigtab ファイルのこれらのサブシステム属性値を設定するには、次の行を追加します。

```
proc:
    max-per-proc-address-space = 0x40000000
    max-per-proc-data-size = 0x40000000
vm:
    vm-maxvas = 0x40000000
```

シェル環境の制限を最大値に設定します。C シェルの場合は、次のコマンドを入力します。

```
% limit datasize unlimited
% limit memoryuse unlimited
% limit vmemoryuse unlimited
```

スタックサイズの制限を高く設定しすぎると、**Spike**によって仮想メモリ不足が発生する可能性があります。この問題を回避するには、次の C シェル・コマンドを入力します。

```
% limit stacksize 8192
```

最適化フラグのチェック

Spike には、多数の最適化フラグが用意されています。ただし、**spike** コマンドによる最適化すべてを **Oracle Database** に使用できるわけではありません。**Oracle Database** では、次の **Spike** 最適化フラグの実行が認定されています。

```
-arch, -controlOpt, -fb, -feedback, -map, -nosplit, -nochain, -noporder,
-noaggressiveAlign, -o, optThresh, -splitThresh, -symbols_live, -tune, -v, -V
```

Spike を実行すると、最適化フラグのコピーが、最適化しているバイナリのイメージ・ヘッダー・コメント・セクションに格納されます。**Oracle Database** では、インスタンスの起動時に、使用される **Spike** 最適化をチェックします。**Oracle Database** で認定されていない最適化が検出された場合、または以前に OM (HP 社製の **Spike** の先行モデル) を使用してバイナリを最適化していた場合は、インスタンスの起動に失敗し、ORA-04940 エラー・メッセージが表示されます。インスタンスの起動に失敗した場合は、アラート・ログ・ファイルの詳細情報を確認してください。

注意： **Oracle Database** では、**Spike** 最適化フラグ **-symbols_live** を使用する必要があります。

Spike の実行

Spike を使用して実行可能ファイルを最適化するには、次のいずれかの方法を使用します。

- 静的 **Spike**
- フィードバック付き **Spike** の実行

静的 **Spike** は、いくつかの手順の設定のみで、フィードバック付き **Spike** の約半分のパフォーマンスを実現できます。

フィードバック付き **Spike** は、静的 **Spike** によるすべての最適化以外に、作業負荷関連の最適化をいくつか実行できます。フィードバック付き **Spike** を実行すると、パフォーマンスは最も向上しますが、静的 **Spike** に比べて多くの操作を必要とします。

フィードバック付き **Spike** および静的 **Spike** を両方実行する場合は、**Spike** を実行した **Oracle** バイナリをテスト環境で実行してから、本番環境に移行することをお勧めします。

静的 Spike

静的 **Spike** では、グローバル・ポインタ (gp)・レジスタの操作や CPU アーキテクチャの有効利用など、作業負荷以外の最適化が実行されます。テスト環境では、**Spike** によるパフォーマンス最適化の約半分が、静的 **Spike** で実現されました。さらに、静的 **Spike** は比較的わかりやすく、簡単に実行できます。静的 **Spike** は、操作が簡単でパフォーマンスの向上度が大きく、実用的です。

静的 **Spike** を使用するには、次の手順を実行します。

1. データベースをシャットダウンします。
2. 次のコマンドを入力して、**oracle** イメージを **Spike** します。

```
$ spike oracle -o oracle.spike -symbols_live
```

3. 次のコマンドを入力して、元のイメージを保存し、Spike されたイメージへのシンボリック・リンクを作成します。

```
$ mv oracle oracle.orig
$ ln -s oracle.spike oracle
```

4. データベースを起動します。

注意： オラクル社カスタマ・サポート・センターに連絡するときは、元のイメージを使用して問題を再現してください。

フィードバック付き Spike の実行

フィードバック付き Spike では、静的 Spike で実行できるすべての最適化以外に、ホット / コールド基本ブロック移動などの作業負荷に関連した最適化も実行されます。テスト環境では、Spike によるパフォーマンス最適化の約半分が、フィードバック情報に依存する最適化で実現されました。フィードバック付き Spike では、静的 Spike に比べて多数の手順と多くの操作が必要です。しかし、パフォーマンスを重視する場合は、工数は余分に必要ですが、静的 Spike のみでは得られない効果を期待できます。

フィードバック付き Spike を使用するには、次の手順を実行します。

1. 次のコマンドを入力して、Oracle バイナリをインストールメント処理します。

```
$ pixie -output oracle.pixie -dirname dir -pids oracle_image
```

この例の `oracle_image` は元のイメージです。`dir` は、インストールメント処理された実行可能プログラムがプロファイルしているデータファイルを書き込むディレクトリの名前です。

注意： `-dirname` オプションを指定すると、`oracle.Counts.pid` ファイルが `dir` ディレクトリに保存されます。これらのファイルは容量が大きく、作業負荷に応じてファイル数も多くなるため、ディレクトリに十分なディスク領域が確保されていることを確認してください。

この手順では、後で必要になる `oracle.Addrs` ファイルも作成されます。

`pixie` コマンドの出力には、エラーが含まれている場合があります。これらのエラーは無視してもかまいません。

2. データベースをシャットダウンします。
3. 次のコマンドを入力して、元のイメージを保存し、`pixie` イメージへのシンボリック・リンクを作成します。

```
$ mv oracle oracle.orig
$ ln -s oracle.pixie oracle
```

4. データベースを起動し、対応する作業負荷を実行します。

`pixie` 実行可能プログラムは容量が大きく処理速度が遅いため、標準の実行可能プログラムを使用した場合と同じ数のユーザーは実行できません。Oracle Database を使用しているときは、複数の `oracle.Counts.pid` ファイルが作成されます。`pid` は、対応する Oracle プロセスのプロセス ID です。最適化する各 Oracle プロセスのプロセス ID を追跡します。これらは、クライアントの Oracle シャドウ・プロセスになります。

5. データベースをシャットダウンします。
6. 次のコマンドを入力し、シンボリック・リンクを作成して、元の実行可能ファイルと置換します。

```
$ ln -s oracle.orig oracle
```

7. 特定の `oracle.Counts.pid` ファイルを代表的な作業負荷として選択できる場合は、手順 **a** を実行します。複数のカウント・ファイルをマージして作業負荷を表現する場合は、手順 **b** を実行します。

- a. `pixie` コマンドによって作成された `oracle.Addrs` ファイル、`oracle.Counts.pid` ファイル、および元の Oracle 実行可能ファイルが使用可能であることを確認します。

プロセス ID (pid) を使用して代表的な `oracle.Counts.pid` ファイルを選択し、次のコマンドを入力してコピーします。

```
$ cp oracle.Counts.pid oracle.Counts
```

- b. `prof` ユーティリティを使用して、複数の `oracle.Counts.pid` ファイルをマージします。このユーティリティの詳細は、`prof` の `man` ページを参照してください。

パラレル問合せオプションを使用している場合は、問合せスレーブと問合せコーディネータによって生成された `oracle.Counts.pid` ファイルをマージします。マージされたファイルは、問合せ開始クライアントの Oracle シャドウ・プロセスとなります。

パラレル問合せオプションを使用していない場合は、各 Oracle フォアグラウンド・プロセスからメモリー使用量の最も多い `oracle.Counts.pid` ファイルを選択してマージします。

`oracle.Counts.pid` ファイルをマージするには、次のコマンドを入力します。

```
$ prof -pixie -merge oracle.Counts $ORACLE_HOME/bin/oracle \
oracle.Addrs oracle.Counts.pid1 oracle.Counts.pid2
```

8. `oracle.Addrs` および `oracle.Counts` ファイルがカレント・ディレクトリで使用可能になっていることを確認し、フィードバック情報を使用して `Spike` を実行します。次のコマンドを入力します。

```
$ spike oracle -fb oracle -o oracle.spike_fb -symbols_live
```

`spike` コマンドの出力には、エラーが含まれている場合があります。これらのエラーは無視してもかまいません。

9. 次のコマンドを入力し、新しい `oracle` イメージへのシンボリック・リンクを作成します。

```
$ ln -s oracle.spike_fb oracle
```

10. データベースを起動します。

Solaris システムでの Oracle Database の管理

この付録では、Solaris システムで Oracle Database を管理する方法について説明します。

緊密共有メモリー

Solaris システム上の Oracle Database では、Oracle プロセス間で仮想メモリー・リソースを共有するため、共有メモリー・セグメントとして緊密共有メモリー (ISM) が使用されます。ISM を使用すると、共有メモリー・セグメント全体の物理メモリーが自動的にロックされます。

Solaris 8 および Solaris 9 システムでは、ページング可能な Dynamic Intimate Shared Memory (DISM) を使用できます。DISM を使用すると、Oracle Database はセグメントを共有するプロセス間で仮想メモリー・リソースを共有でき、同時にメモリーのページングも可能になります。このため、オペレーティング・システムでは、共有メモリー・セグメント全体で使用する物理メモリーをロックする必要がありません。

Oracle Database では、次の基準に基づいて、ISM または DISM のいずれを使用するかが起動時に自動的に判断されます。

- Oracle Database では、使用しているシステムで DISM を使用でき、SGA_MAX_SIZE 初期化パラメータの値が、結合されたすべての SGA コンポーネントに必要なサイズよりも大きい場合に、DISM が使用されます。したがって、Oracle Database では、使用される物理メモリー量のみがロックされます。
- Oracle Database では、起動時に共有メモリー・セグメント全体が使用中の場合、または SGA_MAX_SIZE パラメータの値が、結合されたすべての SGA コンポーネントに必要なサイズ以下の場合に、ISM が使用されます。

Oracle Database では、ISM または DISM のいずれを使用するかに関係なく、インスタンスの起動後に、常に動的にサイズ変更できるコンポーネント (バッファ・キャッシュなど)、共有プールおよびラジ・プール間でメモリーを交換できます。Oracle Database では、動的 SGA コンポーネントからメモリーを解放し、それを別のコンポーネントに割り当てることができます。

DISM の使用時は、共有メモリー・セグメントがメモリー内で暗黙的にロックされないため、Oracle Database は起動時に使用している共有メモリーを明示的にロックします。動的な SGA 操作によって共有メモリーが追加されると、Oracle Database は使用中のメモリーを明示的にロックします。動的な SGA 操作によって共有メモリーが解放されると、Oracle Database は解放されたメモリーに対するロックを明示的に解除します。その結果、解放されたメモリーが他のアプリケーションで使用できるようになります。

Oracle Database は、oradism ユーティリティを使用して共有メモリーのロックとロック解除を実行します。oradism ユーティリティは、インストール時に自動的に設定されます。したがって、動的 SGA を使用するための設定作業は不要です。

注意： Solaris システムでは、サーバー・パラメータ・ファイルの LOCK_SGA パラメータを true に設定しないことをお勧めします。true に設定した場合、Oracle Database 10g は起動しません。

注意： oradism ユーティリティのプロセス名は、ora_dism_sid です。sid はシステム識別子です。DISM を使用している場合、このプロセスはインスタンスの起動時に開始され、インスタンスがシャットダウンされると自動的に終了します。

oradism ユーティリティが正しく設定されていないことを示すメッセージがアラート・ログに表示された場合は、oradism ユーティリティが \$ORACLE_HOME/bin ディレクトリに格納されていること、およびスーパーユーザー権限が付与されていることを確認してください。

データベースの制限

この付録では、UNIX システム固有のデータベースの制限について説明します。

データベースの制限

表 G-1 に、CREATE DATABASE 文または CREATE CONTROLFILE 文のパラメータのデフォルト値および最大値を示します。

注意： これらのパラメータ間の相互依存によって、許容値に影響を与える場合があります。

表 G-1 CREATE CONTROLFILE および CREATE DATABASE のパラメータ

パラメータ	デフォルト	最大値
MAXLOGFILES	16	255
MAXLOGMEMBERS	2	5
MAXLOGHISTORY	100	65534
MAXDATAFILES	30	65534
MAXINSTANCES	1	63

表 G-2 に、UNIX 固有の Oracle Database ファイル・サイズ制限をバイト単位で示します。

表 G-2 ファイル・サイズの制限

ファイル・タイプ	オペレーティング・システム	最大サイズ
データファイル	すべて	4,194,303 と DB_BLOCK_SIZE パラメータの値の積
インポート / エクスポート・ファイル および SQL*Loader ファイル	Tru64 UNIX	16TB
	AIX、HP-UX、Linux、Solaris: 32 ビット (32 ビットのファイルを扱う)	2,147,483,647 バイト
	AIX、HP-UX、Linux、Solaris: 32 ビット (64 ビットのファイルを扱う)	無制限
	AIX、HP-UX、Linux、Mac OS X、 Solaris: 64 ビット	無制限
制御ファイル	Solaris、HP-UX、Linux、Mac OS X	20000 データベース・ブロック
	AIX	10000 データベース・ブロック
	Tru64 UNIX	19200 データベース・ブロック

数字

32 ビットおよび 64 ビットのクライアント・アプリケーションのサポート, 6-5

A

adapters ユーティリティ, 5-3
aio_task_max_num サブシステム属性
 Tru64 UNIX, E-7
AIX での Oracle Database によるミラー復元, A-13
AIX での共有メモリー, 8-13
AIX でのブロック・サイズ, A-4
AIX のツール, 8-5
 PTX Agent, 8-5
 PTX Manager, 8-5
 SMIT, 8-5
AIX 論理ボリューム・マネージャ, A-5
ASM_DISKSTRING
 オペレーティング・システムのデフォルト値, 1-8
Automatic Storage Management プロセス
 再起動, 2-3
 停止, 2-2

B

Base Operation System ツール, 8-5
Bourne シェル・スクリプトのトレース, 1-3

C

C
 Pro*C/C++, 6-6
CLUSTER_INTERCONNECTS
 初期化パラメータ, 1-8
coraenv, 1-3
coraenv ファイル
 説明, 1-6
CPU_COUNT 初期化パラメータ
 HP-UX 動的プロセッサ再構成, B-10
CPU のスケジューリングおよびプロセスの優先順位
 AIX, A-14
CREATE CONTROLFILE のパラメータ, G-2
CREATE DATABASE のパラメータ, G-2
CURSOR_SPACE_FOR_TIME 初期化パラメータ
 HP-UX システムでの Oracle Database のチューニング, B-7

D

Database Configuration Assistant
 使用, 3-3
Database Control
 起動, 2-6
 停止, 2-6
Database Upgrade Assistant
 使用, 3-3
DB_BLOCK_SIZE
 最大値, 1-7
DB_BLOCK_SIZE 初期化パラメータ, 8-12
DB_CACHE_SIZE 初期化パラメータ, 8-12
DB_FILE_MULTIBLOCK_READ_COUNT パラメータ
 AIX, A-11
DBA, 「管理者」を参照
dba グループ, 1-10
dbhome ファイル, 1-7
DISM, F-2

E

/etc/privgroup ファイル, B-3, B-4

F

FORMAT プリコンパイラ, 6-12
Pro*COBOL, 6-13

G

getprivgrp コマンド, B-3, B-4
Glance/UX, 8-7
glogin.sql ファイル, 4-2
GPFS
 使用する際の考慮事項, A-7

H

HMP パラメータのチューニング
 HP-UX, B-9
HP-UX SCHED_NOAGE スケジューリング・ポリシー
 説明, B-2
HP-UX システムでの Oracle Database のチューニング
 永続的な専用 SQL 領域とメモリー, B-7
 大規模メモリーの割当て, B-7
HP-UX での Oracle Database のチューニング
 チューニングに関する推奨事項, B-9

HP-UX 動的プロセッサ再構成
CPU_COUNT 初期化パラメータ, B-10
HP-UX のツール, 8-6, 8-7
HP-UX パフォーマンス・チューニング・ツール, 8-6
HP-UX パフォーマンス分析ツール, 8-7
HP-UX 非同期ドライバ
Oracle Database に対して設定されているかどうかの
検証, B-6
HP の非同期フラグ, B-7

I

@ 略称, 1-2
I/O
AIX での非同期, A-9
HP での非同期, B-4
Tru64 での非同期, E-7
チューニング, 8-10
iostat, 8-4
I/O スレーブ, A-10
I/O バッファおよび SQL*Loader
AIX, A-5
IPC プロトコル, 5-5
IRECLen, 6-3
ISM, F-2
iSQL*Plus
起動, 2-5
停止, 2-5

J

JAVA_POOL_SIZE 初期化パラメータ, 8-12
JFS および JFS2
考慮事項, A-7

L

LARGE_POOL_SIZE 初期化パラメータ, 8-12
LD_OPTION 環境変数, 1-5
Linux のツール, 8-7
LOG_ARCHIVE_DEST 初期化パラメータ
ASYNC の最大値, 1-8
LOG_BUFFERS 初期化パラメータ, 8-12
login.sql ファイル, 4-2
lsps コマンド, 8-4

M

max_async_req パラメータ, E-6
max_objs パラメータ, E-6
max_sessions パラメータ, E-6
MAXDATAFILES パラメータ, G-2
maxfree パラメータ, A-2
MAXINSTANCES パラメータ, G-2
MAXLOGFILES パラメータ, G-2
MAXLOGHISTORY パラメータ, G-2
MAXLOGMEMBERS パラメータ, G-2
MAXPERM パラメータ
AIX でのチューニング, A-3
maxperm パラメータ, A-2, A-3
MicroFocus COBOL コンパイラ, 6-10
minfree パラメータ, A-2
MINPERM, A-3

minperm パラメータ, A-2, A-3
MLOCK 権限, B-4
HP-UX, B-4
mpstat コマンド, 8-7
msg_size パラメータ, E-6

N

NUMA
「指定配置最適化」を参照

O

oinstall グループ, 1-10
oper グループ, 1-10
Oracle Advanced Security, 5-6
Oracle C++ Call Interface, 6-16
Oracle Call Interface
ユーザー・プログラム, 6-17
Oracle Call Interface と Oracle C++ Call Interface, 6-16
デモ・プログラム, 6-16
Oracle Data Migration Assistant
データベースのアップグレード, 3-3
Oracle Database Configuration Assistant
説明, 3-3
Oracle Database の SCHED_NOAGE
Oracle Database での有効化, B-3
Oracle Database の環境変数
Oracle Database の変数, 1-2
Oracle Database のチューニングと大規模メモリーの
割当て
HP-UX システム上, B-7
Oracle Database プロセス
再起動, 2-3
停止, 2-2
Oracle Management Agent
起動, 2-7
停止, 2-7
Oracle Net Configuration Assistant
使用, 3-2
説明, 3-2
Oracle Net Listener
再起動, 2-4
停止, 2-4
Oracle Net Services
IPC プロトコル, 5-5
Oracle Advanced Security, 5-6
protocol support, 5-4
TCP/IP プロトコル, 5-6
プロトコル, 5-4
Oracle Net Services 構成ファイル, 5-2
Oracle Net 構成ファイル, 5-2
Oracle Real Application Clusters
HP-UX でのチューニング, B-9
Oracle Real Application Clusters と NUMA の同時
有効化
Tru64 UNIX, E-7
Oracle Real Application Clusters とフォルト・トレラント
IPC
AIX, A-16
Oracle Real Application Clusters の要件
Tru64 UNIX でのダイレクト I/O とコンカレント I/O
のサポート, E-9

oracle アカウント, 1-11
Oracle インスタンス用の共有メモリー・セグメント
 HP-UX, B-2
Oracle 環境変数
 ORA_NLS10, 1-2
 ORA_TZFILE, 1-3
 ORACLE_BASE, 1-3
 ORACLE_HOME, 1-3
 ORACLE_PATH, 1-3
 ORACLE_SID, 1-3
 ORACLE_TRACE, 1-3
 ORAENV_ASK, 1-3
 SQLPATH, 1-3, 1-4
 TWO_TASK, 1-4
Oracle システム識別子, 1-3
Oracle 初期化パラメータ
 Tru64 UNIX, E-3
Oracle 製品
 追加に対するデータベースの構成, 3-2
Oracle プリコンパイラと OCI のリンクおよび Make
 ファイル
 カスタム Make ファイル, 6-19
 未定義シンボル, 6-19
Oracle ブロック・サイズ
 調整, 8-9
Oracle ユーザー
 アカウントの構成, 1-12
Oracle ランタイム・システム
 Pro*COBOL, 6-11
oradism コマンド, F-2
ORAENV_ASK, 1-3
oraenv ファイル
 説明, 1-6
ORAINVENTORY, 1-10
orapwd コマンド, 1-11
orapwd ユーティリティ, 1-11
ORECLEN, 6-3
OS_AUTHENT_PREFIX パラメータ, 1-11
OSDBA, 1-10
OSDBA および OSOPER グループ, 1-12
OSOPER, 1-10

P

PL/SQL カーネル・デモ, 7-2
PL/SQL のデモ, 7-2
Polycenter Advanced ファイル・システム, 8-10
privgroup ファイル, B-3, B-4
Pro*C/C++
 Make ファイル, 6-6
 シグナル, 6-20
 使用, 6-6
 デモ・プログラム, 6-6
 ユーザー・プログラム, 6-7
Pro*COBOL, 6-8
 FORMAT プリコンパイラ, 6-12, 6-13
 Oracle ランタイム・システム, 6-11
 環境変数, 6-9
 デモ・プログラム, 6-11
 ネーミングの相違点, 6-8
 ユーザー・プログラム, 6-12
PRODUCT_USER_PROFILE 表, 4-2

Programmer's Analysis Kit (HP PAK), 8-7
PTX Agent, 8-5

R

rad_gh_regions パラメータ, E-3
RAD に対するプロセスの親和性
 Tru64 UNIX, E-4
RAW デバイス, 1-13
 AIX でのバックアップ, A-13
 Oracle Real Application Clusters のインストール,
 1-13
 RAW ディスク・パーティションの可用性, 1-13
 Tru64, E-11
 使用方法のガイドライン, 1-13
 設定, 1-13
 バッファ・キャッシュ・サイズ, 8-14
RAW デバイスの設定, 1-14
RAW 論理ボリュームからジャーナル・ファイル・シス
 テムへの移動, A-8
rdg_max_auto_msg_wires パラメータ, E-6
Red Hat Enterprise Linux AS 2.1 および SUSE Linux
 Enterprise Server 8 でのラージ・ページの有効化,
 C-3
Red Hat Enterprise Linux AS 2.1 での hugetlbfs の使用,
 C-4
Red Hat Enterprise Linux AS 3 での hugetlbfs の使用,
 C-5
Red Hat Enterprise Linux の場合のみ
 VLM ウィンドウ・サイズおよびプロセスごとの最大
 ロック・メモリーのデフォルト・サイズ, C-3
Reliable Data Gram, E-6
 Tru64 UNIX, E-6
root.sh スクリプト, 1-7

S

sar, 8-3
sar コマンド, 8-9
SCHED_NOAGE パラメータ, B-2
semtimedop のサポート
 Linux 上, C-8
setprivgrp コマンド, B-3, B-4
SGA, 8-12
 確認, 8-13
SGA_MAX_SIZE パラメータ, F-2
SGA アドレス空間の増加, C-6
SGA の非同期フラグ
 HP-UX システム上, B-7
SHARED_POOL_SIZE 初期化パラメータ, 8-12
shm_allocate_striped パラメータ, E-3
shm_max パラメータ, 8-12
shmmax パラメータ, 8-12
shm_seg パラメータ, 8-12
shmseg パラメータ, 8-12
SIGCLD 2 タスク・シグナル, 6-20
SIGINT 2 タスク・シグナル, 6-20
SIGIO 2 タスク・シグナル, 6-20
SIGPIPE 2 タスク・シグナル, 6-20
SIGTERM 2 タスク・シグナル, 6-20
SIGURG 2 タスク・シグナル, 6-20
SMP システムでのプロセッサ・バインディング
 AIX での使用, A-14

Solaris のツール, 8-7
Spike 最適化ツール, E-13
 Tru64 UNIX, E-13
Spike の使用
 Tru64 UNIX, E-13
SPOOL コマンド
 SQL*Plus, 4-5
 使用, 4-5
SQL*Loader, A-5
SQL*Loader のデモ, 7-2
SQL*Module for Ada
 使用, 6-15
 ユーザー・プログラム, 6-16
SQL*Module for Ada (AIX の場合のみ), 6-15
SQL*Plus
 PRODUCT_USER_PROFILE 表, 4-2
 SPOOL コマンド, 4-5
 エディタ, 4-4
 オペレーティング・システム・コマンドの実行, 4-4
 コマンドライン SQL*Plus の使用, 4-4
 コマンドライン・ヘルプ, 4-3
 サイト・プロファイル, 4-2
 システム・エディタ, 4-4
 制限事項, 4-5
 設定ファイル, 4-2
 デフォルト・エディタ, 4-4
 デモンストレーション表, 4-3
 ユーザー・プロファイル, 4-2
 割込み, 4-4
SQL*Plus のコマンドライン・ヘルプ
 インストール, 4-3
SQL*Plus のコマンドライン・ヘルプのインストール,
 4-3
SQL*Plus のコマンドライン・ヘルプの削除, 4-3
SQL*Plus への割込み, 4-4
ssm_threshold パラメータ, E-3
swapinfo コマンド, 8-4
swapon コマンド, 8-4
swap コマンド, 8-4
SYSDATE, 1-7
SYSTEM, 1-12
SYS アカウント, 1-12

T

TCP/IP プロトコル, 5-6
TCP/IP プロトコル・サポート, 5-5
TIMED_STATISTICS パラメータ, E-10
Tru64 UNIX サブシステム属性
 説明, E-3
Tru64 UNIX でのダイレクト I/O サポートの無効化, E-9

U

UDP IPC, E-6
UDP のチューニング, A-14
Ultra Search
 起動, 2-6
 停止, 2-5
unified ファイル・システム, 8-10
UNIX System V ファイル・システム, 8-10

UNIX 環境変数
 CLASSPATH, 1-4
 DISPLAY, 1-4
 DYLD_LIBRARY_PATH, 1-4
 HOME, 1-5
 LANG, 1-5
 LANGUAGE, 1-5
 LD_LIBRARY_PATH, 1-5
 LD_OPTIONS, 1-5
 LIBPATH, 1-5
 LPDEST, 1-5
 PATH, 1-5
 PRINTER, 1-5
 SHLIB_PATH, 1-5
 TMPDIR, 1-6
 XENVIRONMENT, 1-6
UNIX グループ
 dba, 1-10
 oinstall, 1-10
 oper, 1-10
UNIX コマンド
 getprivgrp, B-3, B-4
 setprivgrp, B-3, B-4
USE_INDIRECT_DATA_BUFFERS パラメータ
 Linux 上, C-2
UTLRP.SQL
 無効な SQL モジュールの再コンパイル, 3-3

V

Veritas ファイル・システム, 8-10
vm_stat コマンド, 8-9
vmstat, 8-2
vmstat コマンド, 8-9

X

XA 機能, 6-22

あ

アカウント
 SYS, 1-12
 SYSTEM, 1-12
アップグレード済データベース
 構成, 3-3

い

移行, 3-3
インストール
 Oracle Internet Directory, xi
 Oracle Workflow, xi
インストール後の作業
 Configuration Assistant, 3-2
インポート・ユーティリティ用の BUFFER パラメータ
 AIX, A-5

う

受取りルーチン, 6-21
 例, 6-21

お

オペレーティング・システム・コマンドの実行, 4-4
オペレーティング・システムのツール, 8-2

か

拡張ファイル・システム, 8-10
拡張バッファ・キャッシュのサポート
 Linux 上, C-2
拡張バッファ・キャッシュの制限事項
 Linux 上, C-2
仮想メモリー・データ・ページ
 HP-UX での Oracle Database のチューニング, B-7
仮想メモリー・ページ・サイズ
 HP-UX での Oracle Database のチューニング, B-8
環境変数, 6-10
 MicroFocus COBOL コンパイラ, 6-10
 Pro*COBOL, 6-9
 TNS_ADMIN, 5-2
環境変数 ADA_PATH, 1-4
環境変数 CLASSPATH, 1-4
環境変数 DISPLAY, 1-4
環境変数 DYLD_LIBRARY_PATH, 1-4
環境変数 HOME, 1-5
環境変数 LANG, 1-5
環境変数 LANGUAGE, 1-5
環境変数 LD_LIBRARY_PATH, 1-5
環境変数 LIBPATH, 1-5
環境変数 LPDEST, 1-5
環境変数 ORA_NLS10, 1-2
環境変数 ORA_TZFILE, 1-3
環境変数 ORACLE_BASE, 1-3
環境変数 ORACLE_HOME, 1-3
環境変数 ORACLE_PATH, 1-3
環境変数 ORACLE_SID, 1-2, 1-3
環境変数 PATH, 1-5
環境変数 PRINTER, 1-5
環境変数 SHLIB_PATH, 1-5
環境変数 SQLPATH, 1-3
環境変数 TMPDIR, 1-6
環境変数 TWO_TASK, 1-4
環境変数 XENVIRONMENT, 1-6
管理者
 UNIX アカウント, 1-12
関連ドキュメント, xi

き

キャッシュ・サイズ, 8-14
共通の環境
 設定, 1-6

く

クライアント共有ライブラリ, 6-4
クライアント静的ライブラリ, 6-4
クラスタ・ファイル・システム, 8-10
グループ
 OSDBA および OSOPER, 1-12

け

軽量タイマー, B-3
軽量タイマーの実装
 HP-UX, B-3
言語, 1-2

こ

構成
 Oracle ユーザーのアカウント, 1-12
構成ファイル
 Oracle Net, 5-2
 Oracle Net Services, 5-2
 プリコンパイラ, 6-2
高速ネットワークのサポート
 Linux 上, C-8
コマンドライン SQL*Plus の使用, 4-4
コマンドライン SQL の管理, 4-2

し

シグナル・ハンドラ
 使用, 6-20
シグナル・ルーチン, 6-21
 例, 6-21
システム・エディタ
 SQL*Plus, 4-4
システム時刻, 1-7
システムに存在する RAD 数のサブセットへの Oracle
 Database の実行制限
 Tru64 UNIX, E-4
実行可能ファイルの再リンク, 3-4
指定配置最適化, E-2
 Tru64 UNIX で実行するための要件, E-2
 Tru64 UNIX での使用, E-3
 Tru64 UNIX での無効化, E-3
 Tru64 UNIX での有効化, E-2
ジャーナル・ファイル・システム, 8-10
ジャーナル・ファイル・システムから RAW 論理ボ
 リュームへの移動, A-8
シャドウ・プロセス, 1-10
順次先読み
 AIX でのチューニング, A-11
初期化パラメータ, 1-7
BACKGROUND_DUMP_DEST, 1-15
DB_BLOCK_SIZE, 8-12
DB_CACHE_SIZE, 8-12
JAVA_POOL_SIZE, 8-12
LARGE_POOL_SIZE, 8-12
LOG_ARCHIVE_DEST の ASYNC の最大値, 1-8
LOG_BUFFERS, 8-12
MAX_DUMP_FILE, 1-15
SHARED_POOL_SIZE, 8-12
USER_DUMP_DEST, 1-15

す

ストライプ化された論理ボリューム
 AIX での設計, A-6
スレッドのサポート, 6-20
スワップ領域, 8-8
 チューニング, 8-8

せ

制限事項 (SQL*Plus), 4-5
 ウィンドウのサイズ変更, 4-5
 リターン・コード, 4-5
静的および動的リンク
 Oracle ライブラリとプリコンパイラ, 6-3
セキュリティ, 1-10
 2 タスク構造, 1-10
 UNIX の機能, 1-10
 グループ・アカウント, 1-10
 ファイル所有権, 1-10
設定ファイル
 SQL*Plus, 4-2

そ

ソフトウェアの配布, 1-3

た

ダイレクト I/O サポート
 Linux 上, C-8
ダイレクト I/O サポートおよびコンカレント・ダイレク
 ト I/O サポート
 Tru64 UNIX, E-8

ち

チューニング, 8-8
 AIX での順次先読み, A-11
 HP-UX での Oracle Real Application Clusters, B-9
 I/O ボトルネック, 8-10
 UDP, A-14
 アーカイブ・バッファ, A-5
 ディスク I/O, 8-10
 トレースおよびアラート・ファイル, 1-15, 8-14
 メモリー管理, 8-8
チューニング・ツール
 Glance/UX, 8-7
 lsps, 8-4
 mpstat, 8-7
 Programmer's Analysis Kit (HP PAK), 8-7
 PTX Agent, 8-5
 PTX Manager, 8-5
 swap, 8-4
 swapinfo, 8-4
 swapon, 8-4
チューニングに関する推奨事項
 HP-UX での Oracle Database, B-9

て

ディスク
 パフォーマンスの監視, 8-11
ディスク I/O
 I/O スレーブ, A-10
 チューニング, 8-10
 ファイル・システム・タイプ, 8-10
ディスク I/O の歩調合せ
 AIX でのチューニング, A-12

データベース

 AIX でのブロック・サイズ, A-4
 アップグレード, 3-3
 移行, 3-3
 作成方法
 Oracle Database Configuration Assistant, 3-3
データベース・ブロック・サイズ
 AIX での設定, A-4
デバッガ・プログラム, 6-3
デモ
 プリコンパイラ, 7-4
デモ・プログラム
 64 ビット・ドライバを使用する Oracle JDBC/OCI
 プログラムの実行, 6-18
 Oracle Call Interface, 6-16
 Pro*C/C++, 6-6
 Pro*COBOL, 6-11
 Pro*FORTRAN, 6-13
 SQL*Module for Ada, 6-15
デモンストレーション表
 SQL*Plus, 4-3

と

動的キャッシュ・パラメータ
 Linux 上, C-2
ドキュメント
 以前のリリースからの移行とアップグレード, xi
 管理とチューニング, xi
 関連, xi
特殊アカウント, 1-10
トレースおよびアラート・ファイル
 アラート・ファイル, 1-15
 使用, 1-15, 8-14
 トレース・ファイル名, 1-15

は

バッファ・キャッシュ
 チューニング, A-2
バッファ・キャッシュ・サイズ, 8-14
 チューニング, 8-14
バッファ・キャッシュのページング・アクティビティ
 AIX での制御, A-2
バッファ・マネージャ, 8-8
パラメータ
 CREATE CONTROLFILE, G-2
 CREATE DATABASE, G-2
 max_async_req, E-6
 max_objs, E-6
 max_sessions, E-6
 MAXDATAFILES, G-2
 maxfree, A-2
 MAXLOGFILES, G-2
 MAXLOGHISTORY, G-2
 MAXLOGMEMBERS, G-2
 maxperm, A-2, A-3
 minfree, A-2
 minperm, A-2, A-3
 msg_size, E-6
 OS_AUTHENT_PREFIX, 1-11
 rad_gh_regions, E-3
 rdg_max_auto_msg_wires, E-6

- SCHED_NOAGE, B-2
- SGA_MAX_SIZE, F-2
- shm_allocate_stripped, E-3
- shm_max, 8-12
- shmmax, 8-12
- shm_seg, 8-12
- shmseg, 8-12
- TIMED_STATISTICS, E-10

ひ

- 非同期 I/O
 - HP-UX, B-4
 - HP-UX での検証, B-6
 - Linux 上, C-7
- 非同期 I/O の実装
 - HP-UX, B-4

ふ

- ファイル
 - dbhome, 1-7
 - /etc/privgroup, B-3, B-4
 - root.sh, 1-7
 - トレース・ファイル, 1-15
- ファイル・システム, 8-10
 - AdvFS, 8-10
 - AIX のオプション, A-7
 - CFS, 8-10
 - ext2/ext3, 8-10
 - GPFS, 8-10
 - JFS, 8-10
 - OCFS, 8-10
 - S5, 8-10
 - UFS, 8-10
 - VxFS, 8-10
- ファイル・バッファ・キャッシュ
 - AIX のチューニング, A-2
- 複数のシグナル・ハンドラ, 6-20
- プリコンパイラ
 - IRECLEN および ORECLEN の値, 6-3
 - 大文字から小文字への変換, 6-3
 - 概要, 6-2
 - シグナル, 6-20
 - デモの実行, 7-4
 - ベンダー提供のデバッグ・プログラム, 6-3
- プリコンパイラ構成ファイル, 6-2
- プリコンパイラ実行可能ファイル
 - 再リンク, 6-2
- プリコンパイラの README ファイル, 6-3
- プロセスごとの最大ロック・メモリー
 - VLM ウィンドウ・サイズおよび Red Hat Enterprise Linux 3, C-3
- プロトコル, 5-4

へ

- ページアウト・アクティビティ, 8-9
- ページング, 8-9
 - 過剰な, A-2, A-4
 - 不十分, A-3
- ページングの制御
 - AIX, A-4

- ページング領域, 8-8
 - AIX での十分な割当て, A-3
 - チューニング, 8-8, 8-9

め

- メモリー
 - 競合, A-2
 - チューニング, 8-8
- メモリー管理, 8-8
 - スワップ領域, 8-8
 - ページングの制御, 8-9
- メモリーとページング
 - AIX, A-2
- メモリー内ファイル・システム
 - Linux で拡張バッファ・キャッシュをサポートしている場合, C-2

ゆ

- ユーザー・プログラム
 - Oracle Call Interface と Oracle C++ Call Interface, 6-17
 - Pro*C/C++, 6-7
 - Pro*COBOL, 6-12
 - SQL*Module for Ada, 6-16
- ユーザー・プロファイル
 - SQL*Plus, 4-2
- ユーザー割込みハンドラ, 6-21

ら

- ライブラリ
 - クライアント共有ライブラリとクライアント静的ライブラリ, 6-4

り

- リアルタイム・クロック, E-10
- リスナー
 - TCP/IP または SSL 付き TCP/IP 用の設定, 5-6

