

Oracle8

for Intel UNIX (DG/UX Intel, NCR MP-RAS, SCO UnixWare, Solaris Intel)
管理者リファレンス

リリース 8.0.6

1999 年 9 月

部品番号 : J00642-01

おもなトピック :

[Oracle8 での最適フレキシブル・アーキテクチャ \(OFA \)](#)

[Intel UNIX での Oracle8 の管理](#)

[Intel UNIX での Oracle8 のチューニング](#)

[Intel UNIX での SQL*Plus の管理](#)

[Intel UNIX での Oracle プリコンパイラおよび Oracle コール・インタフェースの使用](#)

[Oracle Net8 の構成](#)

[Intel UNIX での Oracle データ・カートリッジ・デモの実行](#)

Oracle8 for Intel UNIX (DG/UX Intel, NCR MP-RAS, SCO UnixWare, Solaris Intel) 管理者リファレンス
リリース 8.0.6

部品番号 : J00642-01

原本名 : Oracle8 Administrator's Reference, Release 8.0.6 for Intel UNIX (DG/UX Intel, NCR MP-RAS,
SCO UnixWare, Solaris Intel)

原本部品番号 : A75318-01

原本著者 : Kevin Adams

原本協力者 : Nicholas Hind, Lynn Robinson, Zeynep Taspinar, Cynthia Chin-Lee, Sharad Lal,
Pauline Hoi, Binoy Sukumaren

Copyright © 1996, 1999 Oracle Corporation. All rights reserved.

Printed in Japan.

制限付権利の説明

プログラムの使用、複製または開示は、オラクル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に関する法律により保護されています。

当ソフトウェア (プログラム) のリバース・エンジニアリング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ことがあります。オラクル社は本ドキュメントの無
謬性を保証しません。

* オラクル社とは、Oracle Corporation (米国オラクル) または日本オラクル株式会社 (日本オラクル)
を指します。

危険な用途への使用について

オラクル社製品は、原子力、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
を用途として開発されておりません。オラクル社製品を上述のようなアプリケーションに使用すること
についての安全確保は、顧客各位の責任と費用により行ってください。万一かかる用途での使用により
クレームや損害が発生いたしましても、日本オラクル株式会社と開発元である Oracle Corporation (米
国オラクル) およびその関連会社は一切責任を負いかねます。当プログラムを米国国防総省の米国政府
機関に提供する際には、『Restricted Rights』と共に提供してください。この場合次の Legend が適用さ
れます。

Restricted Rights Legend

Programs delivered subject to the DOD FAR Supplement are "commercial computer software" and use,
duplication and disclosure of the Programs shall be subject to the licensing restrictions set forth in the
applicable Oracle license agreement. Otherwise, Programs delivered subject to the Federal Acquisition
Regulations are "restricted computer software" and use, duplication and disclosure of the Programs shall
be subject to the restrictions in FAR 52.227-14, Rights in Data -- General, including Alternate III (June
1987). Oracle Corporation, 500 Oracle Parkway, Redwood City, CA 94065.

このドキュメントに記載されているその他の会社名および製品名は、あくまでその製品および会社を識
別する目的にのみ使用されており、それぞれの所有者の商標または登録商標です。

目次

はじめに	ix
1 Oracle8 での最適フレキシブル・アーキテクチャ (OFA)	
最適フレキシブル・アーキテクチャ (OFA)	1-2
OFA に準拠したデータベースの特長	1-2
Oracle8 for UNIX で適用される OFA	1-4
マウント・ポイントの名前	1-5
ディレクトリの名前	1-5
ファイルの名前	1-7
表領域の名前	1-9
OFA 構造に基づいた Oracle ファイル	1-10
OFA ファイルのマッピング	1-10
RAW デバイスのサイズ	1-12
複数インスタンスにおける OFA に準拠したデータベースのファイル・マッピング	1-12
ディレクトリ構造	1-14
デフォルトの OFA データベース	1-18
2 Intel UNIX での Oracle8 の管理	
initstid.ora ファイルのカスタマイズ	2-2
サンプル initstid.ora ファイル	2-2
環境設定	2-5
環境変数の表示および設定	2-5
共通の環境設定	2-5
データベースの例	2-6

Oracle8 の環境変数	2-7
UNIX における Oracle 環境変数	2-8
Oracle8 で使用する UNIX 環境変数	2-11
システム時刻の設定	2-13
Oracle8 Server のメモリー使用量の見積り	2-13
クラスタ・サイズおよび索引サイズの計算	2-14
クラスタ・サイズの計算	2-14
索引サイズの計算	2-14
サーバー・リソースの制限	2-15
初期化パラメータ	2-15
デフォルトの初期化パラメータの値	2-15
システム・グローバル領域の制御	2-17
SGA のサイズ制限	2-17
SGA のサイズの計算	2-18
SGA の再配置	2-19
特殊アカウントおよび特殊グループの管理	2-20
セキュリティの管理	2-21
グループとセキュリティ	2-21
Oracle Server ユーティリティのセキュリティ	2-22
Server Manager コマンドのセキュリティ	2-22
データベース・ファイルのセキュリティ	2-23
ネットワーク・セキュリティ	2-23
Oracle Net8 での自動ログイン	2-24
確認の順番	2-26
セキュリティとリモート・パスワード	2-26
ログイン・ホーム・ディレクトリの管理	2-29
デモンストレーションの作成と実行	2-30
PL/SQL デモンストレーションのロード	2-30
PL/SQL デモンストレーションの実行	2-31
SQL*Loader のデモンストレーション	2-33
SQL*Loader の管理	2-33
Oracle Security Server	2-35
Oracle8 Server の SQL リファレンス	2-35
CREATE CONTROLFILE パラメータ	2-35
トレース・ファイルとアラート・ファイルの使用	2-36
トレース・ファイルの名前	2-36

アラート・ファイル	2-36
RAW デバイス	2-37
RAW デバイスの欠点 (NCR MP-RAS、SCO UnixWare および Solaris Intel の場合)	2-37
RAW デバイスを使用する場合のガイドライン	2-38
RAW デバイスの設定	2-39
動的共有メモリー (DSHM) の使用 (SCO UnixWare の場合)	2-42

3 Intel UNIX での Oracle8 のチューニング

チューニングの重要性	3-2
システムをチューニングする前に	3-2
パフォーマンス・ボトルネックのタイプ	3-2
システム・パフォーマンス監視用のツール	3-3
TKPROF および EXPLAIN PLAN ユーティリティ	3-3
SQL スクリプト utlbstat および utlestat	3-4
V\$SYSTEM_EVENT & V\$SESSION_WAIT 表	3-5
Intel UNIX のツール	3-6
vmstat (Solaris Intel の場合)	3-6
sar	3-7
iostat (Solaris Intel の場合)	3-9
swap (NCR MP-RAS、Solaris Intel および SCO UnixWare の場合)	3-10
admswap (DG/UX Intel の場合)	3-10
truss (NCR MP-RAS、Solaris Intel および SCO UnixWare の場合)	3-11
mpstat (Solaris Intel の場合)	3-12
Oracle8 Server のチューニング手順	3-12
アプリケーション設計のチューニング	3-13
参照ドキュメント	3-13
LONG データの個別保存	3-13
Large Object Base (LOB)	3-13
SQL 文の再使用	3-14
データ・アクセスのチューニング	3-14
メモリー管理のチューニング	3-14
十分なスワップ領域の割当て	3-15
ページングの制御	3-15
単一共有メモリー・セグメントでの SGA の確保	3-16
物理メモリーへの SGA のロック	3-17

データベース・バッファ数の最適化	3-17
間接データベースのバッファ (DG/UX Intel および SCO UnixWare の場合)	3-18
REDO バッファ数の最適化	3-18
共有プール・サイズの最適化	3-19
データ・ディクショナリ・キャッシュの有効性の検証	3-19
適切なライブラリ・キャッシュ領域の割当て	3-20
大きい PL/SQL ブロックの共有プールへのロック	3-20
セッション・キャッシュ・カーソルの最適化	3-21
ディスク I/O のチューニング	3-22
ディスク・パフォーマンスの監視	3-22
索引と表の分離	3-22
REDO ログの専用ディスク・デバイスへの配置	3-23
論理ボリュームの使用 (NCR MP-RAS および Solaris Intel の場合)	3-24
データベース・ライターをチューニングして書込みバンド幅を大きくする	3-24
大きなディスクへの要求キューの検索	3-26
他のディスクへのホット・ファイルの移動	3-26
ホット・ファイルへの I/O の低減	3-26
データベースの過度な断片化の確認	3-27
より多くのデータベースの使用	3-28
適切なファイル・システム・タイプの選択	3-28
RAW パーティションおよび RAW デバイス (I/O パウンドの場合)	3-28
ディスク I/O の最適化が失敗したとき	3-29
ブロック・サイズのチューニング	3-29
表のストライプ化	3-30
CPU の使用状況のチューニング	3-30
Oracle ユーザー / プロセスの優先順位をすべて同じにする	3-30
使用状況パターンの再編成	3-30
マルチ・プロセッサ・システムでのプロセッサ親和性 / バインディングの使用	3-30
クライアント / サーバー構成の使用	3-31
大量のエクスポート / インポートおよび SQL*Loader ジョブでの シングルタスク・リンクの使用	3-31
Oracle リソース競合のチューニング	3-31
V \$ 表を使用して競合を特定する方法	3-31
競合を起こすセグメントの特定	3-32
ラッチ解放競合の低減	3-34

ロールバック・セグメントの競合の低減	3-34
REDO ログ・バッファ・ラッチの競合の低減	3-35
パラレル問合せおよびパラレル DML の競合	3-35
マルチ・プロセッサ・システムでのスピン・カウントのチューニング	3-36
UNIX カーネル・パラメータのチューニング	3-37
Oracle Parallel Server (OPS) のリソース競合のチューニング (DG/UX Intel および NCR MP-RAS の場合)	3-38
CPU の競合	3-39
ロックの競合	3-40
ローカル・ディスク以外での I/O	3-42
Intel UNIX バッファ・キャッシュ・サイズのチューニング	3-42

4 Intel UNIX での SQL*Plus の管理

SQL*Plus の管理	4-2
設定ファイル	4-2
Site Profile	4-2
User Profile	4-2
PRODUCT_USER_PROFILE 表	4-3
デモンストレーション表	4-3
ヘルプ機能	4-4
SQL*Plus の使用	4-6
SQL*Plus からのシステム・エディタの使用	4-6
エディタの順序設定	4-6
_editor オプションの設定	4-6
環境変数の設定	4-7
デフォルト設定	4-7
SQL*Plus からのオペレーティング・システム・コマンドの実行	4-7
SQL*Plus への割込み	4-8
SPOOL コマンドの使用	4-8
制限事項	4-8
COPY コマンド	4-8
ウィンドウのサイズ変更	4-9
リターン・コード	4-9

5 Intel UNIX での Oracle プリコンパイラおよび Oracle コール・インタフェースの使用

Oracle プリコンパイラの概要	5-2
プリコンパイラ実行ファイルの再リンク	5-2
プリコンパイラ構成ファイル	5-3
すべてのプリコンパイラに共通の問題	5-3
参照ドキュメント	5-4
Pro*C/C++	5-4
Pro*C/C++ の管理	5-4
Pro*C/C++ の使用	5-4
Pro*COBOL (DG/UX Intel、SCO UnixWare および NCR MP-RAS の場合)	5-7
Pro*COBOL の管理	5-7
環境変数	5-8
Pro*COBOL の使用	5-8
Pro*FORTRAN (DG/UX Intel の場合)	5-11
Pro*FORTRAN の管理	5-11
Pro*FORTRAN の使用	5-11
Oracle コール・インタフェース	5-13
Oracle コール・インタフェースの使用	5-13
Oracle プリコンパイラ、Oracle コール・インタフェースのリンクおよび Make ファイル	5-16
カスタム Make ファイル	5-16
未定義シンボル	5-16
Oracle ライブラリを使用した静的リンクおよび動的リンク	5-17
シグナル・ハンドラの使用	5-18
シグナル	5-18
XA 機能	5-21

6 Oracle Net8 の構成

参照ドキュメント	6-2
README ファイルへの補足情報	6-2
主な Net8 製品および特徴	6-3
Net8 ファイルおよびユーティリティ	6-3
Oracle Connection Manager	6-4
マルチスレッド・サーバー	6-4
Oracle Names	6-5

Net8 Assistant	6-5
Oracle Net8 Protocol Adapter	6-6
BEQ Protocol Adapter	6-8
BEQ Protocol Adapter の概要	6-8
BEQ の ADDRESS 指定	6-8
IPC Protocol Adapter	6-10
IPC Protocol Adapter の概要	6-10
IPC の ADDRESS 指定	6-10
RAW Protocol Adapter	6-11
TCP/IP Protocol Adapter	6-12
TCP/IP Protocol Adapter の概要	6-12
TCP/IP の ADDRESS 指定	6-12
SPX/IPX Protocol Adapter (DG/UX Intel、SCO UnixWare および Solaris Intel の場合)	6-13
ntisbsdmd 同報通信デーモン	6-13
ntspxctl ユーティリティ	6-14
SPX/IPX Protocol Adapter のコマンドの要約	6-15
getname コマンド	6-16
SPX/IPX の ADDRESS 指定	6-18
Oracle Enterprise Manager (OEM) Intelligent Agent	6-19
Oracle Advanced Networking Option	6-19

7 Intel UNIX での Oracle データ・カートリッジ・デモの実行

データ・カートリッジ共通の要件	7-2
Oracle8 Time Series Cartridge	7-2
Time Series Cartridge デモのインストール	7-2
Oracle8 Visual Information Retrieval Cartridge	7-3
デモの作成	7-3
Oracle8 Image Cartridge	7-4
デモの作成	7-4

索引

はじめに

用途

このマニュアルでは、Oracle8 Server を管理およびチューニングするために必要な Intel UNIX (DG/UX Intel、NCR MP-RAS、SCO UnixWare、Solaris Intel) 固有の情報を説明します。このマニュアルの内容は、Oracle8 マニュアル・セットの内容を補足しています。

対象読者

このマニュアルは、Intel UNIX システムでの Oracle8 Server の管理者を対象としています。

Oracle8 および Oracle8 Enterprise Edition

特に指示がない限り、このマニュアルで説明している特長や機能は、Oracle8 および Oracle8 Enterprise Edition に共通です。

このマニュアルで使用する表記規則

クーリエ体	クーリエ体は、UNIX コマンド、ディレクトリ名、パス名およびファイル名を表します。
大カッコ []	大カッコで囲まれた語は、キーの名前を表します（たとえば、[Return] キーを押す）。ただし、コマンド構文の中で使用される場合、大カッコは別の意味になります。
イタリック体	イタリック体は、変数（ファイル名の中で変化する部分を含む）を表します。
大文字	大文字は、SQL コマンド、初期化パラメータまたは環境変数を表します。

UNIX では大文字と小文字が別のものとして認識されるため、このマニュアルで使用する表記規則は、他の Oracle 製品のマニュアルで使用されている表記規則と多少異なります。

コマンドの構文

コマンド構文は、クーリエ体で表します。コマンドで使用する構文は、次のとおりです。

バックスラッシュ \ バックスラッシュは、1 行にコマンドが入りきらない場合に使用します。このマニュアルで記載されているとおりに入力する（バックスラッシュを付ける）か、またはバックスラッシュを付けずに 1 行で入力します。

```
dd if=/dev/rdisk/c0t1d0s6 of=/dev/rst0 bs=10b \
count=10000
```

中カッコ {} 中カッコは、必ず選択する項目を表します。

```
.DEFINE {macro1}
```

大カッコ [] 大カッコは、任意に選択する項目を表します。

```
cvtcrt termname [outfile]
```

ただし、本文中で使用される場合、大カッコは別の意味になります。

省略記号 ... 省略記号は、同じ項目を任意の回数だけ繰り返すことを表します。

```
CHKVAL fieldname value1 value2 ... valueN
```

イタリック体 イタリック体は、変数を表します。変数の箇所を適切な値に置き換えてください。

```
library_name
```

縦棒線 | 縦棒線は、中カッコまたは大カッコで囲まれている選択項目を表します。

```
SIZE filesize [K|M]
```

カスタマ・サポート・センターへの連絡方法

オラクル社カスタマ・サポート・センターへは、次の電話番号におかけください（営業時間は、各地域のカスタマ・サポート・センターにお問い合わせください）。

- アメリカ合衆国の場合：+1.650.506.1500
- ヨーロッパの場合：+44.1344.860160
- アジアの場合：+81.3.5717.1860

詳細は、<http://www.oracle.com/support> にアクセスしてください。

参照ドキュメント

本番データベース・システム用の拡張構成およびチューニングの詳細は、次のドキュメントを参照してください。

- 『Oracle8 Server 管理者ガイド』
Oracle8 Server に関するさまざまなタスク（データベースの作成、データベース・オブジェクトの管理、ユーザーの作成など）を行う際に、まずお読みください。
- 『Oracle Net8 管理者ガイド』
- 『Oracle8 Server チューニング』

Oracle リレーショナル・データベース管理システムに慣れていない場合は、最初に『Oracle8 Server 概要』の第1章「Oracle8 Server の基礎知識」をお読みください。

Oracle8 での最適フレキシブル・ アーキテクチャ (OFA)

- 最適フレキシブル・アーキテクチャ (OFA)
- Oracle8 for UNIX で適用される OFA

最適フレキシブル・アーキテクチャ（OFA）

オラクル社では、Oracle8 に対して OFA 標準を適用することをお薦めしています。OFA 標準は、高速で、メンテナンスの必要性を最小限に抑えた信頼性の高いデータベース構成を実現するためのガイドラインです。

OFA には、次の目的があります。

- デバイスのボトルネックやパフォーマンスの低下が起きないように、ディスク上の大量で複雑なソフトウェアおよびデータを編成します。
- データの損傷を起こす可能性のある管理タスク（ソフトウェアやデータのバックアップ機能など）を簡略化します。
- 複数の Oracle データベースでの移動を簡単にします。
- データベースの拡張を適切に管理します。
- データ・ディクショナリの空き領域の断片化を回避し、その他の領域で起きる断片化を分離し、リソースの競合を最小限に抑えます。

OFA に準拠したデータベースの特長

OFA に準拠したデータベースには、次の利点があります。

ファイル・システムの編成

ファイル・システムは、管理が容易で、かつ次のスケーラビリティに対応できるように編成されています。

- 既存のデータベースへのデータの追加
- ユーザーの追加
- データベースの作成
- ハードウェアの追加

I/O 負荷の分散

I/O 負荷を十分な数のディスク・ドライブに分散させるので、パフォーマンスのボトルネックが起きません。

ハードウェア・サポート

操作上の考慮事項に矛盾しない範囲で、ハードウェア・コストを最小限に抑えられます。

ドライブの障害に対する予防措置

複数ドライブにアプリケーションを分散することによって、ドライブに障害が起きた場合でも、アプリケーションへの影響を最小限に抑えられます。

ホーム・ディレクトリの分散

次のものを、複数のディスク・ドライブに分散できます。

- 複数のホーム・ディレクトリ
- 各ホーム・ディレクトリの内容

ログイン・ホーム・ディレクトリの整合性

ログイン・ホーム・ディレクトリを参照するプログラムを修正しなくても、ログイン・ホーム・ディレクトリを追加、移動または削除できます。

UNIX ディレクトリ・サブツリーの独立性

ファイルのカテゴリは、独立した UNIX ディレクトリ・サブツリーに分類されています。そのため、あるカテゴリのファイルでの操作によって、別のカテゴリのファイルが受ける影響を最小限に抑えることができます。

アプリケーション・ソフトウェアの同時実行サポート

複数バージョンのアプリケーション・ソフトウェアを同時に実行できます。これによって、ユーザーはアプリケーションの古いバージョンを削除しなくても、新しいリリースのアプリケーションを使用できます。アップグレード後に新しいバージョンへ移る作業は、管理者にとっては簡単な作業で、ユーザーがその作業を意識する必要はありません。

各データベースの管理情報の区別

データベースごとに管理情報を区別できるため、管理データをわかりやすい構造で編成および保存できます。

データベース・ファイルのネーミング規則

データベース・ファイルには、次のネーミング規則があります。

- 他のファイルと区別しやすい名前にします。
- 2つのデータベースに属するファイルが、どちらのデータベースに属するかを区別できる名前にします。
- 制御ファイル、REDO ログ・ファイルおよびデータ・ファイルであることが識別できる名前にします。
- データ・ファイルと表領域の関係を明確に識別できる名前にします。

表領域の内容の分割

表領域の内容が分割されているため、次の利点があります。

- 表領域内の空き領域の断片化を最小限に抑えられます。
- I/O 要求の競合を最小限に抑えられます。
- 管理面の柔軟性が高くなります。

すべてのドライブに分散されている I/O 負荷のチューニング

Oracle データを RAW デバイスに保存しているドライブを含め、すべてのドライブへの I/O 負荷をチューニングします。

Parallel Server における OFA のその他の利点

Oracle Parallel Server のインストールでは、他にも次の利点があります。

- すべての管理データを一元管理できます。
- あるインスタンスの管理データは、ファイル名によってそのインスタンスと対応付けることができます。

Oracle8 for UNIX で適用される OFA

データベース・ファイルのネーミング規則に従うことによって、データ管理のさまざまな問題を回避できます。ここで説明している OFA 規則は、『The OFA Standard: Oracle8 for Open Systems』(Cary V.Millsap 著)に記載されている OFA 推奨事項に対応しています。

マウント・ポイントの名前

マウント・ポイントの構文

すべてのマウント・ポイントは、*/pm* という構文で名前を付けます。この場合、*p* は文字列定数、*m* は各マウント・ポイントを区別するための一意の固定長キー（通常、2 桁の数字）です。たとえば、*/u01* および */u02*、*/disk01* および */disk02* などです。

超大規模データベース（VLDB）のマウント・ポイントの名前

1 つのアプリケーションが使用するデータベース・ファイルが各ディスク・ドライブにあり、各データベースで I/O のボトルネックが発生しないほど十分なドライブがある場合、*/q/dm* という構文で、マウント・ポイントに名前を付けます（表 1-1 を参照）。

表 1-1 マウント・ポイントの名前の構文

<i>q</i>	Oracle データが保存されていることを示す文字列
<i>dm</i>	初期化パラメータ DB_NAME の値 (単一インスタンス・データベースのインスタンス <i>sid</i> と同じ)

たとえば、*/u01/oradata/test01* および */u01/oradata/test02* という名前のマウント・ポイントは、Oracle テスト・データベースに 2 つのドライブを割り当てます。

ディレクトリの名前

ホーム・ディレクトリの構文

/pm/h/u という構文で、ホーム・ディレクトリに名前を付けます（表 1-2 を参照）。

表 1-2 ホーム・ディレクトリの名前の構文

<i>pm</i>	マウント・ポイントの名前
<i>h</i>	標準のディレクトリ名
<i>u</i>	ディレクトリの所有者の名前

たとえば、*/u01/app/oracle* は、Oracle Server ソフトウェアの所有者のホーム・ディレクトリ（ORACLE_BASE の値で、Installer を使用してインストールする場合のデフォルト値）で、*/u01/app/applmgr* は、Oracle アプリケーション・ソフトウェアの所有者のホーム・ディレクトリです。

ホーム・ディレクトリを UNIX ファイル・システム内の同一レベルに置くと、別のマウント・ポイントにあるアプリケーション所有者のログイン・ホーム・ディレクトリの集まりを、決まった形式（たとえば、`/* /app/*`）で参照できます。

パス名の参照

明示的なパス名は、そのパス名を保存するためのファイル（`/etc/passwd`、Oracle `oratab` ファイルなど）で参照します。グループのメンバーは、`/etc/group` ファイルで参照します。

ソフトウェア・ディレクトリ

複数バージョンのアプリケーション・ソフトウェアを同時に実行するための OFA 要件を満たすために、各バージョンの Oracle8 Server ソフトウェアを `/pm/h/product/v` という構文の 1 つのディレクトリに保存します（[表 1-3](#) を参照）。

表 1-3 Oracle8 Server ソフトウェアを保存するディレクトリ名の構文

<i>pm</i>	マウント・ポイントの名前
<i>h</i>	標準のディレクトリ名
<i>v</i>	ソフトウェアのバージョン

たとえば、`/u01/app/oracle/product/8.0.6` は、Oracle8 Server ファイルが保存されているディレクトリ構造の先頭を示します。環境変数 `ORACLE_HOME` は、この値に設定します。

ファイルの名前

管理ファイル

管理データの編成を簡略化するために、データベース固有の管理ファイルを `h/admin/d/a/` というサブディレクトリに保存しておくことをお勧めします。この場合、`h` は oracle ソフトウェア所有者のホーム・ディレクトリ、`d` はデータベース名 (DB_NAME)、`a` は次に示すデータベース管理ファイルのそれぞれのサブディレクトリを表しています (表 1-4 を参照)。

表 1-4 データベース管理ファイルのサブディレクトリ

adhoc	指定したデータベースの非定型 SQL スクリプト
arch	アーカイブ REDO ログ・ファイル
adump	監査ファイル (<code>configdb_name.ora</code> の <code>AUDIT_FILE_DEST</code> を、この位置に設定します。このサブディレクトリは、定期的に整理してください。)
bdump	バックグラウンド・プロセスのトレース・ファイル
cdump	コア・ダンプ・ファイル
create	データベースを作成するプログラム
exp	データベース・エクスポート・ファイル
logbook	データベースの状態および履歴を記録するファイル
pfile	インスタンス・パラメータ・ファイル
udump	ユーザー SQL トレース・ファイル

たとえば、サブディレクトリ `adhoc` のパス名は、`/u01/app/oracle/admin/sab/adhoc/` です。

データベース・ファイル

次のネーミング規則に従ってデータベース・ファイルに名前を付けることによって、それらの識別が簡単になります。

- 制御ファイルの場合、`/pm/q/d/control.ctl`
- REDO ログ・ファイルの場合、`/pm/q/d/redon.log`
- データ・ファイルの場合、`/pm/q/d/tn.dbf`

表 1-5 に、この構文の詳細を示します。

表 1-5 データベース・ファイルの名前の構文

<i>pm</i>	マウント・ポイント名（前述）
<i>q</i>	Oracle データを他の Oracle ファイルと区別するための文字列 （通常、ORACLE または oradata）
<i>d</i>	データベースの DB_NAME
<i>t</i>	Oracle 表領域名
<i>n</i>	2 桁の数字

注意： */pm/q/d* に、データベース *d* に関連する制御ファイル、REDO ログ・ファイル、データ・ファイル以外のファイルは保存しないでください。

たとえば、この規則に従って */u03/oradata/sab/system01.dbf* という名前のデータ・ファイルを作成すると、そのファイルが属しているデータベースを簡単に参照できます。

要件に応じた個々のセグメント

異なる表領域間で、存在期間、I/O 要求の必要性およびバックアップの頻度別に、セグメントをいくつかのグループに分類します。

Oracle データベースごとに、表 1-6 に示す特殊な表領域を作成します。アプリケーション・セグメントに必要な表領域以外に、これらの表領域が必要です。

表 1-6 特殊な表領域

SYSTEM	データ・ディクショナリ・セグメント
TEMP	テンポラリ・セグメント
RBS	ロールバック・セグメント
TOOLS	汎用ツール
USERS	その他のユーザー・セグメント

この方法でセグメントを分割すると、ディクショナリ・セグメントが削除されることはなく、また、削除できるその他のセグメントが SYSTEM 表領域に保存されることはありません。これによって、表領域の空き領域が断片化したために SYSTEM 表領域を再作成する必要はなくなります。

アプリケーション・データが保存されている表領域にロールバック・セグメントが保存されることはないため、管理者は、アプリケーションの表領域をオフラインにしてメンテナンスを行うことができます。セグメントは種類別に分割されているので、管理者は、複雑なツールを使用しなくてもデータ拡張の記録および見積りができます。

表領域の名前

表領域の名前は、8 文字以下で指定します。

Oracle8 では表領域に 30 文字の名前を付けられますが、UNIX ファイル名としては、14 文字までに制限されています。データ・ファイル名は、*tn.dbf* という構文で指定することをお勧めします。この場合、*t* は表領域名、*n* は 2 桁の数字からなる文字列です。拡張子と 2 桁の数字からなる文字列の合計が 6 文字になるため、表領域名に使用することができるのは、8 文字までになります。

データ・ファイルの名前とそのデータ・ファイルを使用する表領域の関連性がわかるような名前にします。たとえば、General Ledger データと索引を保存するための表領域には、それぞれ GLD および GLX という名前を指定します。

注意： 表領域の名前に「tablespace」という語を連想させる文字列を付ける必要はありません。表領域は、前後関係から識別できるので、名前によってタイプに関する情報を示す必要はありません。

OFA 構造に基づいた Oracle ファイル

表 1-7 に、ファイルのクラスを識別するための構文を示します。

表 1-7 ファイルのクラスを識別するためのディレクトリ構造の構文

/u[0-9][0-9]	ユーザー・データ・ディレクトリ
/*/home/*	ユーザー・ホーム・ディレクトリ
/*/app/*	ユーザー・アプリケーション・ソフトウェア・ディレクトリ
/*/app/applmgr	Oracle apps ソフトウェアのサブツリー
/*/app/oracle/product	Oracle Server ソフトウェアのサブツリー
/*/app/oracle/product/8.0.6	Oracle Server 8.0.6 の配布ファイル
/*/app/oracle/admin/sab	sab データベースの管理サブツリー
/*/app/oracle/admin/sab/arch/*	sab データベースのアーカイブ・ログ・ファイル
/*/oradata	Oracle データのディレクトリ
/*/oradata/sab/*	sab データベース・ファイル
/*/oradata/sab/*.log	sab データベースの REDO ログ・ファイル

OFA ファイルのマッピング

表 1-8 に、ファイルのマウント・ポイント、アプリケーション、データベース、表領域などが含まれる、OFA に準拠したサンプル・データベースのファイル・マッピングの階層を示します。ファイル名によって、ファイルの種類（制御、ログまたはデータ）がわかります。

表 1-8 OFA インストールの階層型ファイル・マッピング

/	root マウント・ポイント
u01/	「ユーザー・データ」のマウント・ポイント #1
app/	app ソフトウェアのサブツリー
oracle/	oracle ソフトウェア所有者のホーム・ディレクトリ
admin/	データベース管理ファイルのサブツリー
TAR/	サポート・ログのサブツリー
db_name1/	db_name1 データベースの管理サブツリー
db_name2/	db_name2 データベースの管理サブツリー
doc/	オンライン・ドキュメント
local/	ローカルの Oracle ソフトウェアのサブツリー
aps6/	Oracle6 管理パッケージ
aps7/	Oracle7 管理パッケージ
product/	配布ファイル
7.3.2/	7.3.2 インスタンスの ORACLE_HOME
7.3.3/	7.3.3 インスタンスの ORACLE_HOME
8.0.6/	8.0.6 インスタンスの ORACLE_HOME
home/	ログイン・ホーム・ディレクトリのサブツリー
ltb/	ユーザーのホーム・ディレクトリ
sbm/	ユーザーのホーム・ディレクトリ
oradata/	Oracle データのサブツリー
db_name1/	db_name1 データベース・ファイルのサブツリー
db_name2/	db_name2 データベース・ファイルのサブツリー
u02/	「ユーザー・データ」のマウント・ポイント #2
home/	ログイン・ホーム・ディレクトリのサブツリー
cvm/	ユーザーのホーム・ディレクトリ
vrm/	ユーザーのホーム・ディレクトリ
oradata/	Oracle データのサブツリー
db_name1/	db_name1 データベース・ファイルのサブツリー
db_name2/	db_name2 データベース・ファイルのサブツリー
u03/	「ユーザー・データ」のマウント・ポイント #3
home/	ログイン・ホーム・ディレクトリのサブツリー
oradata/	Oracle データのサブツリー
db_name1/	db_name1 データベース・ファイルのサブツリー
db_name2/	db_name2 データベース・ファイルのサブツリー

RAW デバイスのサイズ

Oracle データベース・ファイルを保存するための RAW デバイスはすべて、標準的なサイズに設定します。

できるだけ 1 つのサイズに標準化してください。これによって RAW ファイルをあるパーティションから別のパーティションに確実に移動できます。必要な数のデータベース・ファイルを保存できる程度の大きさで、かつ、操作しにくくならない程度の大きさに設定してください。

たとえば、2GB のドライブのパーティションを切る場合、サイズと数のバランスを取るために、大きさが 200MB のパーティションが 10 個できるように区切るとよいでしょう。RAW デバイスを使用するすべての表領域は、複数のドライブにストライプ化してください。できれば、論理ボリューム・マネージャを使用してストライプ化してください。

複数インスタンスにおける OFA に準拠したデータベースのファイル・マッピング

複数インスタンス・データベース（Oracle Parallel Server インストレーション）の場合、ファイル・マッピングについて、次のガイドラインがあります。

Oracle Parallel Server の管理用ホーム・ディレクトリ

Oracle Parallel Server を使用する場合、そのクラスタに対して Oracle 管理用ホーム・ディレクトリとして機能するノードを 1 つ選択します。管理用ホーム・ディレクトリには、管理サプツリーが含まれます。~/admin/d/ にある bdump、cdump、logbook、pfile および udump ディレクトリのデータベースにアクセスする各インスタンスにサブディレクトリを作成します。管理ホームの admin ディレクトリは、すべてのインスタンスの admin ディレクトリにマウントする必要があります。表 1-9 に例を示します。

表 1-9 二重インスタンスを持つ Oracle Parallel Server の管理ディレクトリ構造

u01/app/oracle/admin/sab/		sab データベース用の管理ディレクトリ
adhoc/		スクリプト用のディレクトリ
arch/		すべてのインスタンス用のログ・アーカイブ先
	redo001.arc	アーカイブ REDO ログ・ファイル
bdump/		バックグラウンド・ダンプ・ファイルのディレクトリ
	inst1/	inst1 インスタンスのバックグラウンド・ダンプ先
	inst2/	inst2 インスタンスのバックグラウンド・ダンプ先
cdump/		コア・ダンプ・ファイル用のディレクトリ
	inst1/	inst1 インスタンスのコア・ダンプ先
	inst2/	inst2 インスタンスのコア・ダンプ先
create/		作成スクリプト用のディレクトリ
	l-rdbms.sql	inst データベース作成用の SQL スクリプト
exp/		エクスポート用のディレクトリ
	990920full.dmp	9 月 20 日のフル・エクスポート・ダンプ・ファイル
	export/	エクスポート・ファイル用のディレクトリ
	import/	インポート・ファイル用のディレクトリ
logbook/		inst ログブック項目用のディレクトリ
	inst1/	inst1 インスタンス・レポート用のディレクトリ
		params.lst inst1 インスタンスの v\$parameter レポート
	inst2/	inst2 インスタンス・レポート用のディレクトリ
		params.lst inst2 インスタンスの v\$parameter レポート
	user.lst	dba_users レポート
pfile/		インスタンス・パラメータ・ファイル用のディレクトリ
	inst1/	inst1 インスタンス・パラメータ用のディレクトリ
		init inst1 インスタンスのインスタンス・パラメータ
	inst2/	inst2 インスタンス・パラメータ用のディレクトリ
		init inst2 インスタンスのインスタンス・パラメータ
udump/		ユーザー・ダンプ・ファイル用のディレクトリ
	inst1/	inst1 インスタンスのユーザー・ダンプ先
	inst2/	inst2 インスタンスのユーザー・ダンプ先

ディレクトリ構造

ORACLE_BASE ディレクトリ

ORACLE_BASE は、Oracle ディレクトリ構造のルートです。表 1-10 に、ORACLE_BASE ディレクトリ構造の詳細を示します。Oracle Installer を使用して OFA に準拠したデータベースをインストールすると、ORACLE_BASE はデフォルトで */pm/app/oracle* に設定されています。

表 1-10 ORACLE_BASE ディレクトリ構造および内容

admin	管理ファイル
doc	オンライン・ドキュメント
local	ローカルの Oracle ソフトウェアのサブツリー
product	Oracle ソフトウェア

ORACLE_HOME ディレクトリ

OFA に準拠した Oracle Server をインストールした場合、ORACLE_HOME ディレクトリは、*/mount_point/app/oracle/product/release_number* です。表 1-11 に、ORACLE_HOME ディレクトリ構造の詳細を示します。UNIX の場合、ORACLE_HOME ディレクトリには、次のサブディレクトリ、および各 Oracle 製品用のサブディレクトリが含まれます。

表 1-11 ORACLE_HOME ディレクトリ構造および内容

bin	すべての製品のバイナリ・ファイル
ctx	ConText カートリッジ
db	<i>init^{sid}.ora</i> 、 <i>lks^{sid}</i>
jdbc	JDBC ドライバ
lib	Oracle 製品ライブラリ
md	Spatial カートリッジ
mlx	Xerox Stemmer (ConText カートリッジ用)
network	Net8
nlsrtl	NLS ランタイム読み込み可能データ
ocommon	すべての製品に共通のファイル
odg	データ収集機能
opsm	Parallel Server Manager Components
oracore	コア・ライブラリ

表 1-11 ORACLE_HOME ディレクトリ構造および内容

orainst	マスター・インストール・ファイルおよびプログラム
ord	データ・カートリッジ
otrace	Oracle TRACE
plsql	PL/SQL
precomp	プリコンパイラ
rdbms	データベースに必要なサーバー・ファイルおよびライブラリ
slax	SLAX 解析機能
sqlplus	SQL*Plus
svrmgr	Server Manager

Oracle 製品のサブディレクトリ

Oracle 製品には、システムで利用できる Oracle 製品および購入する製品によって、[表 1-12](#) に示すサブディレクトリがあります。

表 1-12 Oracle 製品のサブディレクトリ

network	Oracle Net8
ocommon	ライブラリおよび SQL メッセージ。このディレクトリは自動的にインストールされ、すべての製品がこのディレクトリを使用します。
plsql	プロシージャ・オプションの PL/SQL バージョン 2
sqlplus	SQL*Plus
svrmgr	Server Manager

製品サブディレクトリの内容

各製品のサブディレクトリには、[表 1-13](#) に示すサブディレクトリが含まれます。

表 1-13 製品のサブディレクトリの内容

admin	管理 SQL およびシェル・スクリプト (catalog.sql、catexp.sql、demo.sql など)
admin/*	他の製品専用ディレクトリ
admin/resource	リソース・ファイル
admin/terminal	ランタイム端末ファイル
demo	デモンストレーション・スクリプトおよびデータ・ファイル
doc	README ファイル (readmeunix.doc など)
install	製品のインストール・スクリプト
lib	製品ライブラリおよび配布された Make ファイル
log	トレース・ファイルおよびログ・ファイル (orasrv.log、*.trc ファイルなど)
mesg	アメリカ英語メッセージ・ファイル、多言語オプション (旧各国語サポート) のメッセージ・テキスト、およびバイナリ・ファイル (oraus.msg、oraus.msb など)

製品のサブディレクトリの例

[表 1-14](#) に、製品のサブディレクトリの例を示します。

表 1-14 製品のサブディレクトリの例

rdbms	install、lib、admin、doc、mesg、log
sqlplus	install、demo、lib、admin、doc、mesg

admin ディレクトリのファイルのネーミング規則

表 1-15 に、rdbms/admin ディレクトリに含まれる SQL スクリプトを示します。

表 1-15 admin ディレクトリのファイルのネーミング規則

cat*.sql	カタログおよびデータ・ディクショナリ表 / ビューを作成します。 インストール時に次のファイルが自動的に実行されます。 catalog.sql (すべてのインストレーション) catproc.sql (すべてのインストレーション) catparr.sql (Parallel Server オプションのインストレーション) catrep.sql (すべてのインストレーション)
dbms*.sql	その他のデータベース・パッケージ
utl*.sql	データベース・ユーティリティ用の表およびビューを作成するスクリプト

ファイル名の拡張子

表 1-16 に、ファイル名の拡張子を示します。

表 1-16 ファイル名の拡張子

.a	オブジェクト・ファイル・ライブラリ、Ada ランタイム・ライブラリ
.ada	Ada ソース・ファイル
.aud	Oracle 監査ファイル
.bdf	X11 フォント記述ファイル
.bmp	X11 ビットマップ・ファイル
.c	C ソース・ファイル
.ctl	SQL*Loader 制御ファイル、Oracle Server 制御ファイル
.dat	SQL*Loader データ・ファイル
.dbf	Oracle Server 表領域ファイル
.dei	ORCA 削除スクリプト
.dmp	エクスポート・ファイル
.doc	ASCII テキスト・ファイル
.env	環境設定用のシェル・スクリプト・ファイル
.f	FORTRAN ソース・ファイル
.h	C ヘッダー・ファイル。sr.h は SQL*Report Writer のヘルプ・ファイル。
.ins	ORCA インストレーション・スクリプト

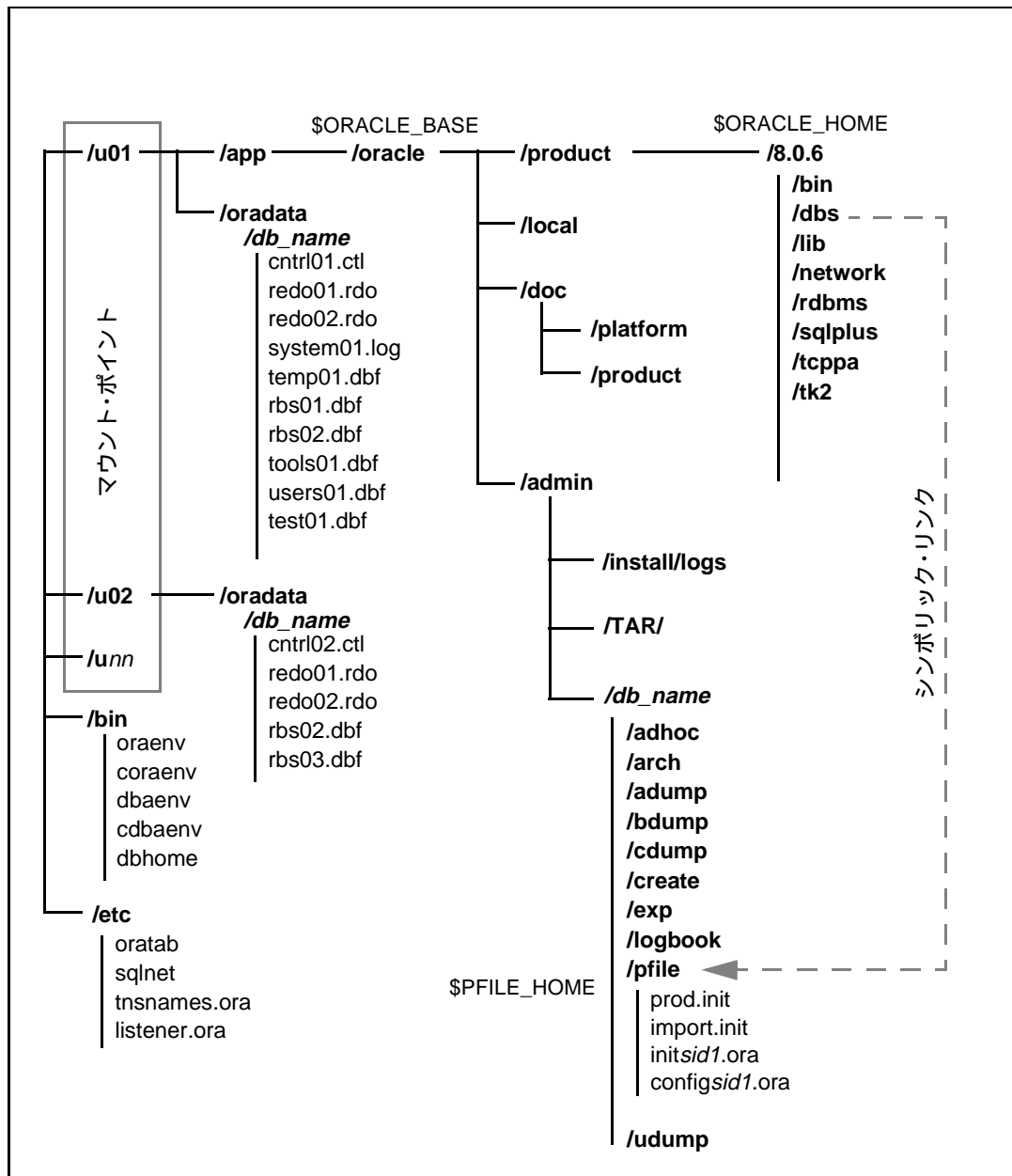
表 1-16 ファイル名の拡張子

.l	UNIX マニュアル・ページ
.lst	SQL*Plus スクリプトの出力
.log	インストール・ログ・ファイル、Oracle Server REDO ログ・ファイル
.map	Installer 製品コンポーネント・ファイル
.mk	make ファイル
.msb	NLS メッセージ・ファイル (バイナリ)
.msg	NLS メッセージ・ファイル (テキスト)
.o	オブジェクト・モジュール
.ora	Oracle 構成ファイル
.orc	インストール・プロトタイプ・ファイル
.pad	Pro*Ada ソース・ファイル
.pc	Pro*C ソース・ファイル
.pco	Pro*COBOL ソース・ファイル
.ppd	プリンタ・ドライバ・ファイル
.pfo	Pro*FORTRAN ソース・ファイル
.prd	製品登録テンプレート・ファイル (orainst 用)
.res	Toolkit II リソース・ファイル
.sh	Bourne シェル・スクリプト・ファイル
.sql	SQL* スクリプト・ファイル
.sys	Bourne シェル・スクリプト・ファイル
.tab	SQL* スクリプト・ファイル
.trc	トレース・ファイル
.tut	Bourne シェル・スクリプト・ファイル
.us	orainst メッセージ・ファイル
.utd	ユニフォーム端末定義
.vrf	Installer 依存関係検証スクリプト

デフォルトの OFA データベース

図 1-1 に、Oracle Installer によって作成されるデフォルトの OFA データベースを示します。

図 1-1 デフォルトの Oracle インストール



Intel UNIX での Oracle8 の管理

- initsid.ora ファイルのカスタマイズ
- 環境設定
- Oracle8 の環境変数
- Oracle8 Server のメモリー使用量の見積り
- クラスタ・サイズおよび索引サイズの計算
- 初期化パラメータ
- システム・グローバル領域の制御
- 特殊アカウントおよび特殊グループの管理
- ログイン・ホーム・ディレクトリの管理
- デモンストレーションの作成と実行
- Oracle8 Server の SQL リファレンス
- トレース・ファイルとアラート・ファイルの使用
- RAW デバイス
- 動的共有メモリー（DSHM）の使用（SCO UnixWare の場合）

initsid.ora ファイルのカスタマイズ

ここでは、Oracle8 配布で提供されるデフォルトの `initsid.ora` ファイルについて説明します。このファイルは、Oracle Installer によって `$ORACLE_BASE/admin/db_name/pfile` ディレクトリに作成されます。このファイルを変更することによって、Oracle8 インストールレーションをカスタマイズできます。

`initsid.ora` ファイルのパラメータには、どんなサイズのインストールにも共通の設定値があります。インストールのサイズに応じて値を変更する必要があるパラメータに対しては、小、中、大の 3 つのパターンが用意されています。サンプル `initsid.ora` ファイルでは、インストール・サイズによって設定が異なるパラメータが示されています。インストールに適用しない設定があれば、行頭に「#」を付けてコメントにできます。

表 2-1 に、`initsid.ora` ファイルに用意されている 3 つのパターンに対するおよその SGA サイズを示します。

表 2-1 サンプル `initsid.ora` ファイルのブロック・サイズおよび SGA サイズ

インストール/データベース・サイズ			
ブロック・サイズ	小	中	大
2KB	4500KB	6800KB	17000KB
4KB	5500KB	8800KB	21000KB

サンプル `initsid.ora` ファイル

サンプル `initsid.ora` ファイルは、RDMBS インストールレーションをカスタマイズする際に使用すると便利です。サンプル `initsid.ora` ファイルのパラメータには、どんなサイズのインストールにも共通の設定値があります。インストールのサイズに応じて値を変更する必要があるパラメータに対しては、小、中、大の 3 つのパターンが用意されています。インストール・サイズに応じてチューニングする必要のあるパラメータには設定値が 3 つあり、それぞれにインストール・サイズを示すコメントが付いています。

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

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

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

```

db_block_buffers = 100                                # SMALL
# db_block_buffers = 550                                # MEDIUM
# db_block_buffers = 3200                                # LARGE

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

log_checkpoint_interval = 10000

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

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

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

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

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

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

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

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

```

```
# If using public rollback segments, define how many
# rollback segments each instance will pick up, using the formula
# # of rollback segments = transactions / transactions_per_rollback_segment
# In this example each instance will grab 40/10 = 4:
# transactions = 40
# transactions_per_rollback_segment = 10

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

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

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

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

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

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

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

# oracle_trace_enable = TRUE
```

環境設定

環境変数の表示および設定

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

```
$ echo $ORACLE_SID
```

現行セッションでの環境変数値の設定およびエクスポート

Bourne または Korn シェルの場合、次のように入力します。

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

C シェルの場合、次のように入力します。

```
% setenv ORACLE_SID test
```

この場合、test は環境変数 ORACLE_SID に設定する値です。

共通の環境設定

Oracle8 では、データベース管理者 (DBA) がすべてのユーザーに共通の環境を設定できます。共通の環境を設定することによって、システム管理者およびデータベース管理者が Oracle Server システムを変更しやすくなります。

oraenv コマンド・ファイル

oraenv (C シェルの場合は coraenv) コマンド・ファイルは、インストール時に作成されます。このファイルには、Oracle 環境変数の値および次の情報が含まれています。

- データベースの変更に必要なユーザー・アカウントを更新するための主な方法
- Oracle Server データベース間を移動するメカニズム

たとえば、データベースを /usr/oracle から /usr1/oracle へ移動する必要があるとします。共通の環境設定のルーチンがない場合は、ユーザー起動ファイルを個々に更新する必要があります。oraenv を使用すると、各ユーザー・プロファイルから oraenv コマンド・ファイルがコールされ、そのファイルのみが更新されます。

ローカル bin ディレクトリ

oraenv (または coraenv) および dbhome を、Oracle ソフトウェアのホーム・ディレクトリではなくローカル bin ディレクトリに入れると、すべてのユーザーがそれらのファイルにアクセスできるようになります。また、ローカル bin ディレクトリに入れることによって、別の ORACLE_HOME を示すようにパスを変更した場合でも、oraenv (または coraenv) は変更せずにそのまま使用できます。

データベース間の移動

あるデータベースまたはインスタンスから別のデータベースまたはインスタンスへ移動するには、oraenv ルーチンをコールし、プロンプトに対して移動先のデータベースの sid を指定します。必ず oraenv コマンド・ファイルのフルパス名を入力してください。次に例を示します。

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

データベースの例

次に、ローカル bin ディレクトリが /usr/lbin で、本番データベースが PROD の場合の例を示します。ORACLE_SID を入力するプロンプトが起動時に表示されないようにするには、環境変数 ORAENV_ASK を NO に設定します。

次の例では、oraenv が実行された後、ORAENV_ASK はデフォルトの YES にリセットされます。これによって、次に oraenv が実行されたときに、別の ORACLE_SID を入力するプロンプトが表示されます。

単一インスタンス

Bourne シェルまたは Korn シェルの場合、.profile ファイルに次の行があります。

```
. local_bin_directory/oraenv
```

これを次の行に置き換えます。

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


C シェルの場合、.cshrc ファイルに次の行があります。

```
source local_bin_directory/coraenv
```

これを次の行に置き換えます。

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

複数インスタンス

複数インスタンスの場合は、起動時に *sid* を定義します。

Bourne シェルまたは Korn シェルの場合、次のように入力します。

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

C シェルの場合、次のように入力します。

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

Oracle8 の環境変数

Oracle システムをインストールする前に、UNIX 環境の特定の変数を設定する必要があります。

参照： 詳細は、『Oracle8 for Intel UNIX (DG/UX Intel, NCR MP-RAS, SCO UnixWare, Solaris Intel) インストレーション・ガイド』を参照してください。

UNIX における Oracle 環境変数

表 2-2 に、Oracle8 環境変数の機能、構文および例を示します。

表 2-2 UNIX における Oracle8 環境変数

環境変数	項目	定義
EPC_DISABLED	機能	Oracle TRACE を使用禁止にします。
	構文	TRUE または FALSE
	例	TRUE
NLS_LANG	機能	出力に使用する言語とキャラクタ・セットを指定します。 許容値の詳細は、『Oracle8 for Intel UNIX (DG/UX Intel, NCR MP-RAS, SCO UnixWare, Solaris Intel) インストール・ガイド』を参照してください。
	構文	言語_地域.キャラクタ・セット
	例	french_france.we8dec
ORA_NLS33	機能	言語とキャラクタ・セットが保存されているディレクトリを指定します。
	設定値	\$ORACLE_HOME/ocommon/nls/admin/data
ORACLE_BASE	機能	OFA に準拠した Oracle データベース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	ディレクトリ・パス
	例	/mount_point/app/oracle
ORACLE_HELP	機能	ヘルプ・ファイルがあるディレクトリを指定します。
	構文	ディレクトリ・パス
	例	\$ORACLE_HOME/help/admin/resource
ORACLE_HOME	機能	Oracle 配布ソフトウェアがあるディレクトリを指定します。
	構文	ディレクトリ・パス
	例	/mount_point/app/oracle/product/release_number
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名を指定します。指定しないと、アプリケーションは現在のディレクトリに読み込みおよび書き込みを行います。
	構文	ディレクトリをコロンで区切ったリスト (ディレクトリ:ディレクトリ:ディレクトリ)
	例	/u01/oracle/adhoc/sqlplus: 注意: 最後にピリオドを付けることによって、検索パスに現在の作業ディレクトリが追加されます。

表 2-2 UNIX における Oracle8 環境変数

環境変数	項目	定義
ORACLE_SID	機能	Oracle のシステム ID を指定します。
	構文	英字で始まる英数字列。詳細は、『Oracle8 for Intel UNIX (DG/UX Intel, NCR MP-RAS, SCO UnixWare, Solaris Intel) インストール・ガイド』を参照してください。
	例	SAL1
ORACLE_TERM	機能	端末のタイプ識別子を指定します。Installer および Oracle 製品が正しい Toolkit II (.res) リソース・ファイルを調べる場合に使用します。設定しないと、オペレーティング・システム変数 TERM の値が使用されます。
	構文	文字列
	許容値	tk2c\${ORACLE_TERM}.res が、Toolkit II リソース・ディレクトリにある有効なリソース・ファイルと一致するように、ORACLE_TERM の値を設定します。有効な値の詳細は、『Oracle8 for Intel UNIX (DG/UX Intel, NCR MP-RAS, SCO UnixWare, Solaris Intel) インストール・ガイド』を参照してください。
	例	vt100
ORACLE_TERMINAL	機能	Toolkit II (.res) リソース・ファイルを検索するための追加ディレクトリを指定します。
	構文	ディレクトリ名
	例	\$ORACLE_HOME/guicommon/tk21/admin/terminal
ORACLE_TRACE	機能	インストール時の Bourne シェル・スクリプトのトレース状態を切り替えます。T を設定すると、多くの Oracle シェル・スクリプトは set -x フラグがオンの状態で実行されます。
	許容値	T など
ORAENV_ASK	機能	(c)oraenv が、\$ORACLE_SID または \$ORACLE_HOME を入力するためのプロンプトを表示するかどうかを指定します。NO を設定すると、プロンプトは表示されません。NO 以外を設定するとプロンプトが表示されます。
	構文	文字列
	許容値	NO など

表 2-2 UNIX における Oracle8 環境変数

環境変数	項目	定義
TNS_ADMIN	機能	Oracle Net8 構成ファイルがあるディレクトリを指定します。
	構文	ディレクトリ・パス
	許容値	任意のディレクトリ。詳細は、『Oracle8 for Intel UNIX (DG/UX Intel, NCR MP-RAS, SCO UnixWare, Solaris Intel) インストレーション・ガイド』を参照してください。
	例	\$ORACLE_HOME/network/admin
TWO_TASK	機能	tnsnames.ora ファイルで定義されている、Oracle Net8 接続文字列記述子の別名のデフォルトを指定します。
	構文	使用可能なネットワーク別名
	許容値	tnsnames.ora ファイルで定義されている有効な Oracle Net8 の別名
	例	PRODDB_TCP

注意： 環境変数に、Oracle Server プロセスと同じ名前（たとえば、arch、pmon、dbwr など）は設定しないでください。

ORACLE_HOME および ORACLE_SID の略称

Oracle8 Server のファイルおよびプログラムでは、疑問符「？」は ORACLE_HOME の値を表します。たとえば、Oracle8 では、次の SQL 文中の疑問符を ORACLE_HOME のフルパス名として使用します。

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

@ マークは、\$ORACLE_SID を表します。たとえば、ファイルが特定のインスタンスに属していることを指定する場合、次のように入力します。

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

Oracle8 で使用する UNIX 環境変数

表 2-3 に、Oracle8 で使用する UNIX 環境変数の構文および例を示します。

表 2-3 Oracle8 で使用する UNIX 環境変数

環境変数	項目	定義
DISPLAY	機能	X ベースのツールで使します。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、各ベンダーの X-Window ドキュメントを参照してください。
	構文	hostname:display hostname はディスプレイ・デバイスのネットワーク ID、display は常に 0。
	例	135.287.222.12:0 bambi:0
HOME	機能	ユーザーのホーム・ディレクトリを指定します。
JAVA_HOME	機能	/usr/java (SCO UnixWare の場合)
LANG または LANGUAGE	機能	メッセージなどを出力するためにオペレーティング・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および『Oracle8 for Intel UNIX (DG/UX Intel, NCR MP-RAS, SCO UnixWare, Solaris Intel) インストール・ガイド』を参照してください。
LDOPTS	機能	プラットフォームによっては、デフォルトのリンカー・オプションを指定します。詳細は、ld の man ページを参照してください。
LPDEST	機能	System V ベースのシステムの場合に、ユーザーのデフォルト・プリンタを指定します。
	構文	プリンタ名
	例	docqms
LDPATH	機能	共有オブジェクト・ライブラリを検索するためにリンカーが使用するデフォルトのディレクトリを指定します。詳細は、ld の man ページを参照してください。
LD_LIBRARY_PATH	機能	プラットフォームによっては、実行時に共有ライブラリ・ローダー (ld.so) が共有オブジェクト・ライブラリを検索するために使用します。詳細は、ld.so の man ページを参照してください。 (SCO UnixWare の場合) /usr/java/lib 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 (ディレクトリ:ディレクトリ:ディレクトリ)
	例	/usr/dt/lib:\$ORACLE_HOME/lib

表 2-3 Oracle8 で使用する UNIX 環境変数

環境変数	項目	定義
PATH	機能	シェルが実行プログラムを配置するために使用します。 \$ORACLE_HOME/bin が含まれている必要があります。 (SCO UnixWare の場合) /usr/java/bin 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 (ディレクトリ:ディレクトリ:ディレクトリ)
	例	/bin:/usr/bin:/usr/local/bin: /usr/bin/X11:\$ORACLE_HOME/bin:\$HOME/bin: 注意:最後にピリオドを付けることによって、検索パスに現在の作業ディレクトリが追加されます。
PRINTER	機能	Intel UNIX システムでのユーザーのデフォルト・プリンタを指定します。
	構文	プリンタ名
	例	docqms
SHELL	機能	ホスト・コマンドで使用するコマンド・インタプリタを指定します。
	構文	シェルのパス名
	許容値	/bin/sh、/bin/csh、/bin/ksh またはご使用のシステムに付属のその他のコマンド・インタプリタを指定します。
	例	/bin/sh
TERM	機能	Oracle Toolkit II キャラクタ・モード・ツールなどの UNIX ツールが端末タイプを判断する場合に使用します。
	例	vt100
TMPDIR	機能	テンポラリ・ディスク・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、このディレクトリにテンポラリ・ファイルを作成します。
	構文	ディレクトリ・パス
	例	/u02/oracle/tmp
XENVIRONMENT	機能	X-Window システムのリソース定義を含むファイルを指定します。 詳細は、X-Window のドキュメントを参照してください。

システム時刻の設定

環境変数 TZ には、タイム・ゾーンを設定します。ご使用のオペレーティング・システムのドキュメントを参照して、オペレーティング・システムでこの環境変数を使用できるかどうかを確認してください。

環境変数 TZ によって、時間を夏時間に変更したり、別のタイム・ゾーンにしたりできます。調整した時刻は、ファイルのタイムスタンプを決めたり、date コマンドの出力を生成したり、現在の SYSDATE の値を調べたりする際に使用します。

警告： できるだけ、ユーザーが個々に TZ 値を変更しないでください。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た日が変わってしまいます。その場合、SYSDATE を使用する Oracle アプリケーション（Oracle Financials など）が影響を受けます。この問題を回避するために、表の順序を決定する場合は、日付列ではなく順序番号を使用してください。

Oracle8 Server のメモリー使用量の見積り

Oracle8 Server を起動する前に、次の式を使用して仮想メモリー要件を見積もることができます。

<Oracle 実行ファイルのテキストのサイズ>
 + <SGA のサイズ>
 + $n \times ($ <Tool 実行ファイルのプライベート・データ・セクションのサイズ>
 + <Oracle 実行ファイルの未初期化データ・セクションのサイズ>
 + <スタック用に 65536 バイト>
 + <プロセス・ユーザー領域用に 2048 バイト>)

この場合、 n はバックグラウンド・プロセスの数を示します。

各 Oracle バックエンド接続に対して、次の式を使用して仮想メモリー要件を見積もります。

<Oracle 実行ファイルのデータ・セクションのサイズ>
 + <Oracle 実行ファイルの未初期化データ・セクションのサイズ>
 + <スタック用に 65536 バイト>
 + <プロセス・ユーザー領域用に 2048 バイト>
 + <アプリケーションに必要なカーソル領域>

実行ファイルのテキスト・サイズ、プライベート・データ・セクションのサイズおよび未初期化データ・セクションのサイズ（または *bss*）を見積もるには、size コマンドを使用します。すべての Oracle 実行テキストは常に共有されているため、プログラムが何度起動されても、プログラム・テキストがカウントされるのは 1 回だけです。

実行中のデータベースにユーザーが接続しているときに、実際の Oracle 物理メモリの使用量を計算するには、ps コマンドを使用します。すべてのフロント・エンド、サーバー、および Oracle バックグラウンド・プロセスの項目を検索します。項目ごとに、「プロセスの実際のサイズ (real size of process)」列を加算して、常駐メモリー使用量の小計を計算します。そして、Oracle 実行ファイルのテキスト・サイズと、システムで実行中のその他のすべての Oracle Tool 実行ファイルのテキスト・サイズを、その小計に加算します。実行ファイルが何度起動されても、実行ファイルのサイズがカウントされるのは 1 回です。

参照： ps コマンドで使用できるスイッチのリストの詳細は、各オペレーティング・システムの man ページまたはドキュメントを参照してください。

クラスタ・サイズおよび索引サイズの計算

クラスタ・サイズの計算

表 2-4 のガイドラインに従って、『Oracle8 Server 管理者ガイド』の付録 A に記載されている式を使用してクラスタ・サイズを計算します。

表 2-4 クラスタ・サイズの値

タイプ	サイズ
固定長ヘッダー・サイズ	68 バイト
可変長トランザクション・ヘッダー	表の INITRANS の値 × 24
行ディレクトリ	クラスタ化表の行あたり 4 バイト

索引サイズの計算

表 2-5 のガイドラインに従って、『Oracle8 Server 管理者ガイド』の付録 A に記載されている式を使用して索引サイズを計算します。

表 2-5 索引サイズの値

タイプ	サイズ
固定長ヘッダー・サイズ	113 バイト
可変長トランザクション・ヘッダー	索引の INITRANS の値 × 24
項目ヘッダー	5 バイト

サーバー・リソースの制限

Intel UNIX では、親プロセスからリソースの制限を継承します（オペレーティング・システムのドキュメントの `getrlimit(2)` を参照）。この制限は、ユーザー・プロセスに対して実行される Oracle8 Server のシャドウ・プロセスに適用されます。Intel UNIX のデフォルトのリソース制限は、どの Oracle8 Server のシャドウ・プロセスまたはバックグラウンド・プロセスに対しても十分高い値です。ただし、この制限を低くすると、Oracle8 Server システムに影響が出る場合があります。詳細は、Intel UNIX システム管理者にお問い合わせください。

Oracle dba ユーザー ID 用に設定したディスクの割当て制限によって、Oracle8 システムの操作性が悪くなる場合があります。ディスクの割当て制限を設定する前に、Oracle8 のデータベース管理者および Intel UNIX のシステム管理者にお問い合わせください。

初期化パラメータ

初期化パラメータは、Oracle8 Server インスタンスの `initsid.ora` ファイルで変更できます。

参照： 詳細は、『Oracle8 Server 管理者ガイド』を参照してください。

デフォルトの初期化パラメータの値

表 2-6 に、Intel UNIX における初期化パラメータのデフォルト値を示します。`initsid.ora` ファイルのこれらのパラメータに別の値を設定しなかった場合、すべての Oracle8 Server インスタンスでこれらの値が使用されます。`initsid.ora` ファイルには、デフォルトとは異なる値のパラメータのみを設定してください。

システム・パラメータの現在の設定を表示するには、Server Manager を使用して SQL 文 SHOW PARAMETERS を実行します。

参照： 詳細は、『Oracle8 Server リファレンス・マニュアル』を参照してください。

表 2-6 デフォルトの初期化パラメータ

パラメータ	デフォルト値
BACKGROUND_DUMP_DEST	\$ORACLE_BASE/admin/ <i>sid</i> /bdump
BITMAP_MERGE_AREA_SIZE	1048576
COMMIT_POINT_STRENGTH	1
CONTROL_FILES	\$ORACLE_HOME/dbs/ctrl@.dbf (@ は ORACLE_SID)
CREATE_BITMAP_AREA_SIZE	8388608
DB_BLOCK_BUFFERS	200
DB_BLOCK_SIZE	2048
DB_FILES	80 (最大値は 2000000)
DB_FILE_DIRECT_IO_COUNT	64 (最大値は 1048576)
DB_FILE_MULTIBLOCK_READ_COUNT	8 (許容値は 1 ~ 128。ただし、 DB_BLOCK_BUFFERS の 1/4 以下で設定 してください。)
DISTRIBUTED_TRANSACTIONS	16
HASH_AREA_SIZE	0
HASH_MULTIBLOCK_IO_COUNT	1
LOCK_SGA	FALSE
LOCK_SGA_AREAS	0
LOG_ARCHIVE_BUFFER_SIZE	64
LOG_ARCHIVE_BUFFERS	4 (最大値は 128)
LOG_ARCHIVE_DEST	\$ORACLE_HOME/dbs/arch/
LOG_ARCHIVE_FORMAT	"%t_%s.dbf"
LOG_BUFFER	8192
LOG_CHECKPOINT_INTERVAL	10000
LOG_SMALL_ENTRY_MAX_SIZE	80
MTS_MAX_DISPATCHERS	5
MTS_MAX_SERVERS	20

表 2-6 デフォルトの初期化パラメータ

パラメータ	デフォルト値
MTS_SERVERS	0
MTS_LISTENER_ADDRESS	ADDRESS=address (第 6 章を参照してください)
NLS_LANGUAGE	AMERICAN
NLS_TERRITORY	AMERICA
OBJECT_CACHE_MAX_SIZE_PERCENT	10
OBJECT_CACHE_OPTIMAL_SIZE	102400
OPEN_CURSORS	50
OS_AUTHENT_PREFIX	ops\$
PROCESSES	50
SHARED_POOL_SIZE	3500000
SORT_AREA_SIZE	65536
SORT_READ_FAC	5
SORT_SPACEMAP_SIZE	512
USER_DUMP_DEST	\$ORACLE_BASE/admin/ <i>sid</i> /udump

システム・グローバル領域の制御

システム・グローバル領域 (SGA) は、共有メモリー内に常駐する Oracle 構造体です。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、十分な共有メモリーが必要です。

SGA のサイズ制限

1 つの共有メモリー領域の最大サイズは、Intel UNIX の SHMMAX パラメータで指定します。SGA が 64MB の場合、16MB の共有メモリー領域を 4 つ使用できます。

SGA のサイズが共有メモリー・セグメントの最大サイズ (SHMMAX) を超える場合、Oracle8 は、要求された SGA サイズになるように、連続セグメントをさらに連結しようとします。SHMSEG は、プロセスが接続できるセグメントの最大数です。SHMMAX のサイズが制限されているシステムでは、連続するアドレスでセグメントを連結するために、SHMMAX を最大値に設定する必要があります。

注意： SHMMAX がデータベースの SGA サイズより小さい場合、緊密共有メモリー (ISM) によって問題が発生する場合があります。この問題が発生するのは、Solaris Intel の場合のみです。

SGA のサイズは、次の `init.ora` ファイルのパラメータによって決まります。

- `DB_BLOCK_BUFFERS`
DG/UX Intel NUMA システムの場合、`DB_BLOCK_BUFFERS` は 1024 以上に設定する必要があります。
- `DB_BLOCK_SIZE`
- `SORT_AREA_SIZE`
- `SHARED_POOL_SIZE`

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、マシン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。すべてのインスタンスの SGA サイズの合計は、物理 RAM の合計の 1/3 以下にすることを薦めます。

SGA のサイズの計算

インスタンスの SGA の適切なサイズは、次の式で計算できます。

$$\begin{aligned} & (\text{DB_BLOCK_BUFFERS} \times \text{DB_BLOCK_SIZE}) \\ & + \text{SORT_AREA_SIZE} \\ & + \text{SHARED_POOL_SIZE} \\ & + 1\text{MB} \end{aligned}$$

実行中のデータベースの SGA サイズを表示するには、Server Manager の `show sga` コマンドを使用します。このコマンドによって、SGA サイズがバイト単位で表示されます。

SGA の再配置

SGA を連結するアドレスは、SGA 内のデータベース・バッファやユーザーのアプリケーション・データ領域のカーソルに使用できる仮想アドレス領域の大きさに影響します。

1. 共有メモリー・セグメントを連結するための有効な仮想アドレス範囲を決めます (`tstshm` 表示では、「Lowest shared memory address」および「Highest shared memory address」の行が有効な範囲を示しています)

```
$ tstshm
```

注意： 緊密共有メモリー (ISM) の使用中に `tstshm` を実行すると、システムに問題が発生する場合があります。ISM の使用をやめるには、`initstsid.ora` ファイルに次の行を追加してください。

```
use_ism = false
```

2. `tstshm` の「Segment boundaries」出力を調べて、共有メモリー・セグメントを連結できる有効な仮想アドレス範囲を調べます。
3. SGA のサイズを決めます。データベース・システムを起動すると、ヘッダー「Total System Global Area」の横に SGA サイズが表示されます。
4. 次のように入力して、`$ORACLE_HOME/rdbms/lib` ディレクトリに移動し、`genksms` を実行して `ksms.s` ファイルを生成します。

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

この場合、`sgabeg` は SGA の起動アドレス (DG/UX Intel、NCR MP-RAS、Solaris Intel および SCO UnixWare でのデフォルトは、`0x20000000`) で、手順 2 で調べた範囲内で設定します。

5. 既存の Oracle データベースを停止します。
6. 次のように入力して、`$ORACLE_HOME/rdbms/lib` ディレクトリで Oracle 実行ファイルを作成し直します。

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

`ioracle` を使用して次の処理を行います。

- 古い実行ファイル (`oracle0`) のバックアップを取ります。
- 新しい Oracle 実行ファイルに正しい権限を割り当てます。
- 新しい実行ファイルを `$ORACLE_HOME/bin` ディレクトリに移動します。

その結果、*sgabeg* で指定するアドレスに SGA をロードする新しい Oracle カーネルが作成されます。

genksms -b

genksms -b ユーティリティは、固定 SGA の起点を調整する場合に使用します。

特殊アカウントおよび特殊グループの管理

データベース管理者 (DBA) には、Oracle Server が必要とする特殊アカウントに関する知識と経験が必要で、それらのアカウントが適切なグループに属するように設定する必要があります。ここでは、特殊ユーザー・アカウントについて説明します。[表 2-7](#) に UNIX アカウント、[表 2-8](#) に Oracle Server アカウント、[表 2-9](#) に特殊グループ・アカウントを示します。

表 2-7 UNIX アカウント

<code>oracle</code>	<code>oracle</code> ソフトウェア所有者は、Oracle8 ソフトウェアを所有するアカウントを示します。このメンテナンス・アカウントには、データベースを作成、起動、停止、または INTERNAL 接続するための DBA 権限が必要です。 <code>oracle</code> ソフトウェア所有者は、スーパーユーザーではありません。
<code>root</code>	<code>root</code> ユーザーは、最高の権限 (スーパーユーザー権限) が与えられた特殊 UNIX アカウントです。このアカウントを使用することによって、UNIX カーネルを構成したり、ネットワーク・ソフトウェアを構成およびインストールしたりできます。また、ユーザー・アカウントおよびグループを作成することができます。

表 2-8 Oracle Server アカウント

<code>SYS</code>	インストール時に自動的に作成され、DBA 権限が与えられた標準的な Oracle8 アカウントです。 <code>SYS</code> アカウントは、データ・ディクショナリの実表すべてを所有します。このアカウントは DBA が使用します。
<code>SYSTEM</code>	インストール時に自動的に作成され、DBA 権限が与えられた標準的な Oracle8 アカウントです。 <code>SYSTEM</code> ユーザーでその他の表またはビューを作成することができます。DBA は <code>SYSTEM</code> としてログインし、データベースを監視またはメンテナンスします。

表 2-9 特殊グループ・アカウント

dba グループ	oracle ソフトウェア所有者は、dba グループの唯一の必須メンバーです。dba グループには、root ユーザーなどの UNIX ユーザーを追加できます。このグループのメンバーは、Server Manager の特殊権限機能にアクセスできます。アカウントが dba グループのメンバーでない場合、INTERNAL 接続するためのパスワードを入力するか、または、Server Manager のその他の管理機能へのアクセス権限を取得する必要があります。デフォルトのグループ ID は dba です。
oper グループ	ユーザーが作成することのできる任意の UNIX グループです。メンバーにはデータベースの OPERATOR 権限が与えられます。DBA 権限の一部を制限したものが OPERATOR 権限です。
root グループ	root グループのメンバーになれるのは root ユーザーのみです。

セキュリティの管理

Oracle8 では、UNIX オペレーティング・システムのいくつかの機能を使用し、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。

Oracle8 の 2 タスク・アーキテクチャによって、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。

グループとセキュリティ

Oracle8 データベースのセキュリティを高めるため、オペレーティング・システム・レベルでユーザー・グループを作成します。グループは、UNIX ファイルの `/etc/group` が制御します。Oracle プログラムは、セキュリティを高めるために 2 つのグループに分けられます。すべてのユーザー（UNIX 用語では *other*）がアクセスできる実行プログラムおよび DBA 専用の実行プログラムです。次のようにして、セキュリティを高めることをお勧めします。

- Oracle Server をインストールする前に、データベース管理者のグループ（dba）を作成し、このグループに root および oracle ソフトウェア所有者 ID を割り当てます。dba で実行できるプログラムにのみ、権限 710 が与えられています。システム権限が与えられた Server Manager コマンドは、インストール時に自動的に dba グループに割り当てられます。
- 権限ユーザーで構成される oracle グループを追加して、UNIX ユーザーのサブセットが Oracle8 へのアクセスを制限されるようにします。Oracle ユーティリティに oracle グループ ID を付けます。SQL*Plus など、一般的に実行できるプログラムは、このグ

グループから実行できるようにする必要があります。ユーティリティの権限を 710 に設定して、このグループに実行権限を与えます。other に実行権限は与えないでください。

- other が実行できるプログラムに、権限 711 を与えます。この権限は、データベース・セキュリティに影響しないプログラムにのみ与えてください。

データベース管理者のグループには任意の名前を割り当てられますが、デフォルトのグループ名は dba で、このマニュアルでもその名前を使用しています。このグループ名を変更すると、Oracle Installer はインストール時にカーネルを自動的に再リンクします。同じ ORACLE_HOME でデータベースが複数ある場合（このような構成はお薦めしません）は、それらのデータベース管理者のグループは同じにする必要があります。通常のユーザーのグループ名（oracle グループ）に、この制限はありません。

警告： oracle ソフトウェア所有者および root ユーザーは dba グループに属しますが、oracle ソフトウェア所有者は root グループのメンバーにはなれません。root グループのメンバーになれるのは、root ユーザーのみです。

Oracle Server ユーティリティのセキュリティ

権限のないユーザーが Oracle8 実行プログラムを使用しないように保護する必要があります。Oracle8 実行プログラムを保護する方法は、環境やシングルタスク・ユーティリティを使用するかどうかによって異なります。次の方法で Oracle8 実行プログラムを保護します。

- すべてのプログラムを \$ORACLE_HOME/bin ディレクトリに置き、oracle ソフトウェア所有者に所有権を与えます。
- マシン上のすべてのユーザーが Oracle Server にアクセスできるように、すべてのユーザー・ユーティリティ（sqlplus、exp、imp）に 711 の保護を与えます。
- すべての DBA ユーティリティ（Server Manager など）に 700 の保護を与えることによって、通常、それらのユーティリティを使用できるのは oracle ソフトウェア所有者である DBA ユーザー名のみになります。

Server Manager コマンドのセキュリティ

SQL*Plus がない場合は、Server Manager を使用して SQL 問合せを実行できます。ただし、Server Manager へのアクセスを割り当てるには注意が必要です。システム権限が与えられた次の文にアクセスできるのは、特別なオペレーティング・システム権限を持つ oracle ソフトウェア所有者および dba グループのユーザーのみとしてください。

- STARTUP
- SHUTDOWN
- CONNECT INTERNAL

警告： システム権限が与えられた文は、正しく使用しないとデータベースを破壊する可能性があります。dba グループ以外のユーザーでも、必要なパスワードを知っていれば、INTERNAL として接続できるので注意してください。

データベース・ファイルのセキュリティ

Oracle8 のインストールに使用するユーザー ID は、データベース・ファイルの所有者である必要があります。デフォルトのユーザー ID は、oracle ソフトウェア所有者です。これらのファイルの認可を権限 0600 に設定します。これによって、所有者のみに読みおよび書き込み (rw) 権限が与えられ、グループや他のユーザーに書き込み権限は与えられません。

oracle ソフトウェア所有者は、データベース・ファイルを含むディレクトリの所有者である必要があります。セキュリティを高めるために、グループおよび他のユーザーの読み権限を取り消してください。

保護されているデータベース・ファイルにアクセスするには、Oracle プログラムでそのセット・ユーザー ID (setuid) ビットをオンにする必要があります。このビットを設定するには、次のように入力します。

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

これによって、Oracle プログラムの認可が次のように設定されます。

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

ユーザー ID の設定

Oracle Installer は自動的にユーザー ID を設定します。ユーザーの実行フィールドにある「s」は、Oracle プログラムを実行する場合に、そのプログラムを起動した実際のユーザー ID に関係なく、そのプログラムの事実上のユーザー ID は、oracle になることを示しています。

ネットワーク・セキュリティ

ネットワーク上でのパスワードの使用

ネットワーク上のリモート・ユーザーは、クリア・テキストまたは暗号化テキストを使用してパスワードを入力できます。クリア・テキストを使用すると、許可されていないユーザー

にパスワードを知られてしまい、セキュリティが失われます。Oracle Net8 では、暗号化パスワードが使用できます。

ネットワークでの DBA 権限

ネットワークで DBA 権限を制御するには、次のいずれかの設定を行います。

- NCR MP-RAS および Solaris Intel では、`/var/opt/oracle/listener.ora` ファイルで、リモート DBA アクセスを `denied` (拒否) に設定します。
- DG/UX Intel では、`/etc/listener.ora` ファイルで、リモート DBA アクセスを `denied` (拒否) に設定します。
- DBA 権限の `orapwd` で特別なパスワードを設定します。

自動 (ops\$) ログイン

Oracle8 では、ネットワークでの自動ログイン (オペレーティング・システムで許可されたログイン) をサポートしています。

UNIX では、ドル記号 (\$) が環境変数の始まりを示します。そのため、コマンド行またはスクリプトの中で、オペレーティング・システムによって許可された (ops\$) ログインを指定する場合、バックスラッシュ (\) で \$ をエスケープします。たとえば、リモートにログインする場合、ユーザー ID `scott` には、`ops$\scott` を指定する必要があります。

root ユーザー ID では、自動ログインできません。

注意： PC、Apple Macintosh、および OS/2 ユーザーが自動ログインすることはお薦めしません。だれでも Oracle 構成ファイルを編集したり、そのユーザー ID を変更したりできます。それらのシステムのユーザーがネットワークでログインする場合は、セキュリティ上の理由から `listener.ora` ファイルの `ops$` ログインを使用禁止にしてください。

Oracle Net8 での自動ログイン

Oracle Net8 は、自動ログインおよびリモート DBA ログインを制御しません。自動ログインおよびリモート DBA ログインは Oracle8 Server によって制御され、`initsid.ora` ファイルのパラメータを使用して構成されます。自動ログインはサポートされていますが、デフォルトでは使用禁止になっています。使用可能にするには、`REMOTE_OS_AUTHENT` 初期化パラメータを `TRUE` に設定し、データベースを起動します。

Oracle がそれらのログインを制御するため、root への `setuid` として Oracle Net8 リスナーを実行する必要はありません。

参照： Oracle Net8 の構成の詳細は、第 6 章を参照してください。

Oracle Net8 での自動ログインを実行するには、`/etc/passwd` ファイルに `daemon` というユーザーを作成します。この `daemon` ユーザーは、どのローカル・データベースにも `ops$` アカウントを持ってはいけません。また、どの DBA グループにも属してはいけません。つまり、外部ユーザーがローカル・データベースに侵入できる `ops$daemon` アカウントを持ってはいけません。

DBA グループ ID のキーワード

表 2-10 に、リモート・ログインを使用可能にして制御するためにファイルで使用するキーワードを示します。このファイルは、DG/UX Intel では `/etc/listener.ora` ファイル、NCR MP-RAS、SCO UnixWare および Solaris Intel では `/var/opt/oracle/listener.ora` ファイルです。

表 2-10 リモート・ログインを制御するキーワード

DBA_GROUP	名前が、リスナーのサービス対象となるすべてのインスタンスの定数の場合に使用します。
DBA_GROUP_sid	2 つ以上の <code>\$ORACLE_HOME</code> がリスナーのサービス対象で、そのグループ ID が異なる場合、 <code>ORACLE_SID</code> ごとに使用します。
OPS_DOLLAR_LOGIN_ALLOWED OPS_DOLLAR_LOGIN_DENIED	リモート・ログインを制御します。デフォルトは、 <code>OPS_DOLLAR_LOGIN_DENIED</code> です。
REMOTE_DBA_OPS_ALLOWED REMOTE_DBA_OPS_DENIED	リモート DBA アクセスを制御します。デフォルトは、 <code>REMOTE_DBA_OPS_DENIED</code> です。

アクセスするデータベースの DBA グループ ID がデフォルト名 (`dba`) でない場合、デフォルト以外の名前を指定できます。

リモート・ログインおよびリモート DBA のパラメータには、ネットワーク上のデータベースの個々の `ORACLE_SID` に設定するか、またはすべての `sid` を一度に設定します。たとえば、次の文はいずれも有効です。

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

`sid` にどちらの権限が割り当てられているかを調べるには、次のように入力します。

```
$ lsnrctl status
```

確認の順番

システムは、リモート・ログインのパラメータを次の順に確認します。

1. アクセスを拒否するパラメータ
2. アクセスを許可するパラメータ
3. デフォルト値（拒否）

これらの権限は、Oracle Net8 リスナーの対象となるシャドウ・プロセスのユーザー ID とグループ ID を操作することによって実行されます。次に例を示します。

- 指定したインスタンスの OPS_DOLLAR_LOGIN_DENIED が TRUE に設定されている場合、または、クライアント側のオペレーティング・システムによって通知されたユーザー ID のアカウントがデータベース・ホスト・マシンにない場合、そのユーザー ID およびグループ ID は daemon 項目の下の /etc/passwd ファイルにあります。
- 指定した ORACLE_SID の OPS_DOLLAR_LOGIN_ALLOWED および REMOTE_DBA_OPS_ALLOWED が TRUE で、クライアントのオペレーティング・システムによって通知されたユーザー ID のアカウントがこのシステムにない場合、そのユーザー ID およびグループ ID はこのユーザー ID の /etc/passwd にあります。
- 指定した ORACLE_SID の OPS_DOLLAR_LOGIN_ALLOWED が TRUE で、REMOTE_DBA_OPS_ALLOWED が FALSE の場合、そのユーザー ID に DBA 権限が与えられていれば、そのプロセスのユーザー ID およびグループ ID は daemon になります。それ以外の場合、プロセスのユーザー ID およびグループ ID は、このユーザーのユーザー ID およびグループ ID になります。

注意： デフォルトでは、REMOTE_DBA_OPS_ALLOWED は FALSE です。この値は変更しないでください。このパラメータを FALSE に設定すると、DBA 権限を持つユーザーは、オペレーティング・システムで認可されたログインをネットワークで実行できなくなります。ただし、通常のネットワーク・ログイン（パスワードの入力が必要なネットワーク・ログイン）は実行できます。

セキュリティとリモート・パスワード

オペレーティング・システム・アカウントがなくても、パーソナル・コンピュータなどのリモート・マシンからデータベースにアクセスしたり、データベースを管理したりできます。ユーザー認証は、orapwd ユーティリティで作成および管理される Oracle8 パスワード・ファイルを使用して行われます。オペレーティング・システム・アカウントをサポートするシステムでは、パスワード・ファイル検証も行えます。

ローカル・パスワード・ファイルは \$ORACLE_HOME/dbs ディレクトリにあり、1 つのデータベースのユーザー名およびパスワード情報が入っています。1 つのマシンに複数の \$ORACLE_HOME ディレクトリがある場合、それぞれにパスワード・ファイルがあります。

orapwd の実行

orapwd ユーティリティは \$ORACLE_HOME/bin にあり、oracle ソフトウェア所有者が実行します。orapwd を実行するには、次のように入力します。

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

表 2-11 に、orapwd の構文を示します。

表 2-11 orapwd の実行構文

file	パスワード情報が書き込まれているファイル名。ファイル名は orapwsid で、フルパス名を指定します。その内容は暗号化されていて、ユーザーには読めません。このパラメータの入力は必須です。
password	INTERNAL および SYS 用に選択する初期パスワードです。このパスワードは、データベースの作成後に ALTER USER 文を使用して変更できます。このパラメータの入力は必須です。
entries	データベースに SYSDBA または SYSOPER として接続できるユーザーの最大数です。このパスワード・ファイルを EXCLUSIVE にする必要がある場合のみ、このパラメータの入力は必須です。

注意： ユーザーの最大数を増やす場合は、新しいパスワード・ファイルを作成しなくてはなりません。そのため、*max_users* は、必要な数より多く設定してください。

orapwd の例

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

参照： 詳細は、『Oracle8 Server 管理者ガイド』を参照してください。

複数のデータベース用の共有パスワード・ファイル

デフォルトのパスワード・ファイル /dbs/orapwd は、複数のデータベースに対して初期化パラメータ REMOTE_LOGIN_PASSWORDFILE に SHARED を設定する場合に使用されます。複数のデータベースに対する *sid* 固有のパスワード・ファイルはありません。

リモート PC からデータベースへのアクセス

Oracle8 パスワード・ファイルがあれば、ネットワーク上の PC ユーザーは、このデータベースに INTERNAL としてアクセスできます。権限がないユーザーの場合は、このデータベースを使用する Oracle アプリケーションを起動することによって、データベースにアクセスできます。権限のあるユーザーがデータベースに対して DBA 機能を実行する場合は、PC から、Server Manager コマンドに dba ユーザーのパスワードを追加して入力します。次に例を示します。

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

OPERATOR として接続するには、同じ Server Manager コマンドに OPERATOR パスワードを追加して使用します。

リモート認証

表 2-12 に、安全性の低いプロトコルを使用したリモート接続の動作を制御する `initnsid.ora` ファイルのパラメータを示します。

表 2-12 リモート接続を制御するパラメータ

REMOTE_OS_AUTHENT	ops\$ 接続を使用可能または使用禁止にします。
OS_AUTHENT_PREFIX	ops\$ アカウントが使用します。
REMOTE_OS_ROLES	リモート接続でロールを使用可能または使用禁止にします。

注意： REMOTE_OS_AUTHENT を TRUE に設定すると、リモート・マシンの dba グループのメンバーであるユーザーは、パスワードなしで INTERNAL として接続できます。

停止時のエラー・メッセージ

Oracle インスタンスの停止時、インスタンスに接続しているクライアントでは、SQL 処理に関して次のいずれかのエラー・メッセージが表示されます。

```
ORA-03113: 通信チャンネルでファイルの終わりが検出されました。  
ORA-12571: TNS: パケットの書込み機に障害が発生しました。
```

ログイン・ホーム・ディレクトリの管理

ログイン・ホーム・ディレクトリを参照するプログラムを変更せずにそのディレクトリを追加または移動するには、次のようにします。

- 明示的なパス名は、そのパス名を保存するファイル（`/etc/passwd`、`/var/opt/oracle/oratab` など）で参照します。
- グループのメンバーは、`/etc/group` ファイルで参照します。

次のいずれかの方法でユーザーのホーム・ディレクトリを参照できるため、主要な参照ファイル以外にパス名を記録しておく必要はありません。

- C シェルおよび Korn シェルのユーザーは、`~login` を使用してユーザーのホーム・ディレクトリを参照できます。
- Bourne シェルのユーザーは、ユーザーのホーム・ディレクトリを参照するための簡単なプログラムを作成することができます。この項で示すサンプル `lhd` スクリプトを参照してください。

同じように、グループのメンバーも `/etc/group` で参照できます。この項で示すサンプル `grp` スクリプトを参照してください。

注意： 共通の目的で使用するローカルなユーティリティは、`/usr/local/bin` ディレクトリに置いてください。

サンプル `lhd` スクリプト

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

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

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

サンプル grpx スクリプト

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

例 2-1 lhd スクリプトおよび grpx スクリプトの例

次に、管理者が、.profile ファイルの基本的な構成をグループの各メンバーのホーム・ディレクトリに反映させる例を示します。clerk グループのメンバー・リストを変更しても、コードを修正する必要はありません。

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

デモンストレーションの作成と実行

PL/SQL デモンストレーションのロード

PL/SQL には、ロードできるサンプル・プログラムが多数あります。デモンストレーション・ファイルおよびメッセージ・ファイルは、rdbms ディレクトリにあります。Oracle8 Server を起動してマウントした状態で、次の操作を行ってください。

1. 次のように入力して、Server Manager を起動し、scott/tiger (ユーザー / パスワード) で接続します。

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


2. 次のように入力して、デモンストレーションをロードするために、Server Manager から `exampbld.sql` を起動します。

```
SVRMGR> @exampbld
```

注意： デモンストレーションの作成は、権限がある Oracle アカウントで行ってください。また、デモンストレーションの実行も、同じアカウントで行ってください。

PL/SQL デモンストレーションの実行

表 2-13 に、カーネルのデモンストレーションを示します。

表 2-13 カーネルのデモンストレーション

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

表 2-14 に、プリコンパイラのデモンストレーションを示します。

表 2-14 プリコンパイラのデモンストレーション

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

カーネル PL/SQL デモンストレーションを実行するには、そのデモンストレーションを作成するときに使用したユーザー / パスワードで SQL*Plus を起動し、カーネルに接続します。デモンストレーション名の前に @ マークまたは `start` を入力し、デモンストレーションを起動します。たとえば、`examp1` のデモンストレーションを起動するには、次のように入力します。

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

プリコンパイラ PL/SQL デモンストレーションを作成するには、次のように入力します。

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

1つのデモンストレーションを作成する場合は、make コマンドにそのデモンストレーション名を引数として指定します。たとえば、`examp9.pc` 実行プログラムを作成する場合、次のように入力します。

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

現在のシェルから `examp9` デモンストレーションを起動するには、次のように入力します。

```
$ examp9
```

`extproc` デモンストレーションを実行するには、まず、ファイル `tnsnames.ora` に次の行を追加します。

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

次に、ファイル `listener.ora` に次の行を追加します。

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

現在のシェルで次のように入力します。

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

最後に、テストを実行します。

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

SQL*Loader のデモンストレーション

SQL*Loader のデモンストレーションには、次の要件があります。

- ユーザー `scott/tiger` に `CONNECT` 権限と `RESOURCE` 権限があること
- 空の `EMP` 表および `DEPT` 表があること

デモンストレーションを作成して実行するには、次の手順に従います。

1. `Server Manager` を起動し、`scott/tiger` (ユーザー / パスワード) でデータベースに接続します (ライン・モード)。
2. 実行するデモンストレーションの `ulcasen.sql` を実行します。
3. 次のように入力して、`scott/tiger` でコマンド行からデモンストレーションを起動します。

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

`scott/tiger` で、次の順に SQL*Loader のデモンストレーションを実行します。

- `ulcase1`: 前述の手順 1 ~ 3 を実行します。
- `ulcase3`: 前述の手順 1 ~ 3 を実行します。
- `ulcase4`: 前述の手順 1 ~ 3 を実行します。
- `ulcase5`: 前述の手順 1 ~ 3 を実行します。
- `ulcase2`: `scott/tiger` で `ulcase1.sql` を実行し、次のように入力します。

```
$ sqlldr scott/tiger ulcase2
```

- `ulcase6`: `scott/tiger` で `ulcase6.sql` スクリプトを実行し、コマンド行で次のように入力します。

```
$ sqlldr scott/tiger ulcase1 DIRECT=true
```

- `ulcase7`: `scott/tiger` で `ulcase6.sql` スクリプトを実行し、コマンド行に次のように入力します。

```
$ sqlldr scott/tiger ulcase7
```

SQL*Loader の管理

Oracle8 Server には、SQL*Loader 機能が組み込まれています。デモンストレーション・ファイルおよびメッセージ・ファイルは、`rdbms` ディレクトリにあります。

ファイル処理オプション

SQL*Loader リリース 1.1 の制御ファイルには、次のファイル処理オプション文字列が追加されています。デフォルトは `str` で、引数は指定されていません。

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

`str` デフォルトです。改行文字で終了するレコードを、1 度に 1 レコードずつ読み込むストリームを指定します。

`fix` ファイルが n バイトの固定長レコードで構成されていることを示します。 n は整数です。

`var` ファイルが n バイトの可変長レコードで構成されていることを示します。 n は整数です。レコードの最初の 5 文字を指定します。

ファイル処理オプションを選択しないと、情報はデフォルトでレコード・ストリームとして処理されます。`fix` モードではレコード終了記号を走査する必要がないため、デフォルトの `str` モードより高いパフォーマンスが得られます。

固定長レコードでの改行

各レコードが改行で終了する固定長レコードを含むファイルを読むために `fix` オプションを使用する場合、レコード長に改行文字の長さ（1 文字）を含めて、SQL*Loader に指定してください。

次に例を示します。

```
AAA[ 改行 ]  
BBB[ 改行 ]  
CCC[ 改行 ]
```

改行文字も含めるため、`fix 3` ではなく `fix 4` を指定します。

固定長レコードのファイルの最後のレコードが改行文字で終了しない場合は、その他のレコードも改行文字で終了しないでください。同様に、最後のレコードを改行文字で終了する場合は、すべてのレコードを改行文字で終了してください。

警告： `vi` などの特定のテキスト・エディタを使用すると、ファイルの最後のレコードは自動的に改行文字で終了します。この場合、ファイル内のその他のレコードが改行文字で終了していないと、不整合が発生します。

改行文字の削除

改行文字をロードしないで、固定長レコードから改行文字を削除するには、制御ファイルの `position(x:y)` ファンクションを使用します。制御ファイルに次のように入力します。

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

これによって、固定長レコードの 4 桁目にあるために、改行文字は削除されます。

Oracle Security Server

参照： 詳細は、『Oracle Security Server ガイド』を参照してください。

Oracle8 Server の SQL リファレンス

CREATE CONTROLFILE パラメータ

[表 2-15](#) を参照して、データベースの制御ファイルのサイズを判断します。

表 2-15 制御ファイルのサイズの判断

パラメータ	デフォルト値	最大値
MAXDATAFILES	30	65534
MAXINSTANCES	1	63
MAXLOGFILES	16	255
MAXLOGMEMBERS	2	5
MAXLOGHISTORY	100	65534

トレース・ファイルとアラート・ファイルの使用

ここでは、操作上の問題を診断して解決するために Oracle Server が作成するトレース・ファイル（またはダンプ・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・ファイルの名前

トレース・ファイル名の構文は、`processname_sid_unixpid.trc` です。[表 2-16](#) に、詳細を示します。

表 2-16 トレース・ファイル名の構文

<i>processname</i>	トレース・ファイルの作成元である Oracle8 を示す 3 ~ 4 文字のプロセス名（たとえば、PMON、DBWR、ORA、RECO など）
<i>sid</i>	インスタンスのシステム識別子
<i>unixpid</i>	UNIX プロセスの ID 番号
<i>.trc</i>	すべてのトレース・ファイル名に付くファイル名の拡張子

アラート・ファイル

`alert_sid.log` ファイルは、データベースと関連付けられており、`init_sid.ora` ファイルの `BACKGROUND_DUMP_DEST` パラメータに指定されているディレクトリにあります。デフォルト値は、`$ORACLE_HOME/rdbms/log` です。

RAW デバイス

RAW デバイスの欠点（NCR MP-RAS、SCO UnixWare および Solaris Intel の場合）

Intel UNIX で RAW デバイスを使用する場合、次のような欠点があります。

注意： DG/UX Intel では、このような欠点はありません。

- 1MB より大きな表（もう 1 つのディスクのパーティションなど）をエクスポートするときに発生する可能性がある問題を、ULIMIT では解決できない場合があります。
- RAW デバイスおよびオペレーティング・システムのファイルが 1 つの Oracle8 データベース内に混在している場合、オペレーティング・システムのファイルも ULIMIT パラメータの値の範囲内である必要があります。
- ディスクに Oracle 配布の内容を読み込むときに発生する可能性がある問題を、ULIMIT では解決できない場合があります。
- 小規模クライアント・システムでは、通常、十分な大きさの RAW デバイスのパーティションを使用できません。ディスク・パーティションが半端なサイズになり、優れたデータベース・アーキテクチャに適さなくなります。
- 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すとパフォーマンスが向上する場合、I/O アクティビティの少ないドライブには受け入れ可能なサイズのセクションがないことがよくあります。UNIX の利点であるデータ・ファイルの移動は、RAW デバイスでは欠点になります。
- RAW デバイス環境では、表領域を増やすことは困難です。初期構成時に、すべての RAW パーティションがデータ・ファイルに割り当てられる場合があるため、通常の表領域の拡張に対応する RAW 記憶領域が残らないことがあります。

RAW デバイスを使用する場合の基準

RAW デバイスを使用するかどうかを判断する場合、次の要素を考慮する必要があります。

- Oracle Parallel Server（OPS）のインストレーション
- RAW ディスク・パーティションの可用性

Oracle Parallel Server インストレーション (NCR MP-RAS の場合)

OPS の各インスタンスには、それぞれ個別のログ・ファイルがあります。そのため、表領域および制御ファイルに必要なパーティションの他に、各インスタンスのログ・ファイル用に、少なくとも 3 つのパーティションが必要になります。すべてのファイルは、Intel UNIX クラスタのすべてのノードで共有できるディスク上にある必要があります。

UNIX クラスタでは、クラスタのすべてのノード間の共有ファイル・システムにアクセスできません。そのため、データベースと関連付けられているすべてのファイルは RAW デバイス上に作成する必要があります。

RAW ディスク・パーティションの可用性

ご使用のサイトに、少なくとも Oracle 表領域と同数の RAW ディスク・パーティションがある場合、Oracle ファイルに RAW デバイスを使用してください。

RAW ディスク・パーティションがすでにフォーマットされている場合、表領域のサイズをパーティションのサイズにできるだけ合わせて、無駄な領域をなくします。

RAW デバイスを使用する場合のガイドライン

RAW ディスク・パーティションを作成する場合は、次のガイドラインに従ってください。

- 各インスタンスのログ・ファイル用に 3 つのパーティション
- データ・ファイル SYSTEM、ROLLBACK、TEMP、USERS、TOOLS のそれぞれに 1 つのパーティション
- データ・ファイル用に少なくとも 3 つのパーティション

構成計画

論理ディスクは複数のディスク間で移動できるため、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応じて論理ディスクを作成することができます。適切な論理ディスク・サイズを割り当てるために、ディスク・ドライブを再フォーマットする必要はありません。

動的パフォーマンスのチューニング

負荷が集中するディスクのデータを負荷の低いドライブに移すことによって、データベースがオンラインにある場合のディスク・パフォーマンスを最適化できます。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、この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置換え

論理ボリュームはミラー化できるので、ミラー化によってデータ損害からデータを保護してください。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置き換える機能を提供している場合もあります。

注意： Oracle Parallel Server の場合、論理ボリュームは、UNIX クラスタにある複数のマシンで共有できるドライブだけではなく、1 台の UNIX マシンに関連付けられているドライブでも使用できます。複数のマシンに関連付けられているドライブで論理ボリュームを使用する場合、Oracle Parallel Server と関連付けられているすべてのファイルを、これらの共有論理ボリュームに置けるようになります。

RAW デバイスの設定

警告： RAW デバイスの設定は、経験のあるシステム管理者、またはご使用のマシンについて詳しい知識のある方が行ってください。

ご使用のシステムで RAW デバイスを設定するには、次の手順に従います。

1. (この手順は、Oracle Parallel Server の場合にのみ行います。) 追加するパーティションが共有ディスク上にあることを確認します。
2. 空きディスク・パーティションの名前を決定します。

空きパーティションとは、Intel UNIX ファイル・システムとして使用されていないパーティションのことです。このパーティションは、次の制限事項に従っています。

- `/etc/mount` コマンドの実行時にリストに含まれない。
- スワップ・デバイスとして使用されない。
- スワップ・パーティションとオーバーラップしない。
- その他の Intel UNIX アプリケーション (Oracle の他のインスタンスなど) が使用していない。
- Intel UNIX のファイル・システムとオーバーラップしない。
- ファイル・システムがすでに使用している領域を使用しない。

パーティションが使用可能かどうかを調べるには、デバイス上のパーティションの開始位置とパーティションのサイズを確認し、空き領域を調べます。パーティションによっては、現在マウントされていないファイル・システム、および `/etc/mount` 出力のリストに含まれないファイル・システムがある場合があるので注意してください。

注意： パーティションをシリンダ 0 で開始しないでください。

3. RAW デバイスを設定して、Oracle8 Server で使用できるようにします。

ディスクのパーティションを切っていることを確認した後、設定を始めます。パーティションを切っていない場合は、次のいずれかを行ってください。

- NCR MP-RAS および Solaris Intel の場合、`format` ユーティリティを使用してパーティションを切ります。
- DG/UX Intel の場合、`sysadm` ユーティリティを使用してパーティションを切ります。
- SCO UnixWare の場合、`fdisk` および `edvtoc` ユーティリティを使用してパーティションを切ります。

次に、`oracle` ソフトウェア所有者がこのパーティションを所有するようにします。`chown` コマンドを使用して、必要に応じてブロックおよびデバイスの文字ファイルの所有権を変更します。

NCR MP-RAS および Solaris Intel の場合、たとえば、次のように入力します。

```
$ chown oracle /dev/dsk/c120t0d0s0
$ chown oracle /dev/rdisk/c120t0d0s0
```

DG/UX Intel の場合、たとえば、次のように入力します。

```
$ chown oracle /dev/rdisk/file1
```

SCO UnixWare の場合、たとえば、次のように入力します。

```
$ chown oracle /dev/rdisk/c0b0t0d0s1
```

`chmod` を使用して、`oracle` ソフトウェア所有者のみがこのパーティションにアクセスできるようにします。

NCR MP-RAS および Solaris Intel の場合、たとえば、次のように入力します。

```
$ chmod 600 /dev/dsk/c120t0d0s0
$ chmod 600 /dev/rdisk/c120t0d0s0
```

DG/UX Intel の場合、たとえば、次のように入力します。

```
$ chmod 600 /dev/rdsk/file1
```

SCO UnixWare の場合、たとえば、次のように入力します。

```
$ chmod 600 /dev/rdsk/c0b0t0d0s1
```

4. 必要な RAW デバイスへのシンボリック・リンクを作成します。

NCR MP-RAS および Solaris Intel の場合、たとえば、次のように入力します。

```
$ ln -s /dev/rdsk/c120t0d0s0 /oracle_data/datafile.dbf
```

DG/UX Intel の場合、たとえば、次のように入力します。

```
$ ln -s /dev/rdsk/file1 /oracle_data/datafile.dbf
```

SCO UnixWare の場合、たとえば、次のように入力します。

```
$ ln -s /dev/rdsk/c0b0t0d0s1 /oracle_data/datafile.dbf
```

ブロック専用デバイスではなく、文字専用デバイスを使用していることを確認してください。確認したら、次のコマンドを実行します。

```
$ ls -Ll datafile
```

これによって、次の値が戻されます。

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

前述のコマンドの `L` はシンボリック・リンクを、`l` は長いリストを表します。

注意： このシンボリック・リンクは、Parallel Server のノードごとに設定されている必要があります。2 つのシンボリック・リンクが、同じ RAW デバイスを示すことがないようにしてください。

5. 新しいデータベースに新しいパーティションを作成または追加します。

Server Manager で SQL 文 CREATE DATABASE を実行し、指定した RAW パーティションを使用してデータベースを作成します。

注意： RAW パーティションに作成した 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 Oracle ブロック・サイズ × 2 以上小さくする必要があります。

例 2-2

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

既存の Oracle データベースの表領域にパーティションを追加する場合は、次のように入力します。

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

同じ手順で、REDO ファイルに RAW デバイスを設定できます。

動的共有メモリー（DSHM）の使用（SCO UnixWare の場合）

動的共有メモリー（DSHM）オプションを使用する場合は、ご使用のシステムに UnixWare カーネルとともに出荷される DSHM パッケージをインストールおよび構成する必要があります。

- pkgadd を使用して DSHM パッケージを追加します。
- /stand/boot ファイルに、enable_4gb_mem=Y を追加します。
- /etc/conf/sdevice.d/pse ファイルの N を Y に変更して、pse ドライバを使用可能にします。
- idtune を使用してカーネル・パラメータ DEDICATED_MEMORY を変更します。
詳細は、『Oracle8 for Intel UNIX (DG/UX Intel, NCR MP-RAS, SCO UnixWare, Solaris Intel) インストレーション・ガイド』の第 2 章の「SCO UnixWare での UNIX カーネル・パラメータ」の項を参照してください。

注意： 詳細は、SCO UnixWare のドキュメントを参照してください。

- Oracle バイナリの権限を変更して、plock を可能にします。次に例を示します。

```
$ su
$ filepriv -f plock $ORACLE_HOME/bin/oracle
```

Intel UNIX での Oracle8 のチューニング

- チューニングの重要性
- システム・パフォーマンス監視用のツール
- Intel UNIX のツール
- Oracle8 Server のチューニング手順
- アプリケーション設計のチューニング
- データ・アクセスのチューニング
- 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
- CPU の使用状況のチューニング
- Oracle リソース競合のチューニング
- Oracle Parallel Server (OPS) のリソース競合のチューニング (DG/UX Intel および NCR MP-RAS の場合)
- Intel UNIX バッファ・キャッシュ・サイズのチューニング

チューニングの重要性

Oracle8 Server は、高度に最適化できるソフトウェア製品です。Oracle for UNIX では、アプリケーションの有効範囲が広がり複雑になったので、チューニングを頻繁に行うことによって、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を防ぎます。

この章の説明は、シングル・プロセッサ・システムを対象としていますが、パフォーマンス・チューニングに関するヒントの多くは、Oracle Parallel Server オプションを使用する場合にも適用できます。

注意： ここで説明するパフォーマンスのチューニングに関するヒントは、オラクル社がインプリメントした Oracle8 の本番データベースで実証済みです。ほとんどのヒントに対して、パフォーマンスの向上率を概算で示します。

注意： 今回のリリースの SCO UnixWare および Solaris Intel では、Oracle Parallel Server オプションはサポートされていません。

システムをチューニングする前に

システムのチューニングを始める前に、後述の項の Intel UNIX のツールを使用して、通常の動作を監視します。

Oracle Parallel Server の場合、動的パフォーマンスのチューニング・ビューは、システムの監視およびチューニングで使用できます。

参照： 詳細は、『Oracle8 Parallel Server 概要および管理』および『Oracle8 Server チューニング』を参照してください。

パフォーマンス・ボトルネックのタイプ

メモリー

メモリーの競合は、プロセスが使用できるメモリーが不足した場合に発生します。このメモリー不足を解消するために、システムは、メモリーとディスク間でプロセスのページングとスワッピングを行います。

ディスク I/O

ディスク I/O の競合は、メモリー管理がうまく行われていない（そのためページングとスワッピングが発生する）ことや、表領域とファイルのディスクへの分散が適切でないことが原因です。I/O 負荷は、すべてのディスクに均等に分散する必要があります。

CPU

CPU でも、プロセスの競合が発生する場合があります。ほとんどの場合、UNIX カーネルによって CPU が効果的に割り当てられますが、プロセスが多数あると、それらが CPU サイクルを競合します。システムに複数の CPU がある場合（マルチ・プロセッサ環境）は、さまざまな CPU で異なるレベルの競合が発生することがあります。

Oracle リソース

競合は、ロックやラッチなどの Oracle リソースでもよく起こります。

システム・パフォーマンス監視用のツール

TKPROF および EXPLAIN PLAN ユーティリティ

TKPROF および EXPLAIN PLAN ユーティリティは、SQL トレース機能の出力ファイルをフォーマットします。TKPROF を使用して、次のことを行います。

- OLTP のアプリケーションのトランザクションごとに必要な物理 I/O の数を推定します。
- パフォーマンスが低い SQL 文を識別およびチューニングします。

EXPLAIN PLAN を使用して、問合せにどの索引を使用するかを決めます。索引を正しく使用することで、パフォーマンスが向上します。

参照： SQL トレース、TKPROF および EXPLAIN PLAN ユーティリティの詳細は、『Oracle8 Server チューニング』を参照してください。

SQL スクリプト utlbstat および utlestat

SQL スクリプト utlbstat および utlestat は、Oracle データベースのパフォーマンスを監視し、共有グローバル領域（SGA）データ構造をチューニングします。これらのスクリプトは、データベースのパフォーマンス統計のスナップショットにも使用されます。スクリプトは、\$ORACLE_HOME/rdbms/admin ディレクトリに保存されています。

これらのスクリプトは、問題のある領域を識別するための表を作成します。たとえば、ラッチの統計表にはラッチの競合が表示されます。その情報は、SPIN_COUNT パラメータのチューニングに使用できます。

これらのスクリプトを実行する前に、TIMED_STATISTICS 初期化パラメータを設定します。データベースを安定させてから、これらのスクリプトを実行します。

次のように入力して、統計を初期化します。

```
$ cd $ORACLE_HOME/rdbms/admin
$ svrmgrl
SVRMGR> CONNECT INTERNAL
SVRMGR> CONNECT SYS/password
SVRMGR> @utlbstat
```

utlbstat ユーティリティは、SYS アカウントで一連の表およびビューを作成します。これらの表には、パフォーマンス・テスト開始時のシステム状態のスナップショットが表示されます。これを表 3-1 に示します。

表 3-1 パフォーマンス・テスト開始時に utlbstat によって作成されるシステム状態表

ビューおよび表の名前	説明
STATS\$BEGIN_STATS	V\$SYSSTAT の一般的な統計
STATS\$FILE_VIEW	ファイル I/O 統計のビュー
STATS\$BEGIN_FILE	STATS\$FILE_VIEW のファイル I/O 統計の表
STATS\$BEGIN_LATCH	V\$LATCH のラッチの統計
STATS\$BEGIN_ROLL	V\$ROLLSTAT のロールバック・セグメントの統計
STATS\$BEGIN_KQRST	X\$KQRST のディクショナリ・キャッシュの統計
STATS\$DATES	開始日を示す表

統計を終了するには、utlestat スクリプトを実行します。utlbstat と同じように、utlestat によって、SYS アカウントで一連の表およびビューが作成されます。これらの表には、パフォーマンス・テスト終了時のシステム状態のスナップショットが表示されます。これを表 3-2 に示します。

表 3-2 パフォーマンス・テスト終了時に utlestat によって作成されるシステム状態表

ビューおよび表の名前	説明
STAT\$SEND_STATS	V\$SYSSTAT の一般的な統計
STAT\$SEND_FILE	STAT\$FILE ビューのファイル I/O の統計表
STAT\$SEND_LATCH	V\$LATCH のラッチの統計
STAT\$SEND_ROLL	V\$ROLLSTAT のロールバック・セグメントの統計
STAT\$SEND_KQRST	X\$KQRST のディクショナリ・キャッシュの統計

また、utlestat ユーティリティによって、開始時と終了時のシステム状態の違いを表示する一連の表が SYS アカウントで作成されます。表 3-3 に、作成されるデータベース表を示します。

表 3-3 utlestat によって作成されるシステム状態の違いを示す表

ビューおよび表の名前	説明
STAT\$STATS	一般的なシステムの統計
STAT\$FILE	ファイル I/O の統計
STAT\$SLATCHES	ラッチの統計
STAT\$ROLL	ロールバック・セグメントの統計
STAT\$KQRST	ディクショナリ・キャッシュの統計
STAT\$DATES	終了日を示す表

utlbstat によって、表 3-3 の各表のパフォーマンス統計を示すレポートが、現在のディレクトリに作成されます。このレポートには、init.ora ファイルのパラメータのリスト、および utlbstat、utlestat を実行した日が含まれています。

V\$SYSTEM_EVENT & V\$SESSION_WAIT 表

Oracle V\$SYSTEM_EVENT 表には、カーネルのアクティビティに関する情報が表示されます。Oracle V\$SYSTEM_EVENT 表は、データベース内で問題が起きる可能性のある領域を識別するのに役立ちます。V\$SESSION_WAIT 表によって、監視したイベントの詳細がわかります。これらの表に示されるパラメータの詳細は、『Oracle8 Server 概要』および『Oracle8 Server リファレンス・マニュアル』を参照してください。

Intel UNIX のツール

Intel UNIX では、データベースのパフォーマンスを評価し、データベースの要件を決定するパフォーマンス監視ツールを提供しています。

これらのツールは、Oracle プロセスの統計だけでなく、システム全体の CPU の使用状況、割込み、スワップ、ページングおよびコンテキスト・スイッチングについての統計情報も提供します。

参照： Intel UNIX 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を参照してください。

vmstat (Solaris Intel の場合)

注意： DG/UX Intel、NCR MP-RAS および SCO UnixWare では、vmstat コマンドは使用できません。

vmstat ユーティリティは、Intel UNIX におけるプロセス、仮想メモリー、ディスク、ページングおよび CPU のアクティビティを表示します。表示内容は、コマンドで切り換えます。システム・アクティビティのサマリーを 5 秒ごとに 8 回表示する場合は、次のように入力します。

```
% vmstat -S 5 8
```

次に、vmstat コマンドの出力例を示します。

図 3-1 vmstat コマンドの出力例

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

w 列 (procs の下) は、スワップ・アウトされた (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 以外の場合は、スワッピングが起きていて、システムはメモリー不足になっています。si 列および so 列は、それぞれ 1 秒あたりのスワップ・インおよびスワップ・アウトの回数を示します。スワップ・アウト (so 列) は、常に 0 にしてください。

sar

sar コマンドは、スワッピング、ページング、ディスクおよび CPU のアクティビティを監視します。監視する内容は、コマンドで切り換えます。

ページング・アクティビティのサマリーを 10 秒ごとに 10 回表示する場合は、次のように入力します。

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

次に、DG/UX Intel、NCR MP-RAS および Solaris Intel で sar -p コマンドを実行した場合の出力例を示します。

図 3-2 sar -p コマンドの出力例

14:14:55	atch/s	pgin/s	ppgin/s	pflt/s	vflt/s	slock/s
14:15:05	0.00	0.00	0.00	0.60	1.00	0.00
14:15:15	0.00	0.00	0.00	0.10	0.60	0.00
14:15:25	0.00	0.00	0.00	0.00	0.00	0.00
14:15:35	0.00	0.00	0.00	0.00	0.00	0.00
14:15:45	0.00	0.00	0.00	0.00	0.00	0.00
14:15:55	0.00	0.00	0.00	0.00	0.00	0.00
14:16:05	0.00	0.00	0.00	0.00	0.00	0.00
14:16:15	0.00	0.00	0.00	0.00	0.00	0.00
14:16:25	0.00	0.00	0.00	0.00	0.00	0.00
14:16:35	0.00	0.00	0.00	0.00	0.00	0.00
Average	0.00	0.00	0.00	0.07	0.16	0.00

次に、SCO UnixWare で `sar -p 10 10` コマンドを実行した場合の出力例を示します。

図 3-3 sar -p 10 10 コマンドの出力例

14:01:36	atch/s	atfree/s	atmiss/s	pgin/s	ppgin/s	pflt/s	vflt/s	slock/s
14:01:46	4647.39	312.90	188.14	23.63	175.51	4245.34	367.24	0.00
14:01:56	4481.26	278.17	171.98	22.43	174.78	4164.11	218.64	0.10
14:02:06	2915.95	301.00	192.52	25.22	193.82	2598.21	207.58	0.00
14:02:16	3000.39	315.80	187.13	25.40	201.22	2638.37	231.82	0.00
14:02:26	2799.40	309.28	191.92	25.35	183.83	2446.91	231.14	0.00
14:02:36	1817.56	288.60	175.10	24.09	182.50	1503.26	207.29	0.00
14:02:46	4445.31	363.57	184.73	23.85	185.83	3350.30	285.23	671.66
14:02:56	2545.91	298.90	188.72	27.15	204.19	1209.08	250.20	1003.99
14:03:06	3069.29	292.42	165.70	21.64	172.98	1770.19	224.93	966.90
14:03:16	1377.84	280.67	178.84	23.66	187.91	70.61	216.17	1007.58
Average	3106.63	303.78	182.49	24.23	186.16	2397.89	243.71	363.77

ディスク・アクティビティのサマリーを 5 秒ごとに表示する場合は、次のように入力します。

```
$ sar -d 5 5
```

次に、SCO UnixWare で `sar -d 5 5` コマンドを実行した場合の出力例を示します。

図 3-4 sar -d 5 5 コマンドの出力例

14:08:22	device	%busy	avque	r+w/s	blks/s	await	avserv
14:08:27	sd011	5	1.9	7	95	6.8	7.9
14:08:27	sd012	0	1.0	0	1	0.0	10.0
14:08:27	sd013	74	4.3	72	2568	34.1	10.3
14:08:32	sd011	0	1.0	0	9	0.0	13.3
14:08:32	sd012	0	1.0	0	0	0.0	10.0
14:08:32	sd013	84	11.3	136	6917	63.3	6.2
Average	sd011	2	2.3	3	42	11.4	9.1
Average	sd012	0	1.0	0	0	0.0	10.0
Average	sd013	73	5.2	120	3422	25.7	6.1

iostat (Solaris Intel の場合)

注意： DG/UX Intel、NCR MP-RAS および SCO UnixWare では、
iostat コマンドは使用できません。

iostat ユーティリティは、端末およびディスクのアクティビティを表示します。iostat は、ディスクのビジー（使用中）状況を表示します。ただし、ディスク要求キューは表示されません。iostat で表示される情報は、I/O 負荷のバランスを調整する際に役立ちます。

端末およびディスクのアクティビティのサマリーを 5 秒ごとに 5 回表示する場合は、次のように入力します。

```
$ iostat 5 5
```

次に、iostat コマンドの出力例を示します。

図 3-5 iostat コマンドの出力例

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

swap (NCR MP-RAS、 Solaris Intel および SCO UnixWare の場合)

注意： DG/UX Intel では、swap ユーティリティは使用できません。
admswap コマンドを参照してください。

swap -l ユーティリティは、スワップ領域の使用状況を表示します。スワップ領域が不足すると、システムがハングアップしたり、応答時間が長くなったりする場合があります。システムが正しく動作するための十分なスワップ領域があることを確認してください。次に、swap -l コマンドの出力例を示します。

図 3-6 swap -l コマンドの出力例

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

admswap (DG/UX Intel の場合)

DG/UX Intel では、admswap -o list -bu がスワップ領域の使用状況についての情報を表示します。

図 3-7 admswap -o list -bu コマンドの出力例

スワップ領域	ブロック	使用状況
/dev/dsk/swap	2000000	Y
/dev/dsk/swap2	2000000	Y

truss (NCR MP-RAS、Solaris Intel および SCO UnixWare の場合)

注意： DG/UX Intel では、truss コマンドは使用できません。

ご使用のシステムでプロセスが停止した場合、truss コマンドを使用して、実行中のプロセスのシステム・コールおよびシグナルをトレースできます。次に、truss コマンドの出力例を示します。

図 3-8 truss -vall -p <pid> コマンドの出力例

semop(317999,0x08040E30,1)	Err#91 ERESTART
semnum=4 semop=-1 semflg=0	
sigprocmask(SIG_SETMASK,0x08040BD0,0x00000000)	=0
set=0x1FBFF077 0x00000002 0 0	
sigprocmask(SIG_SETMASK,0xBFF014B4,0x00000000)	=0
set=0xFFFFF157 0x00000003 0 0	
sigprocmask(SIG_SETMASK,0xBFF00A2C,0x00000000)	=0
set=0x1FBFF077 0x00000002 0 0	
times(0x08040B24)	=16901364
utim=80 stim=196 cutim=0 cstim=0 (HZ=100)	
sigprocmask(SIG_SETMASK,0x0BFF014B4,0x00000000)	=0
set=0xFFFFF157 0x00000003 0 0	
sigprocmask(SIG_SETMASK,0xBFF00A2C,0x00000000)	=0
set=0x1FBFF077 0x00000002 0 0	
sigprocmask(SIG_SETMASK,0xBFF014B4,0x00000000)	=0
set=0xFFFFF157 0x00000003 0 0	
setcontext(0x08E872A8)	
times(0x08040FE4)	=16901364
utim=80 stim=196 cutim=0 cstim=0 (HZ=100)	

mpstat (Solaris Intel の場合)

注意： DG/UX Intel、NCR MP-RAS および SCO UnixWare では、mpstat コマンドは使用できません。

mpstat ユーティリティは、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 つ目の表は、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 つ目以降の各表は、アクティビティの間隔を表示します。特に指定しない限り、すべての値は割合（1 秒あたりのイベント数）です。引数は、統計および反復回数（時間間隔）を示します。次に、mpstat コマンドの出力例を示します。

図 3-9 mpstat コマンドの出力例

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

Oracle8 Server のチューニング手順

Oracle Server のパフォーマンスを最大限に引き出すために必要なタスクがいくつかあります。あるチューニング操作で決めたことがその後のチューニング処理に影響する場合もあるため、チューニングに必要なタスクは、次の手順に従って行ってください。

- [アプリケーション設計のチューニング](#)
- [データ・アクセスのチューニング](#)
- [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](#)
- [CPU の使用状況のチューニング](#)
- [Oracle リソース競合のチューニング](#)
- [Oracle Parallel Server \(OPS \) のリソース競合のチューニング \(DG/UX Intel および NCR MP-RAS の場合 \)](#)

- [Intel UNIX バッファ・キャッシュ・サイズのチューニング](#)

注意：一部のチューニング・タスクは、UNIX 固有のタスクではないので、この章では説明していません。このようなタスクについては、適切な参照先を示してあります。

アプリケーション設計のチューニング

データベース設計は、アプリケーションのパフォーマンスを決める最も重要な要素です。

参照ドキュメント

データベース設計の詳細は、次のドキュメントを参照してください。

- 『Oracle8 Server アプリケーション開発者ガイド』
データベース設計に関するアドバイスおよび追加情報を記載しています。
- 『Oracle Performance Tuning』
(Peter Corrigan, Mark Gurry 著、O'Reilly & Associates, Inc.)
アプリケーションのチューニングおよびデータ・アクセスについて説明しています。

LONG データの個別保存

表内の各 LONG（および LONG RAW）列は、他の関連データとは切り離して保存します。これによって、全表走査中に SQL 文が LONG 列を走査することを防ぐことができます。問合せによって返される行が少ない場合は、必要な行の索引付き SELECT 文を実行すると効果的です。

Large Object Base (LOB)

ファイル（BFILE 型）の場合は、次のようにしてください。

- DBA のみが CREATE/DROP ANY DIRECTORY システム権限を持つ。
- DIRECTORY は、データベース・ファイルを含むパスを参照しない。
- ディレクトリおよびファイルへの読み込み権限が Oracle にある。

SQL 文の再使用

再使用できる SQL 文を書くことによって、ライブラリ・キャッシュのパフォーマンスを改善できます。類似している SQL 文は、バインド変数を使用し、バインド変数に異なる値を割り当てることによって同一になります。次に例を示します。

```
SELECT :name FROM EMP WHERE EMP_NO = 135;  
SELECT :name FROM EMP WHERE EMP_NO = 137;
```

これらの文は、次の 1 文に置き換えることができます。

```
SELECT :name FROM EMP WHERE EMP_NO = :variable;
```

複数のユーザーが同じプログラムを実行する場合（特に、ライブラリ・キャッシュが大きい場合）は、かなり多くの再使用が行われます。

データ・アクセスのチューニング

システム・メモリー、ディスク I/O または CPU をチューニングする前に、ご使用のアプリケーションが適切に設計され、適切に作成されていることを確認してください。SQL 文の最適化は、この設計の重要な要素です。

SQL スクリプト `utlbstat` および `utlestat` を使用して、データベースのパフォーマンスに関する統計のスナップショットを取ります。このデータは、アプリケーションが起動した後、実行中に収集するのが最適です。

SQL スクリプト `utlestat` は、パフォーマンスの監視に役立つ情報をレポートします。この情報は、`V$SYSSTAT`、`V$LATCH` および `V$ROLLSTAT` の各表から取得します。

参照： SQL スクリプト `utlbstat` および `utlestat` の詳細は、前述の「[システム・パフォーマンス監視用のツール](#)」を参照してください。

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、ページングおよびスワップ領域をチューニングし、使用可能なメモリーを判断します。

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てバッファ・キャッシュをチューニングすると、Oracle のパフォーマンスはかなり向上する場合があります。各システムの Oracle バッファ・サイズの最適値は、システム全体の負荷と、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

スワッピングはかなりの UNIX オーバーヘッドの原因となるため、最小限に抑える必要があります。スワッピングを調べるには、`sar -w` を使用します。Solaris Intel システムでは、`vmstat -s` でもスワッピングを調べることができます。

システムでスワッピングが行われている場合、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 (`daemon`)・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、使用可能なメモリーを増やします。
- 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： スワップ領域を追加する手順は、UNIX インプリメンテーションによって異なります。NCR MP-RAS、Solaris Intel および SCO UnixWare では、`swap -l` を使用して、使用中のスワップ領域を判断します。DG/UX Intel では、`admswap -o list -vu` を使用して、使用中のスワップ領域およびサイズを表示します。ご使用のシステムに適した手順については、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。

スワップ領域は、まず、システムの RAM の 2 ~ 4 倍に設定します。CASE、Oracle Application または Oracle Office を使用する場合は、もう少し大きい値に設定してください。スワップ領域の使用状況を見ながら、必要に応じて値を大きくしてください。

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がメモリーに常駐する必要はないので、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。少しくらい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、`sar -p` を使用します。Solaris Intel でページングを監視するには、`vmstat -s` も使用できます。`sar -p` の出力では、次の列が重要です。

- `vflt/s` は、アドレス変換ページ不在の数を示します。アドレス変換ページ不在は、プロセスが、メモリー中にないページを参照したときに発生します。
- `rc1m/s` (DG/UX Intel、NCR MP-RAS および Solaris Intel の場合) は、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 にしてください。

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。

- メモリーを増やします。
- 一部の作業を別のシステムに移します。
- カーネルで使用するメモリーを少なく設定します。

単一共有メモリー・セグメントでの SGA の確保

パフォーマンスの向上率: 0 ~ 1%

共有メモリーの構成によってパフォーマンスはわずかに向上するだけですが、十分な共有メモリーを構成しなければ、データベースを起動できません。

共有メモリーを増やすために、UNIX カーネルを構成し直さなければならない場合があります。共有メモリーに関連する UNIX カーネル・パラメータには、SHMMAX、SHMMNI および SHMSEG があります。単一共有メモリー・セグメントは、SGA を確保するだけの十分な大きさが必要です。

SGA のコンポーネント、およびそれぞれのサイズを調べるには、「Server Manager Instance/Database」を選択するか、または次の手順で SGA のサイズを見積もります。

1. DB_BLOCK_BUFFERS と DB_BLOCK_SIZE の値を掛けます。
2. 手順 1 の値と SORT_AREA_SIZE の値を足します。
3. 手順 2 の値と SHARED_POOL_SIZE の値を足します。
4. 手順 3 の値と LOG_BUFFER の値を足します。

Oracle ユーティリティ `tstshm` を使用して、既存の共有メモリー構成を評価します。このユーティリティによって、次の情報が得られます。

- プロセスによって生成および割り当てられた共有メモリーのセグメント数
- 仮想メモリーでの共有メモリーの位置
- 共有メモリーのサイズ
- 最も大きい単一セグメントのサイズ

UNIX ユーティリティ `ipcs` を使用することによっても、共有メモリーの状態を監視できます。

参照： UNIX カーネルの再構成の詳細は、『Oracle8 for Intel UNIX (DG/UX Intel, NCR MP-RAS, SCO UnixWare, Solaris Intel) インストレーション・ガイド』を参照してください。

物理メモリーへの SGA のロック

パフォーマンスの向上率 : 0 ~ 20%

SGA の主な機能は、データベース情報のキャッシングです。SGA がディスクへのページングを開始すると、キャッシングは利点ではなくオーバーヘッドになります。一部のプラットフォーム・ベンダーは、SGA を物理メモリーにロックする技術を提供しています。

SGA を物理メモリーにロックすると、Oracle のパフォーマンスが改善されることがありますが、同じシステム上の他のアプリケーションで利用できる物理メモリーが減るため、これらのアプリケーションのパフォーマンスが低下する可能性もあります。

共有メモリーのロックは、NCR MP-RAS ではサポートされていません。

DG/UX Intel では、DG/UX カーネル・パラメータ PERCENTLOCKABLE で、システムにロックできる物理 RAM のページ数を設定します。PERCENTLOCKABLE のデフォルトは、SGA 全体をロックするには小さい値です。DG/UX カーネルの PERCENTLOCKABLE パラメータが、十分な値に設定されていることを確認してください。

DG/UX Intel および Solaris Intel システムで共有メモリーをロックするには、データベース所有者が root である必要があります。

Oracle で SGA をロックするには、init.ora ファイルの LOCK_SGA パラメータの値を TRUE に設定します。

データベース・バッファ数の最適化

パフォーマンスの向上率 : 0 ~ 200%

DB_BLOCK_SIZE パラメータでも、SGA のデータベース・バッファのサイズを決定できます。DB_BLOCK_BUFFERS パラメータは、システムのパフォーマンスに最も影響するメモリー・パラメータです。

Server Manager のシステム統計モニターを使用して、ヒット率を調べます。バッファ・キャッシュのヒット率は、次の式で計算します。

$$\text{ヒット率} = \frac{\text{論理読込みの数} - \text{物理読込みの数}}{\text{論理読込みの数}}$$

この場合、論理読込みの数 = データベース・ブロック取得の数 + 固定取得の数です。

ヒット率が 60 ~ 70% 以下の場合、DB_BLOCK_BUFFERS の値を増やして、キャッシュ内のバッファの数を増やしてください。

参照： X\$KCBRBH 表を使用して実行中のシステムから収集した統計情報に基づいてバッファ数を見積もる方法については、『Oracle8 Server チューニング』を参照してください。システム統計モニターについては、『Oracle Server Manager ユーザーズ・ガイド』を参照してください。

間接データベースのバッファ（DG/UX Intel および SCO UnixWare の場合）

直接データベースのバッファを使用する場合、DG/UX Intel および SCO UnixWare がサポートする SGA のサイズの最大値は、3GB です。ただし、間接データベースのバッファを使用すると、3GB 以上の SGA を持つことができます。間接データベースのバッファが使用できるようになると、必要に応じてバッファのマッピングおよびアンマッピングのオーバーヘッドを減らすことができます。バッファ・キャッシュのヒット率を上げるために、4GB 以上の物理メモリーを使用して、この機能をシステム上で使用可能にすることをお薦めします。システム上でのメモリー・リソースを最適に活用するには、DB_BLOCK_SIZE パラメータを 4KB の倍数に設定します。

間接データベースのバッファを使用可能にするには、init.ora ファイルのパラメータ USE_INDIRECT_DATA_BUFFERS を TRUE に設定します。

次の制限事項があります。

1. 間接データベースのバッファは、Oracle Parallel Server オプションではサポートされていません。
2. ご使用のデータベースで LOB（大きいオブジェクト）を保存する場合、この機能は使用可能にしないでください。
3. Oracle SGA は、物理メモリーにロックされている必要があります。詳細は、「[物理メモリーへの SGA のロック](#)」の項を参照してください。
4. この機能は、Oracle Enterprise Edition のみでサポートされています。

REDO バッファ数の最適化

REDO ログ領域の統計は、ユーザー・プロセスが REDO バッファ内の空き領域を待つ回数を表します。Server Manager のシステム統計モニターを使用して、REDO バッファの監視を表示します。

REDO ログ領域が要求する Total 列の値は、0（ゼロ）に近い数値、または少なくとも増加しない数値にする必要があります。0（ゼロ）以外の値は、プロセスがバッファの空き領域を待機中であることを示します。この場合、REDO ログ・バッファのサイズを 5% ずつ増やすことを検討してください。

REDO ログ・バッファのサイズは、init_{sid}.ora ファイルの LOG_BUFFER パラメータによって決まります。このパラメータの値は、バイト単位で表します。

共有プール・サイズの最適化

init_{sid}.ora ファイルの SHARED_POOL_SIZE パラメータには、バイト単位で共有プールのサイズを設定します。変更された最低使用頻度アルゴリズムは、データ・ディクショナリ・キャッシュのエントリを優先します。つまり、ライブラリ・キャッシュのチューニングによって、十分な量のメモリーをデータ・ディクショナリで使えるようになります。

V\$SGASTAT 表を使用して、特に空き領域を確認しながら共有プールを監視します。次に、V\$SGASTAT 問合せの例を示します。

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

共有プールに大きすぎる値が設定されることがよくあります。空きメモリー領域がこの例と同じ大きさの場合は、共有プールのサイズを小さくしてください。いずれかの値が増えているかどうかを確認するには、問合せを繰り返し実行してください。

データ・ディクショナリ・キャッシュの有効性の検証

SQL 文解析時にパフォーマンスを最適化するには、データ・ディクショナリ・キャッシュが、最も頻繁にアクセスされるデータを十分に保存できる大きさでなければなりません。データ・ディクショナリのキャッシュ・ミスが発生すると再帰コールが生成され、データベースのパフォーマンスが低下します。

Server Manager の統計画面では、データベース起動以降の再帰コールの回数がの Total 列に表示されます。Oracle8 Server が起動後に継続して再帰コールを行っていないければ、データ・ディクショナリ・キャッシュは、ディクショナリ・データに対して十分な大きさです。アプリケーションの実行中に再帰コールの回数が累積される場合は、データ・ディクショナリ・キャッシュのサイズを増やす必要がある場合もあります。

注意： ディクショナリ・キャッシュが小さすぎる場合は、V\$ROWCACHE 表に問い合せて、キャッシュ・アクティビティを調べてください。

適切なライブラリ・キャッシュ領域の割当て

パフォーマンスの向上率: 0 ~ 50%

ライブラリ・キャッシュには、共有 SQL および PL/SQL 領域が含まれます。SQL を再使用できる場合でも、ライブラリ・キャッシュが小さすぎると再使用は行われません。V\$LIBRARYCACHE 表に問い合せて、ライブラリ・キャッシュ・ミスがパフォーマンスに影響しているかどうかを確認してください。

次の問合せを実行し、V\$LIBRARYCACHE の統計を一定期間監視します。

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

問合せは次のような出力を返します。

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

PINS の合計（最初の列）は、実行のために SQL 文、PL/SQL ブロックおよびオブジェクト定義が合計で 320,871 回アクセスされたことを示しています。RELOADS の合計（最後の列）は、その実行のうちの 549 回はライブラリ・キャッシュ・ミスであったことを示しています。RELOADS の合計が 0（ゼロ）に近く、その割合は 1% 未満であることが必要です。

PINS に対する RELOADS の割合が 1% を超えている場合は、`init$sid.ora` ファイルの `SHARED_POOL_SIZE` パラメータの値を増やして、ライブラリ・キャッシュに割り当てられるメモリーを追加してください。

大きい PL/SQL ブロックの共有プールへのロック

使用頻度が低い共有オブジェクトは、必要に応じて共有プールにロックする必要があります。これは、ライブラリ・キャッシュ・ラッチがボトルネックになった場合に有効です。`dbms_shared_pool` ユーティリティ・パッケージを使用して、共有プールのオブジェクト・サイズを決めます。詳細は、`$ORACLE_HOME/rdbms/admin` ディレクトリにインストールされている PL/SQL スクリプト `dbmspool.sql` のコメントに記載されているドキュメントを参照してください。

次の手順で PL/SQL をインストールし、サーバー出力のバッファを使用可能にして、`sizes()` プロシージャを実行します。

1. `$ORACLE_HOME/rdbms/admin` に移動します。

2. SQL*Plus を起動して、次のコマンドを実行します。

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

xxx には 20000 以上、minsize には 20 を設定してください。頻繁に使用される共有オブジェクトを識別した後は、オブジェクトを共有プールにロックする必要があるごとに、プロシージャ (keep VARCHAR2、flag CHAR DEFAULT 'P') を実行できます。dbms_shared_pool.sizes は、共有プールにある minsize の値より大きいオブジェクトの一覧を表示します。

たとえば、keep (obj_name, "P") は、共有プールに obj_name をロックします。

unkeep プロシージャは、オブジェクトのロックを解除します。

提案： データベース起動の一部として前述のプロシージャを実行するために、SQL*Plus スクリプトを作成することをお勧めします。

セッション・キャッシュ・カーソルの最適化

Oracle Forms アプリケーションを広範囲に使用する場合（または、セッション・カーソルをクローズおよび再オープンする他のプログラムを使用する場合）は、セッション・カーソルをライブラリ・キャッシュに保存するように Oracle Server を設定してください。これによって、パフォーマンスが大幅に向上します。

キャッシュするセッション・カーソルの最大値を、初期化パラメータ SESSION_CACHED_CURSORS に設定します。キャッシュのパフォーマンスを監視し、ヒット率に基づいてキャッシュ・サイズを調整します。たとえば、次の SQL 文は、セッション・カーソルのキャッシュをチューニングするデータを取り出します。

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

ディスク I/O のチューニング

I/O ボトルネックは、最も簡単に識別できるパフォーマンスの問題です。使用可能なディスク全体で I/O を均等に分散させて、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、および Parallel Query Option を使用しないデータベースでは、異なるデータ・ファイルと表領域を使用可能なディスクに分散させてください。

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` を使用します。

表 3-4 に、ディスク・パフォーマンスに影響する `sar -b` 列を示します。

表 3-4 ディスク・パフォーマンスに影響する `sar -b` 列

<code>bread/s</code> , <code>bwrit/s</code>	読み込んだブロック数および書き込んだブロック数 (ファイル・システム・データベースで重要)
<code>pread/s</code> , <code>pwrit/s</code>	パーティションの読み込み数およびパーティションの書き込み数 (RAW パーティションのデータベース・システムで重要)

ディスク・パフォーマンスで重要な `sar -u` 列は `%wio` で、これによってブロック I/O での CPU の待ち時間の割合がわかります。

キー・インジケータは次のとおりです。

- `bread`、`bwrit`、`pread` および `pwrit` の合計は、ディスク I/O サブシステムの状態を示します。この合計が大きいほど、ディスク I/O ボトルネックが起きる可能性が高くなります。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` は 90 を超える値、`%wcache` は 60 を超える値にする必要があります。そうしなければ、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` が常に 20 を超える値の場合、システムは I/O バウンドになります。

索引と表の分離

索引および索引が参照する表が同じドライブにある場合、索引の検索に関連するすべての I/O は、同じディスクに集中します。索引と表は、I/O 負荷を分散するために、別々のドライブに格納する必要があります。

ユーザー `u` が所有する索引を、OLD 表領域から NEW 表領域に移動するには、次の手順を実行します。

1. 索引の保存、および NEW の作成に必要なサイズを計算します。
2. NEW の RESOURCE 権限を *u* に与えます。
3. 次のように指定して、*u* が所有するセグメントをエクスポートします。
`full=n, rows=n, indexes=y`
4. *u* が所有する索引を削除します。
5. 次のように指定して、*u* が所有するセグメントをインポートします。
`full=n, rows=n, indexes=y, indexfile=filename`
6. *filename* を編集して OLD を NEW に変更します。
7. SQL*Plus から *filename* を実行します。

参照： `export` および `import` ユーティリティの詳細は、『Oracle8 Server ユーティリティ』を参照してください。

REDO ログの専用ディスク・デバイスへの配置

パフォーマンスの向上率: 0 ~ 15%

ご使用の Oracle アプリケーションで INSERT および UPDATE アクティビティが頻繁に発生する場合、その他のディスク・アクティビティをサポートしないディスク上に REDO ログを配置することで、Oracle パフォーマンスを最大限に向上させることができます。また、ARCHIVELOG オプションを使用可能にした場合は、各 REDO ログを別々のディスクに置いて、LGWR プロセス（現行の REDO ログへの書込み）と ARCH プロセス（クローズされた REDO ログからの読み込み）との間で生じるディスク競合を最小限に抑えます。

REDO ログを RAW デバイス上に配置すれば、パフォーマンスはさらに向上します。この場合、次の理由から、REDO ログは、最初のファイルの 1 つとして RAW デバイスに配置してください。

- REDO ファイルは連続的に書込みおよび読み込みが行われるため、RAW デバイスの利点が最大限に向上する。
- RAW デバイスでは、非同期読み込みおよび書込みができる。
- REDO ファイルのサイズは固定されているので、RAW デバイスの管理コストを最小限に抑えられる。

参照： 詳細は、第 2 章の「[RAW デバイス](#)」を参照してください。

論理ボリュームの使用（NCR MP-RAS および Solaris Intel の場合）

パフォーマンスの向上率：0 ~ 500%

論理ボリューム・マネージャ（LVM）によって、複数のディスク・ドライブにデータをストライプ化できます。

LVM の使用をお勧めしますが、Oracle8 では、LVM を使用しなくてもデータ・ファイルをストライプ化できます。この場合、CREATE TABLE コマンドの DATAFILE キーワードを使用します。ただし、LVM は小さなストライプ・サイズを使用して、I/O をランダムかつ自動的に分散する傾向があるので、通常の場合は LVM を使用した方がパフォーマンスが向上します。

注意： LVM は、DG/UX Intel および SCO UnixWare では使用できません。DG/UX Intel では、sysadm ユーティリティが複数の物理ディスクに論理ボリュームを作成するため、柔軟性のあるアーキテクチャが提供されます。使用方法の詳細は、『Managing the DG/UX System』マニュアルを参照してください。

データベース・ライターをチューニングして書き込みバンド幅を大きくする

パフォーマンスの向上率：0 ~ 15%

Oracle では、データベース・ライター（DBWR）のアクティビティがボトルネックになることを避けるために、次の方法が用意されています。

- 非同期 I/O の使用
- 複数の DBWR プロセスの使用
- I/O スレーブの使用

これらの方法を使用する前に、システムで DBWR バンド幅を増やす必要があるかどうかを判断します。現在の状態については、V\$SESSION_WAIT ビュー（起動以降の状態については、V\$SYSTEM_EVENT ビュー）に問い合わせます。次の例では、free buffer waits の TOTAL_WAIT 列に値が表示されているので、DBWR がボトルネックを起していると判断します。

```
SELECT * FROM V$SYSTEM_EVENT;

EVENT                                TOTAL_WAIT
-----                                -
...
free buffer waits                    463
...
```

TOTAL_WAIT の値が大きい場合は、以降の項で説明する方法のいずれかを適用することを検討してください。

RDBMS I/O には、新しい初期化パラメータ・セットが付属しています。このパラメータ・セットによって、I/O の動作を制御できます。表 3-5 に、I/O の初期化パラメータを示します。

表 3-5 I/O の初期化パラメータ

パラメータ	許容値	デフォルト値
DISK_ASYNCH_IO	TRUE/FALSE	TRUE
TAPE_ASYNCH_IO	TRUE/FALSE	TRUE
BACKUP_DISK_IO_SLAVES	TRUE/FALSE	FALSE
BACKUP_TAPE_IO_SLAVES	TRUE/FALSE	FALSE
DBWR_IO_SLAVES	0 ~ 999	0
LGWR_IO_SLAVES	0 ~ 999	0
ARCH_IO_SLAVES	0 ~ 999	0
DB_WRITER_PROCESSES	1 ~ 10	1

非同期 I/O

非同期 I/O を使用することによって、プロセスは、書込み後すぐに次の操作に進めます。アイドル時間が短くなるため、システム・パフォーマンスが向上します。

非同期 I/O が必要でなかったり、使用できなかったり場合があります。表 3-5 に記載されているパラメータのうち、最初の 2 つ (DISK_ASYNCH_IO および TAPE_ASYNCH_IO) によって、ディスクおよびテープ・デバイスに対する非同期 I/O をオフにできます。

注意： DG/UX Intel および SCO UnixWare の場合、非同期 I/O が行えるのは、RAW デバイスの場合のみです。DG/UX Intel の場合、非同期 I/O のパフォーマンスを高めるには、Oracle SGA をメモリーにロックする必要があります。詳細は、「物理メモリーへの SGA のロック」の項を参照してください。SCO UnixWare の場合は、SGA をロックする必要はありません。

複数の DBWR プロセス

DB_WRITER_PROCESSES パラメータは、`initsid.ora` ファイルのデータベース・ライターの数を制御します。書込みを頻繁に行うアプリケーションでは、DBWR のバンド幅が大きなボトルネックを起こすことがあります。データベース・ライターの数を変えてみてパフォーマンスを検証し、ご使用のデータベースに最も適したデータベース・ライターの数を決めます。DBWR_IO_SLAVES を使用する場合、DB_WRITER_PROCESSES の設定に関係なく、使用するデータベース・ライター・プロセスの数は 1 つだけです。

DG/UX Intel NUMA システムでは、DB_WRITER_PROCESSES をシステムの NUMA プロセス数以上に設定することをお勧めします。

I/O スレーブ

I/O スレーブは、I/O を実行するだけの特殊なプロセスです。I/O スレーブは Oracle8 の新機能で、Oracle7 の複数 DBWR (DB_WRITERS パラメータ) の概念を一般化したものです。他のプロセスでも同様に実行でき、非同期 I/O が使用可能かどうかに関係なく動作します。

各プロセス・タイプの I/O スレーブ数のデフォルトが 0 のため、特別に設定しない限り、I/O スレーブは実行されません。非同期 I/O が使用できない場合は、DBWR_IO_SLAVES を 0 より大きい値に設定する必要があります。DBWR_IO_SLAVES パラメータを使用するより DB_WRITER_PROCESSES パラメータを増やす方が、作業負荷を減らすことができます。LGWR_IO_SLAVES には、9 以下の値を設定することをお勧めします。

大きなディスクへの要求キューの検索

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。要求キューは、特定のディスクに対する I/O のボリュームが大きい場合や I/O の平均シーク時間が長い場合に起きます。ディスク要求キューは、0 (または 0 に近い値) であることが理想的です。Server Manager の「File I/O Monitor」にある「Resp Time」フィールドには、要求の待機時間が表示されます。

他のディスクへのホット・ファイルの移動

I/O のバランスを取るために、頻繁にアクセスされるホット・ファイルを、比較的使用されていないディスク・デバイスへ移動します。ファイル全体を、実行中のディスクから比較的使用されていないディスクへ移動できます。また、1 つのホット・ファイルを複数のディスクにストライプ化し、ファイルの一部を各ディスクに置くこともできます。

ホット・ファイルへの I/O の低減

パフォーマンスの向上率: 0 ~ 50%

ディスク・デバイスにホット・ファイルが 1 つだけあり、このファイルが担当する要求キューが大きい場合は、ホット・ファイルを他のディスクに移動しても効果がありません。

問題の Oracle ファイルまたは表領域に複数セグメント (表および索引など) のデータがある場合は、頻繁にアクセスされるセグメントを別の表領域および別のファイルへ移動します。

データベース・セグメントの物理デバイスは表領域レベルで指定できます。セグメントが 1 つだけの場合は、そのセグメントのデータを 1 つの表領域内の複数のファイルに置き、表をストライプ化することを検討してください。

データベースの過度な断片化の確認

Oracle データ構造の断片化が起きると、CPU が、複数の物理 I/O から 1 つの論理 I/O 要素を結合しなければなりません。必要以上のオーバーヘッドによって、応答時間が長くなります。

エクステントの断片化

データベース・セグメントに、複数の不連続のディスク領域のエクステントが含まれている場合があります。この場合、不連続のディスク読みや分割 I/O によって、I/O 時間が長くなります。分割 I/O は、要求されるデータがディスク上で不連続エクステントに分散しているために、1 回の I/O 要求が複数の物理 I/O に分割される場合に発生します。

表領域の断片化

Oracle 表領域は、複数の個別ファイルから構成されています。表領域内の Oracle セグメント（表および索引など）は、多数の個別エクステントから構成されるため、表領域の断片化が起きます。表をストライプ化する場合を除いて、表領域の断片化は避けたい問題です。データベース・セグメントを削除することに、表領域の断片化が起きます。表領域が断片化すると、空き領域を有効に使用できません。

表領域の断片化によって、Oracle のマルチブロック読み機能が利用できなくなります。また、連続した空きエクステントよりも大きいエクステントが存在すると、データベース領域が無駄になります。

次の SQL 文を使用して、表領域の断片化を識別できます。

```
SELECT * FROM DBA_EXTENTS;
```

次の SQL 文を使用して、空き領域を問い合わせることができます。

```
SELECT * FROM DBA_FREE_SPACE;
```

空き領域の断片化には、次の 2 種類があります。

- 泡立ち状：アクティブなエクステント間に挟まっていたエクステントを削除したときに、泡のように小さく連続していない空き領域が形成されます。
- 蜂の巣状：隣接したエクステントが削除されるたびに、空き領域が連続的な断片に区分化されます。

utlbstat および utlestat の再帰コールの値が高い場合は、表領域の断片化が考えられます（ただし、データ・ディクショナリ・キャッシュが適切にチューニングされている場合）。

参照： 詳細は、Craig A.Shallahamer の『Avoiding a Database Reorganization』を参照してください。この資料は、WWW の <http://www.europa.com/~orapub> から入手できます。

より多くのデータベースの使用

パフォーマンスの向上率: 0 ~ 10%

ご使用のシステムの I/O が限界にある場合は、データベース・バッファの数を増やして、より多くのデータをキャッシュし、I/O を削減します。ページングが増えない限り、バッファの数（およびヒット率）を増やしてください。

適切なファイル・システム・タイプの選択

ほとんどの UNIX システムでは、ファイル・システムを選択できます。ファイル・システムの特徴はそれぞれ異なり、それらがデータにアクセスする方法は、データベース・パフォーマンスに大きな影響を与えます。次に、代表的なファイル・システムを示します。

- `s5`: UNIX System V ファイル・システム
- `ufs`: BSD UNIX から派生した UNIX ファイル・システム
- `vxfs`: Veritas ファイル・システム

SCO UnixWare の場合、`convosync=dsync` キャッシング・アドバイザリを付けて `vxfs` ファイル・システムをマウントすることによって、パフォーマンスを改善することができます。

- `RAW` デバイス: ファイル・システムなし

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限りません。`ufs` ファイル・システムが異なると、インプリメンテーションも異なるため、比較が困難です。また、パフォーマンスを向上させるために `ufs` が選択される場合があります。この場合、選択したファイル・システムによって、パフォーマンスは 0 ~ 20% 異なります。

RAW パーティションおよび RAW デバイス (I/O バウンドの場合)

パフォーマンスの向上率: 5 ~ 40%

ファイル・システムのかわりに RAW パーティションを使用すると、パフォーマンスが改善されます。これは、データベース・ライターが UNIX バッファ・キャッシュをバイパスし、ファイル・システムのオーバーヘッドをなくすためです。この結果、I/O ごとの命令が減ります。

RAW パーティションを使用している場合は、UNIX バッファ・キャッシュ・サイズを最小限にしてください。UNIX バッファ・キャッシュ・サイズを設定する UNIX カーネル・パラメータは、`BUFPCT` または `NBUF` です。

注意: この処理は、同一システム上の他のアプリケーションに影響を及ぼす場合がありますので注意してください。

ディスク I/O の最適化が失敗したとき

ディスク I/O を最適化しても I/O のボトルネックがなくなる場合、いくつかのアプリケーションを他のシステムに移動するか、またはディスク・ドライブおよびコントローラをご使用のシステムに追加する必要があります。

ブロック・サイズのチューニング

通常、データ記憶領域の単位は MB ですが、Intel UNIX オペレーティング・システムおよび Oracle8 Server では、ブロックというデータ記憶の単位で I/O を行います。オペレーティング・システムのブロック・サイズは、Oracle のブロック・サイズと必ずしも一致しません。

`initsid.ora` ファイルの `DB_BLOCK_SIZE` パラメータを変更してデータベースを作成すると、Oracle8 Server のブロック・サイズを設定できます。

Oracle ブロック・サイズを変更すると、ディスク・ハードウェア、ファイル・システム、およびアプリケーションごとに、データベースのパフォーマンスを変更できます。ブロック・サイズはデフォルトでほとんどの環境に適応する必要がありますが、サイズを変更することによって、パフォーマンスがわずかに向上します。ただし、ブロック・サイズを変更する場合は、新しくデータベースを作成する必要があることに注意してください。新しいデータベースにデータを転送する前にブロック・サイズを変更してみて、最適な構成を判断してください。

Oracle ブロック・サイズの指定

Intel UNIX では、デフォルトの Oracle ブロック・サイズは 2KB、最大ブロック・サイズは 16KB です。

実際のブロック・サイズは、16KB 以下で 2KB の倍数の値を設定します。

アプリケーションによって異なりますが、通常、最適なブロック・サイズはデフォルトのサイズです。異なる Oracle ブロック・サイズでデータベースを作成する場合は、`initsid.ora` ファイルに次の行を追加します。

```
db_block_size=new_block_size
```

表のストライプ化

表のストライプ化は、大きな表のデータを小さな部分に分割し、これらの部分を別々のディスクの別々のデータ・ファイルに格納するプロセスです。これによって、複数のプロセスがディスク競合なしで表の異なる部分に同時にアクセスできます。ストライプ化は、多数の行を持つ表へのランダムなアクセスを最適化する上で特に有効です。

参照： ALTER TABLE コマンドの ALLOCATE EXTENT DATAFILE パラメータの詳細は、『Oracle8 Server SQL リファレンス』を参照してください。

CPU の使用状況のチューニング

Oracle ユーザー / プロセスの優先順位をすべて同じにする

Oracle は、すべてのユーザーおよびバックグラウンド・プロセスを同じ優先順位で操作するように設計されています。優先順位を変更すると、競合が起きたり応答時間に影響が出たりする場合があります。

たとえば、ログ・ライター・プロセス (LGWR) の優先順位を低くすると、LGWR の実行頻度は低くなり、ボトルネックになります。逆に、LGWR の優先順位を高くすると、ユーザー・プロセスの応答時間は長くなってしまいます。

使用状況パターンの再編成

システムの負荷がピークを超えてしまった場合、負荷をピーク以外の時間に分散させてください。たとえば、バッチ処理で夜間にバックアップを取り、アプリケーションを他のシステムに移動するという方法があります。

マルチ・プロセッサ・システムでのプロセッサ親和性 / バインディングの使用

パフォーマンスの向上率 : 0 ~ 10%

マルチ・プロセッサ環境では、できるだけプロセッサの親和性およびバインディングを使用します。プロセッサ・バインディングを使用すると、ある CPU から別の CPU にプロセスが移されることはないので、CPU キャッシュ内の情報をさらに有効に利用できます。サーバー・シャドウ・プロセスをバインドすると、常にアクティブになるため、キャッシュを使用して、バックグラウンド・プロセスを CPU 間で実行できるようになります。プラットフォームによっては、プロセスが自動的にバインドされる場合があります。

クライアント / サーバー構成の使用

ご使用のシステムが CPU バウンドになっている場合は、アプリケーションを別のシステムに移動させ、CPU の負荷を減らします。たとえば、Oracle Form などのフォアグラウンド・プロセスによる負荷をクライアント・マシンに移し、データベース・サーバー・マシンの CPU サイクルを解放します。

大量のエクスポート / インポートおよび SQL*Loader ジョブでのシングルタスク・リンクの使用

大量のデータをユーザーと Oracle8 間で（たとえば、エクスポート / インポートを使用して）転送する必要がある場合、シングルタスク・アーキテクチャを使用すると効果的です。シングルタスク・インポート (impst)、エクスポート (expst) および SQL*Loader (sqlldrst) 実行ファイルを作成する場合、\$ORACLE_HOME/rdbms/lib ディレクトリにある ins_rdbms.mk という Make ファイルを使用します。

次のコマンドを実行すると、impst、expst および sqlldrst の実行ファイルが作成されます。

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

注意： Oracle 実行ファイルをシングルタスクとしてリンクすると、ユーザー・プロセスは SGA 全体に直接アクセスできるようになります。さらに、シングルタスクを実行するには、多くのメモリーが必要です。これは、Oracle 実行ファイル・テキストがフロントエンド・プロセスとバックグラウンド・プロセス間で共有されなくなるためです。

Oracle リソース競合のチューニング

データベースが正しく動作しない場合、および問題の原因が CPU やディスクの競合ではない場合は、Oracle リソースの競合が原因の場合があります。

参照： 詳細は、『Oracle8 Server ユーティリティ』を参照してください。

V\$ 表を使用して競合を特定する方法

V\$SYSTEM_EVENT 表は、データベース・アクティビティのスナップショットに使用します。

V\$SYSTEM_EVENT の統計情報は、Oracle が時間をどのように使用しているかを示します。これによって、潜在的な問題を識別できます。表は、次の SQL 文で問い合わせます。

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

高度にチューニングされたデータベースでは待ち状態が発生します。また、この表にイベントが存在する、または存在しないからといって、それが問題を示しているとは限りません。client message、pmon timer、smon timer、rdbms ipc message、rdbms ipc reply などのイベントは正常です。この表の行数は動的に変化します。イベントに関する情報がない場合は、この表にイベントは表示されません。

V\$SYSTEM_EVENT は、累積される表です。この表は、イベントの発生を測定する表を参照するのも役立ちます。V\$SESSION_WAIT 表を使用するには、次のように入力します。

```
SELECT sid, event, pltext, pl, p2text, p2
FROM V$SESSION_WAIT;
```

この問合せによって、一連のイベントのスナップショットが提供されます。イベントの頻度がデータベースの負荷によってどのように変わるかを観察すると、実行されている Oracle 操作と SQL 文の性質の両方がよくわかります。

次に、V\$SESSION_WAIT の問合せの出力例を示します。

SID	EVENT	P1TEXT	P1	P2TEXT	P2
1	pmon timer		0		0
2	buffer busy waits	file#	7	block#	792
10	latch free	address	8.05E08	number	8
...					

競合を起こすセグメントの特定

Oracle リソースの競合が問題であると判断した場合は、競合を起こしているセグメントを特定します。多数の buffer busy waits が問題であるという判断にいたることがあります。block number および file number を使用し、次のように入力して競合のタイプを判断してください。

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

次のように出力されます。

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

この出力は、索引、クラスタ、ロールバック・セグメントではなく、表セグメントで競合が起きていることを示しています。ファイル番号とブロック番号がわかっているので、次のように入力することで XSBH 表から追加情報を取得できます。

```
SELECT class
FROM XSBH
WHERE dbafile=7 AND dbablk=792;
```

この問合せによって、ブロックのクラス番号が提供されます。この番号は、次の表を使用して解釈できます。

表 3-6 ブロック・タイプとクラス

クラス	ブロック・タイプ
0	システム・ロールバック・セグメント
1	データ・ブロック
2	ソート・ブロック
3	保存 UNDO ブロック
4	セグメント・ヘッダー・ブロック
5	保存 UNDO セグメント・ヘッダー・ブロック
6	空きリスト・ブロック
7	エクステント・マップ・ブロック
8	ビットマップ・ブロック
9	ビットマップ索引ブロック
10+(n*2)	UNDO セグメント・ヘッダー・ブロック
11+((n*2)+1)	UNDO セグメント・ブロック

ラッチ解放競合の低減

V\$SYSTEM_EVENT の出力結果から、latch free の値が競合を起こしていることを示す場合、V\$SESSION_WAIT の出力結果を使用して、競合の原因を判断します。ラッチ番号は、p2 フィールドに表示されます。また、次のように入力してラッチ番号を識別できます。

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

次のように出力されます。

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

cache buffer chains ラッチによって、競合が起きる場合があります。cache buffer lru chain ラッチおよび cache buffer handles ラッチも同じです。通常、これらのラッチから、SGA のバッファ数を大きくする必要があることがわかります。

ロールバック・セグメントの競合の低減

データベースのデータ・ファイルには、ロールバック情報用に割り当てられたセグメントがあります。ロールバック・セグメントを構成するデータベース・ブロックは頻繁にアクセスされるため、ロールバック・セグメントが競合の対象となることがあります。

次の SQL 文を使用して、ロールバック・セグメントの空き領域要求が延期される頻度を判断します。

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

ヒット率は、95% 以上である必要があります。

- ロールバック・セグメントが少なすぎる場合は、ロールバック・セグメントを追加します。
- ユーザーが正しく割り当てられていない場合は、大きいトランザクションを実行するユーザーを、大きいロールバック・セグメントに割り当てます。
- ロールバック領域が不十分な場合、必ずエラー・メッセージ「Snapshot too old」が表示されます。

REDO ログ・バッファ・ラッチの競合の低減

REDO ログ・バッファへのアクセス回数が多すぎると、REDO ログ・バッファ・ラッチの競合が起きる可能性があります。Server Manager Latch Display を使用して、REDO ログ・バッファ・ラッチのアクティビティを調べます。

あるラッチの gets に対する misses の割合が 10% を超える場合、そのラッチの競合によってパフォーマンスが低下することがあります。各 sleep は、ラッチを要求するプロセスが延期されることを示します。

注意： 複数の CPU を持つシステムでは、発生する競合の数が多くても、パフォーマンスが低下することはありません。

redo allocation ラッチの競合を減らすことができます。そのためには、`init.ora` ファイルの `LOG_SMALL_ENTRY_MAX_SIZE` パラメータの値を小さくすることによって、すべてのシングル・プロセスがラッチを獲得している時間を最小限にします。

マルチ・プロセッサ環境の REDO コピー・ラッチの競合を減らすには、次のタスクを行います。

- `LOG_SIMULTANEOUS_COPIES` の値を大きくして、ラッチを追加します。
- ご使用の Oracle8 インスタンスで使用できる CPU の 2 倍の REDO コピー・ラッチを使用します。

パラレル問合せおよびパラレル DML の競合

CPU の使用頻度が限界を超えないように、また、使用可能な問合せサーバーの供給を消耗しないようにするために、パラレル問合せをチューニングします。次のように入力し、`V$Q_SYSSTAT` ビューを使用してアクティブな問合せサーバー数を判断します。

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

次のよう出力されます。

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

サーバーが `PARALLEL_MAX_SERVERS` の値を超えるほどビジー状態になった場合、いくつかのパラレル問合せが連続的に処理されていることがあります。

前回の問合せの他に `sar -u` を実行して、CPU の負荷を監視します。一定期間、これらの測定を監視します。次の表に、`PARALLEL_MAX_SERVERS` に対する `servers busy` の割合を基に、CPU の使用状況ごとのチューニング操作を示します。

表 3-7 CPU の使用状況および `PARALLEL_MAX_SERVERS` を基にした推奨するチューニング操作

Busy/Max Server	大量の CPU の使用 (95 ~ 100%)	平均的な CPU の使用 (60 ~ 80%)	少量の CPU の使用 (0 ~ 30%)
ほとんどの場合 1.0	表および問合せの並列度を大幅に減らします (システムをチューニングします)。	表および問合せの並列度を減らします。	MAX Server を増やします (システムをチューニングします)。
まれに 1.0	CPU の使用が最大になったときに、問合せを識別します (システムをチューニングします)。	MAX Server を増やして監視します。表および問合せの並列度を減らします。	MAX Server を増やします。
.3 ~ .7	Query Servers が 40% 以上の CPU を使用している場合、MAX Server を減らします (システムをチューニングします)。	チューニングされています。	表および問合せの並列度を減らします (MAX Server を増やします)。
0 ~ .2	システムをチューニングします (プロセッサの追加を検討します)。	MAX Server を減らすことを検討します。	表および問合せの並列度を減らします。

マルチ・プロセッサ・システムでのスピン・カウン트의チューニング

マルチ・プロセッサ環境では、初期化パラメータ `SPIN_COUNT` のチューニングによって、パフォーマンスが向上します。

プロセスは、ラッチを獲得するまでラッチを要求し続けます。要求回数が `$SPIN_COUNT` の値に到達すると、プロセスは、ラッチの獲得に失敗し、スリープ状態になった後、再びラッチを獲得しようとします。ラッチは、低レベルのロックであるため、プロセスはラッチを長時間保持できません。そのため、プロセスをスリープ状態にするより、プロセスをスピニングして CPU 時間を使用する方が効率的です。

ラッチの競合レベルを調べるには、`utlbstat` および `utlestat` スクリプトの `miss rate` および `sleep rate` を監視します。スピン・カウン트를チューニングして、`sleep rate` を減らしてみます。競合レベルが高い場合は、ラッチを獲得する前に、スピン・カウンを増やして、プロセスがさらにスピニングできるようにします。ただし、スピン・カウンを増やすと、CPU の使用率も増えるので、システムのスループットが低下することがあります。

UNIX カーネル・パラメータのチューニング

UNIX カーネルをできるだけ小さくすることによって、パフォーマンスが向上します。UNIX カーネルでは、通常、事前に物理 RAM が割り当てられるため、Oracle など他のプロセスで利用できるメモリーが少なくなります。

従来は、NBUF、NFILE、NOFILES などのカーネル・パラメータを使用してカーネル・サイズを調整していました。しかし、これらのカーネル・パラメータは、UNIX 構成ファイルに設定されていても、ほとんどの UNIX インプリメンテーションによって実行時に動的に調整されます。

メモリーが割り当てられているビデオ・ドライバ、ネットワーク・ドライバおよびディスク・ドライバを検索します。それらのドライバは、削除できる場合がほとんどです。それによって、他のプロセスで利用できるメモリーが増えます。

警告： 必ず UNIX カーネルのバックアップを取っておいてください。バックアップ方法の詳細は、ハードウェア・ベンダーのドキュメントを参照してください。

Oracle Parallel Server (OPS) のリソース競合のチューニング (DG/UX Intel および NCR MP-RAS の場合)

注意： 今回のリリースの Solaris Intel および SCO UnixWare では、
Oracle Parallel Server オプションはサポートされていません。

多くの統計情報によって、OPS の競合がわかります。次の表から OPS の競合を判断してください。

- V\$SESSION_WAIT
- V\$SESSION_EVENT
- V\$SYSTEM_EVENT
- V\$SQL_AREA
- V\$CACHE
- V\$PING

通常、ロックの競合は最も重要な要因です。ロックが獲得され、変換されている間は、ロックの競合は、ディスク I/O および遅延を示します。アプリケーションを適切にパーティション化することが、ロックの競合を回避する唯一の方法です。

複数インスタンスのシステム上で Oracle8 Parallel Server を実行すると、パフォーマンスが低下します。次の原因が考えられます。

- CPU の競合
- [ロックの競合](#)
- [ローカル・ディスク以外での I/O](#)

CPU の競合

SHARED モードで Oracle8 を起動し、複数のインスタンスを実行している場合は、必ず作業負荷がバランス良く複数のプロセッサに分散されるようにします。たとえば、すべてのジョブを 1 つのインスタンスで実行することがないように注意してください。また、CPU 集中型の複数のジョブを 1 つのインスタンスで実行し、それ以外のジョブをその他のインスタンスで実行するようなことも避けてください。

1 つのインスタンスに負荷が集中すると、次の問題が起きる場合があります。

- 処理能力の低下
- すべてのデータ・ページを保存する物理メモリーの不足

Oracle Net8 TCP/IP を使用して Oracle8 に接続する際、インスタンスごとに ORACLE_SID またはクラス名を指定できます。クラス名を指定する場合は、必ず `oratabdir` ファイルに存在するクラス名を入力します。その後、次のいずれかの方法で Oracle8 に接続します。

- ORACLE_SID を指定して Oracle Net8 TCP/IP を使用してインスタンスに接続する場合は、指定したインスタンスに直接接続します (そのインスタンスが過負荷状態の場合は、パフォーマンスが低下する可能性があります)。このオプションを使用すると、インスタンス間の CPU の負荷のバランスが取れます。複数インスタンス間の CPU に集中しているジョブを分散させてください。
- クラス名に接続すれば、インスタンスのグループに接続できます。このオプションを使用すると、同じようなジョブの数が増えても簡単に管理できます。

例 3-1

```
$ svrmgrl
SVRMGR> CONNECT SYSTEM/MANAGER@T:node:SID1
```

この例では、`sid SID1` に対応するインスタンスにのみ接続します。

ただし、GEN という名前のクラスがあり、異なるプロセッサ上に、GEN という名前の SID1 および SID2 という `sid` を持つ 2 つのインスタンスがある場合は、次のように入力して、Oracle Net8 TCP/IP を使用して GEN に接続できます。

```
$ svrmgrl
SVRMGR> CONNECT SYSTEM/MANAGER@T:node:GEN
```

これによって、両方のインスタンスに対して接続要求が出されます。

ロックの競合

ロック競合の確認

ブロックピンギングを最も起こしやすいブロックを判断するには、`SYS/password` で `~/rdbms/admin/catparr.sql` を実行し、`V$PING` ビューに問い合わせます。ブロックピンギングを起こしやすいブロックは、複数インスタンスで同時に使用されるブロックです。

ファイルへのロックの割当て

Oracle8 がブロックをロックする必要がある場合、Oracle8 はデータベース・ブロック・アドレスをリソース名の 1 つにハッシュします。初期化パラメータ `GC_DB_LOCKS` を使用すると、システムで使用する異なるリソース名の合計を指定できます。

初期化パラメータ `GC_FILES_TO_LOCKS` には、これらのリソース名がどのように複数のデータベース・ファイルに分散されるかを指定します。

データ・ファイルおよび索引が 1 つずつあり、ほとんどのトランザクションで索引を変更する必要がない場合、索引ファイルではなくデータファイルのブロック名からリソース名までさらに細分化できます。次に、例を示します。

```
GC_DB_LOCKS=1050                # Total no of resources
GC_FILES_TO_LOCKS="1=1000:2=50" # 1000 for datafile,
                                # 50 for index file
```

次の計算式に従って、データファイル *file#1* では、最初のブロックは *resource#1* に、2 番目のブロックは *resource#2* に割り当てられています。

$$resource_number = block_number \text{ MOD } file_resources$$

データベース・ブロックのクラスタ化

1 つのリソース名を複数の連続ブロックで使用できるように、ブロックをクラスタ化します。たとえば、データファイルに 5,000 のブロックがある場合、そのファイルのある領域のみにリソース名が適用されるように、クラスタ要素 5 を使用します (ブロック 1、1001、2001 などに適用される *resource#1* などの名前は使用しません)。次に例を示します。

```
GC_DB_LOCKS=1050                # Total no of resources
GC_FILES_TO_LOCKS="1=1000!5:2=50" # 1000 for datafile,
                                # cluster factor of 5
```

ブロックを保護するために使用されるリソース名は、このように計算します。リソース名が決まったら、Oracle8 はロック状態に関する情報を保存するためにロック要素を使用します。各リソース名には、個別のロック要素があります。

ファイル番号とファイル名の対応は、`V$DBFILE` 表に問い合わせて調べます。

ロックの競合の低減に関するヒント

- 同一のデータセット上で実行中のすべてのプロセスが、同一の Oracle8 インスタンスで実行するように、システムを編成します。
- 複数の大きな表が別々の表領域に保存されるように、データを編成します。この表領域は、1つの大きなファイルを使用するディスクに保存できます。また、1つのファイルの1つの表領域に、読み込み専用の表をグループ化することもできます。読み取り専用の表は、複数のディスクに分散されている場合もあります。変更しない索引も大きなファイルに保存できます。

これによって、変更するブロックで使用できるロック領域を残し、シングル・グローバル・ロックされた読み込み専用データのファイルを保護できます。
- 同一のデータベース・ブロックに保存されている異なるデータセット上で、複数のプロセスを実行しないようにします。そのためには、表を再作成し、`pctfree` と `pctused` を使用して、データファイルを別のものとして編成する必要がある場合があります。
- 記憶領域パラメータ `FREELIST GROUPS` または `FREELISTS` の値を高く設定して表を作成することによって、同時に複数インスタンスを挿入したために表上で競合が発生しないようにします。空き領域リストを活用するために、各インスタンスに別々のエクステンツを割り当てています。表領域に別々のファイルがあれば、これらのエクステンツをファイルに割り当てられます。
- ロックの競合を回避します。ロックの競合が頻繁に発生する場合、アプリケーションは拡張されません。ロックの競合は、`V$SYSSTAT` 表 (`CLASS=32`) のフィールドから判断できます。`CLASS` は `V$SYSSTAT` 表の列で、「32」はグローバル・ロックを示します。

$$\text{ロック変換率} = \frac{\text{固定取得の数} - \text{非同期ロック変換の数}}{\text{固定取得の数}}$$

アプリケーションを拡張させるには、ロック変換率を 95% 以上にする必要があります。ロックの競合が頻繁に起きる場合は、アプリケーションを再評価し、OPS 用に再設計が必要な場合があります。アプリケーションを Oracle が実行していることがロックの競合の最も共通している原因ですが、ロックが十分に割り当てられていないことが原因の場合があります。たとえば、OLTP アプリケーションでは、設計サポート・アプリケーションより多くのロックが必要です。`init.ora` ファイルのパラメータでロックを適切に割り当ててください。

- 索引の競合を回避します。シーケンス・ジェネレータがデータベース・レコードの主キーを作成する場合、索引の表は広範囲に使用されるため、データベース内の競合の原因になる場合があります。通常、シーケンス番号は連続しており、この番号をキーとしてデータを追加する場合は、同一の索引ブロックにエントリが追加されます。これによって、索引ブロックの競合が発生する場合があります。この問題を解決するには、このシーケンスの値を設定しないでおきます。その後、値を選択して、索引を別々のブロックに分散させます。

注意： 索引の競合は、Symmetric Multi-Processor (SMP) で実行する Oracle8 の問題になる場合がありますが、Oracle Parallel Server でのボトルネックになる可能性の方が高くなります。

ローカル・ディスク以外での I/O

ローカル・ディスク以外での I/O は、できるだけノードに対してローカルにします。そのノードに接続されているディスク上のインスタンスに対して、ロールバック・セグメントおよび REDO ログを保存します。この作業は、各ノードで競合を起こさずにデータが使用できるように、データをパーティション化する方法として行ってください。

Intel UNIX バッファ・キャッシュ・サイズのチューニング

RAW デバイスを最大限に活用するには、Oracle8 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、Intel UNIX バッファ・キャッシュのサイズも調整します。

Intel UNIX バッファ・キャッシュは、オペレーティング・システムで提供されます。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に転送されるメモリー内のデータ・ブロックを保存します。

Oracle8 バッファ・キャッシュは、Oracle データベース・バッファを保存するための領域です。Oracle8 では RAW デバイスが使用できるので、Intel UNIX バッファ・キャッシュを使用する必要はありません。

RAW デバイスに移動するときは、Oracle8 バッファ・キャッシュのサイズを大きくしてください。システムで利用できるメモリーに制限がある場合は、それに応じて Intel UNIX バッファ・キャッシュのサイズを小さくしてください。

キャッシュ・ヒット率の判断

キャッシュ・ヒット率によって、キャッシュ内にブロックが発見された時間の割合、およびディスク I/O が不要になった時間の割合がわかります。キャッシュ・ヒット率は、次の計算式で求めます。

$$(\text{logical_reads} - \text{physical_reads}) / \text{logical_reads}$$

logical reads は、CONSISTENT_GETS に DB_BLOCK_GETS を加えた数です。

Oracle8 のキャッシュ・ヒット率は、Server Manager のモニターおよび統計画面に表示されます。また、個々のプロセスのユーザー画面にも表示できます。

Intel UNIX の `sar` コマンドは、調整が必要なバッファ・キャッシュを判断する際に役立ちます。表 3-8 に、`sar` コマンドの構文を示します。

表 3-8 sar コマンドの構文

<code>sar -b</code>	Intel UNIX のバッファ・キャッシュ・アクティビティを表示します。
<code>sar -w</code>	Intel UNIX のスワッピング・アクティビティを表示します。
<code>sar -u</code>	CPU 使用状況を表示します。
<code>sar -r</code>	メモリー使用量を表示します。
<code>sar -p</code>	Intel UNIX のページング・アクティビティを表示します。

キャッシュ・サイズの調整

- キャッシュ・ヒット率が向上する限り、Oracle8 キャッシュ・サイズを大きくします。
- スワッピング / ページング・アクティビティが増える場合は、キャッシュ・サイズを小さくします。

Intel UNIX での SQL*Plus の管理

- [SQL*Plus の管理](#)
- [SQL*Plus の使用](#)
- [制限事項](#)

SQL*Plus の管理

設定ファイル

SQL*Plus の設定ファイルには、サイトごとに定義するグローバルなファイル (glogin.sql) およびユーザーごとに定義するファイル (login.sql) があります。glogin.sql および login.sql ファイルには、SQL*Plus セッションの最初に行われる SQL 文または SQL*Plus コマンドが記述されています。SQL*Plus を起動すると、最初に glogin.sql が読み込まれ、次に login.sql が読み込まれます。

Site Profile

Site Profile ファイルは、\$ORACLE_HOME/sqlplus/admin/glogin.sql です。SQL*Plus は、ユーザーが SQL*Plus を起動し、SQL*Plus が Oracle に接続するときに、このコマンド・ファイルを実行します。SQL*Plus をインストールすると、デフォルトの Site Profile は \$ORACLE_HOME/sqlplus/admin に保存されます。Site Profile がすでに存在する場合は、上書きされます。SQL*Plus を削除すると、既存の Site Profile も削除されます。

User Profile

User Profile ファイルは login.sql です。SQL*Plus は、ユーザーが SQL*Plus を起動し、SQL*Plus が Oracle に接続するときに、このコマンド・ファイルを実行します。User Profile は、Site Profile の後で実行されます。SQL*Plus は、起動されたディレクトリから User Profile を検索します。環境変数 SQLPATH にディレクトリ・パスをコロンで区切って設定すると、SQL*Plus はその順番で User Profile を検索します。

たとえば、現在のディレクトリが /u02/oracle で、次のような SQLPATH を設定しているとします。

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

SQL*Plus は、最初に現在のディレクトリ /u02/oracle で login.sql を検索します。現在のディレクトリに login.sql がない場合、SQL*Plus は /home、/home/oracle および /u01/oracle ディレクトリを検索します。

次に、login.sql ファイルのサンプルを示します。

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

PRODUCT_USER_PROFILE 表

SYSTEM ユーザーで SQL スクリプト `$ORACLE_HOME/sqlplus/admin/pupbld.sql` を実行し、Product 表および User Profile 表を作成します。

`$ORACLE_HOME/sqlplus/admin/pupbld.sql` は、シェル・スクリプト `$ORACLE_HOME/bin/pupbld` を使用して実行することもできます。このスクリプトを使用するには、環境変数 `ORACLE_HOME` および `SYSTEM_PASS` を設定する必要があります。`SYSTEM_PASS` には、SYSTEM ユーザーのユーザー名およびパスワードを設定します。次に例を示します。

```
% setenv SYSTEM_PASS SYSTEM/manager
% pupbld

Installing product user profile tables...

Product user profile tables installed.
```

`pupbld.sql` は、SQL*Plus のインストール時に「Create Database Objects」を選択した場合のみ、Installer によって実行されます。

デモンストレーション表

SQL*Plus 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。

デフォルト・インストール

「Default Install」によって「Create Database Objects」を選択すると、ユーザー SCOTT およびデモンストレーション表が自動的に作成されます。

カスタム・インストール

「Custom Install」によって SQL*Plus をインストールする場合、「Create Database Objects」を選択し、「Would you like to load the SQL*Plus Demo Tables?」というプロンプトに対し「Yes」を入力します。Installer は、ユーザー SCOTT (パスワードは TIGER) およびデモンストレーション表を作成します。

デモンストレーション表の手動作成

デモンストレーション表を手動で作成するには、SQL スクリプト

\$ORACLE_HOME/sqlplus/demo/demobld.sql を実行します。SQL*Plus で demobld.sql ファイルを実行すると、ユーザーはスキーマにデモンストレーション表を作成できます。次に例を示します。

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

次のようにシェル・スクリプト \$ORACLE_HOME/bin/demobld を使用することによって、\$ORACLE_HOME/sqlplus/demo/demobld.sql を実行できます。

```
% demobld scott tiger
```

デモンストレーション表の削除

デモンストレーション表を削除するには、SQL スクリプト

\$ORACLE_HOME/sqlplus/demo/demodrop.sql を実行します。SQL*Plus で demodrop.sql ファイルを実行すると、ユーザーのスキーマからデモンストレーション表を削除できます。次に例を示します。

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

次のようにシェル・スクリプト \$ORACLE_HOME/bin/demodrop を使用することによって、\$ORACLE_HOME/sqlplus/demo/demodrop.sql を実行できます。

```
% demodrop scott tiger
```

注意： SQL スクリプト demobld.sql および demodrop.sql はどちらも、EMP、DEPT、BONUS、SALGRADE および DUMMY 表を削除します。これらの表と同じ名前の表がスキーマに存在しないことを確認してから、どちらかのスクリプトを実行してください。そうしないと、表のデータが失われます。

ヘルプ機能

デフォルト・インストール

「Default Install」によって「Create Database Objects」を選択すると、ヘルプ機能が自動的にインストールされます。

カスタム・インストール

SQL*Plus のインストール時に、「Create Database Objects」を選択し、「Would you like to load the SQL*Plus Help Facility?」というプロンプトに対し「Yes」と入力します。ヘルプ機能がインストールされます。

ヘルプ機能の手動インストール

手でヘルプ機能をインストールするには、シェル・スクリプト `$ORACLE_HOME/bin/helpins` を使用します。このスクリプトを使用するには、環境変数 `ORACLE_HOME` および `SYSTEM_PASS` を設定する必要があります。`SYSTEM_PASS` には、`SYSTEM` ユーザーのユーザー名およびパスワードを設定します。次に例を示します。

```
$ setenv SYSTEM_PASS SYSTEM/manager
$ helpins

SQL*Loader: Release 8.0.6.0.0 - Production

(c) Copyright 1999 Oracle Corporation. All rights reserved.

Commit point reached - logical record count 828

SQL*Loader: Release 8.0.6.0.0 - Production

(c) Copyright 1999 Oracle Corporation. All rights reserved.

Commit point reached - logical record count 1024
Commit point reached - logical record count 1207

SQL*Loader: Release 8.0.6.0.0 - Production

(c) Copyright 1999 Oracle Corporation. All rights reserved.

Commit point reached - logical record count 1024
Commit point reached - logical record count 1304
Commit point reached - logical record count 2328
Commit point reached - logical record count 2724
Commit point reached - logical record count 2835
```

参照： 詳細は、『SQL*Plus ユーザーズ・ガイドおよびリファレンス』および README ファイル `$ORACLE_HOME/sqlplus/doc/release.doc` を参照してください。

SQL*Plus の使用

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`、`vi` などのデフォルトのオペレーティング・システム・エディタが起動されます。起動するには、環境変数 `PATH` にエディタのディレクトリが含まれている必要があります。

グローバルなデフォルト・エディタは、SQL*Plus の `_editor` オプションを使用して、DBA が `glogin.sql` に設定します。`login.sql` にエディタを指定して、この設定を上書きします。SQL*Plus は、起動時に両方のファイルを読み込みますが、ローカル・ファイル (`login.sql`) が優先されます。また、SQL*Plus セッション中に `_editor` オプションを設定し、両方のファイルの設定を上書きすることもできます。

`_editor` オプションが設定されていない場合、環境変数 `EDITOR` および `VISUAL` によって SQL*Plus エディタが指定されます。これらの環境変数は、`glogin.sql` または `login.sql` では設定できません。ユーザー起動ファイルまたはシステム・プロンプトで設定します。環境変数 `EDITOR` および `VISUAL` が両方設定されている場合、環境変数 `EDITOR` が優先されます。

エディタの順序設定

SQL*Plus は、次の順序でデフォルトのエディタを検索します。

1. SQL*Plus セッション中の `_editor` 変数
2. `login.sql` の `_editor` 変数
3. `glogin.sql` の `_editor` 変数
4. 環境変数 `EDITOR`
5. 環境変数 `VISUAL`

これらの値がどれも設定されていない場合、SQL*Plus はエディタに `ed` を使用します。

`_editor` オプションの設定

SQL*Plus の `_editor` オプションを設定するには、`login.sql` ファイルに次の行を追加します。

```
define _editor=editor_name
```

この場合、`editor_name` は UNIX エディタを表します。

環境変数の設定

Bourne シェルまたは Korn シェルの場合、次のように入力して環境変数にデフォルト・エディタを設定します。

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

C シェルの場合、次のように入力して環境変数にデフォルト・エディタを設定します。

```
% setenv UNIX_VAR editor_name
```

表 4-1 に、環境変数の構文の詳細を示します。

表 4-1 UNIX 環境変数の構文

<code>UNIX_VAR</code>	環境変数 EDITOR または VISUAL
<code>editor_name</code>	UNIX エディタ（たとえば、vi や ed）

デフォルト設定

システム・エディタを起動すると、現行の SQL バッファは編集バッファに入れられるため、エディタで利用できる文はすべて SQL 文を変更できます。SQL*Plus は、`afiedt.buf` テンポラリ・ファイルを使用します。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 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を感嘆符 (!) にすると、後続の文字列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際に使用するシェルは、環境変数 SHELL によって設定します。デフォルトのシェルは、`/bin/sh (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。

次の SQL*Plus コマンドを使用すると、それぞれに特定のタスクを実行できます。

- `[!]+command` (1 つのオペレーティング・システム・コマンドを実行する場合)
コマンドの実行後、制御は SQL*Plus に戻ります。
- `[!]+[Return]` (2 つ以上のオペレーティング・システム・コマンドを実行する場合)
終了したら、`[Ctrl]+[d]` を押して SQL*Plus に戻ります。

SQL*Plus への割込み

SQL*Plus の実行中に、次のような割込みができます。

- BSD マシンの場合は [Ctrl]+[c] を、System V マシンの場合は [Delete] 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。
- SQL*Plus プロンプトで [Interrupt] を押すと、別の SQL*Plus プロンプトが表示されます。

SPOOL コマンドの使用

SPOOL コマンドで生成されるファイルの拡張子のデフォルトは、.lst です。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。

次に例を示します。

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

制限事項

COPY コマンド

SQL*Plus の COPY コマンドは、同じバージョンのオペレーティング・システムが実行されているマシン上で使用できます。制限事項はありません。

基本的に、COPY コマンドは異なるバージョンのオペレーティング・システムが実行されているマシン間でも動作しますが、COPY コマンドが失敗した場合、rcp または ftp を使用して接続をテストしてください。ベンダーが提供するネットワーク・ソフトウェアによっては、rcp、ftp または COPY コマンドがシステム間で正常に動作しない場合があります。

注意： rlogin コマンドは、大きなデータ・パケットを送信または受信しないので、接続のテストに適していません。

COPY コマンドがシステム間で動作しない場合、システムおよびユーザー ID へのデータベース・リンクを作成して、コピーする表を指定します。そのためには、次のように入力します。

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

これによって、リモート・システムにあるコピー元の表で選択された行および列が、ローカル・システム上のコピー先の表に挿入されます。

ウィンドウのサイズ変更

SQL*Plus の LINESIZE のデフォルト値は 80、PAGESIZE のデフォルト値は 25 です。これらの環境変数によって、ウィンドウ・サイズが自動的に調整されるわけではありません。

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。

Intel UNIX での Oracle プリコンパイラおよび Oracle コール・インタフェースの使用

- Oracle プリコンパイラの概要
- Pro*C/C++
- Pro*COBOL (DG/UX Intel、SCO UnixWare および NCR MP-RAS の場合)
- Pro*FORTRAN (DG/UX Intel の場合)
- Oracle コール・インタフェース
- Oracle プリコンパイラ、Oracle コール・インタフェースのリンクおよび Make ファイル
- Oracle ライブラリを使用した静的リンクおよび動的リンク
- シグナル・ハンドラの使用方法
- XA 機能

Oracle プリコンパイラの概要

Oracle プリコンパイラとは、Oracle データベースの SQL 文と高水準言語で書かれたプログラムを組み合わせて使用するアプリケーション設計ツールです。Oracle プリコンパイラは、ANSI SQL と互換性があり、Oracle8 やその他の ANSI SQL DBMS 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。

プリコンパイラ実行ファイルの再リンク

すべてのプリコンパイラ実行ファイルは、Make ファイル \$ORACLE_HOME/precomp/lib/ins_precomp.mk を使用して再リンクされます。make コマンドの形式は、次のとおりです。

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

このコマンドを実行すると、新しい実行ファイルが \$ORACLE_HOME/precomp/lib ディレクトリに作成され、その後、その実行ファイルは \$ORACLE_HOME/bin ディレクトリに移されます。\$ORACLE_HOME/bin に移さずに新しい実行ファイルを作成するには、次のコマンドを使用します。

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

表 5-1 に、使用する製品に対する実行ファイル名を示します。

表 5-1 製品と実行ファイル名

製品	実行ファイル
Pro*C/C++	proc
Pro*COBOL 1.8.28	procob18 または rtsora
Pro*COBOL 8.0.6	procob または rtsora
Pro*FORTRAN	profor
Object Type Translator	ott

たとえば、Pro*C/C++ の実行ファイルを再リンクするには、次のように入力します。

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

プリコンパイラ構成ファイル

\$ORACLE_HOME/precomp/admin には、5 つの .cfg システム構成ファイルがあります。
表 5-2 に、製品とその構成ファイルを示します。

表 5-2 システム構成ファイル

製品	構成ファイル
Pro*C/C++ 8.0.6	pcscfg.cfg
Pro*COBOL 8.0.6	pcbcfg.cfg
Pro*COBOL 1.8.28	pcccob.cfg
Pro*FORTRAN 1.8.28	pccfor.cfg
Object Type Translator 8.0.6	ottcfg.cfg

すべてのプリコンパイラに共通の問題

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またはサブプログラム名を小文字に変換します。これによって、「No such user exit」エラーが発生する場合があります。この場合、オプション・ファイルの関数またはサブプログラム名の大文字 / 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グ・プログラム

ベンダー提供のデバッグとプリコンパイラに互換性がない場合があります。デバッガ上で動作するプログラムが、オペレーティング・システムでは動作しない可能性があります。

ireclen および oreclen の値

ireclen および oreclen パラメータに最大値はありません。

参照ドキュメント

次のドキュメントでは、プリコンパイラおよびインタフェース機能についての追加情報を説明しています。

- 『Oracle Pro*C/C++ プリコンパイラ・プログラマーズ・ガイド』
- 『Oracle Pro*COBOL プリコンパイラ・プログラマーズ・ガイド』
- 『Oracle プリコンパイラ・ガイド Pro*FORTRAN サプリメント』
- 『Oracle コール・インタフェース・プログラマーズ・ガイド』
- 『Oracle8 Server アプリケーション開発者ガイド』

Pro*C/C++

Pro*C/C++ リリース 8.0.6 の追加情報は、README ファイル
\$ORACLE_HOME/precomp/doc/proc2/readme.doc を参照してください。

Pro*C/C++ の管理

システム構成ファイル

Pro*C/C++ のシステム構成ファイルは、\$ORACLE_HOME/precomp/admin/pcscfg.cfg です。

Pro*C/C++ の使用

Pro*C/C++ を使用する前に、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。必要なバージョンの詳細は、『Oracle8 for Intel UNIX (DG/UX Intel, NCR MP-RAS, SCO UnixWare, Solaris Intel) インストール・ガイド』の第 1 章を参照してください。

デモンストレーション・プログラム

Pro*C/C++ プリコンパイラのさまざまな機能を参照するために、デモンストレーション・プログラムが提供されています。C、C++ および Object プログラムの 3 種類のデモンストレーション・プログラムがあります。Object プログラムのデモンストレーション・プログラムでは、Oracle8 Object の新機能が表示されます。すべてのデモンストレーション・プログラムは、\$ORACLE_HOME/precomp/demo/proc にあります。また、すべてのデモンストレーション・プログラムでは、\$ORACLE_HOME/sqlplus/demo/demobld.sql 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、パスワードは TIGER であると仮定しています。

SQL*Plus を使用してデモンストレーション・プログラムを作成する場合の詳細は、[4-3 ページの「デモンストレーション表」](#)を参照してください。また、デモンストレーション・プログラムの詳細は、『Oracle Pro*C/C++ プリコンパイラ・プログラマーズ・ガイド』を参照してください。

Make ファイル \$ORACLE_HOME/precomp/demo/proc/demo_proc.mk は、デモンストレーション・プログラムの作成に必要です。たとえば、sample1 というデモンストレーション・プログラムをプリコンパイル、コンパイルまたはリンクする場合は、次のように入力します。

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

また、次の例は構文が明示的になっているだけで、同様の結果になります。

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

デフォルトでは、すべてのプログラムは、クライアント共有ライブラリ \$ORACLE_HOME/lib/libclntsh.so に動的にリンクされます。

すべての Pro*C/C++ の C デモンストレーション・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_proc.mk samples
```

すべての Pro*C/C++ の C++ デモンストレーション・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_proc.mk cppsamples
```

すべての Pro*C/C++ の Object デモンストレーション・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_proc.mk object_samples
```

デモンストレーション・プログラムによっては、実行する際に \$ORACLE_HOME/precomp/demo/sql にある SQL スクリプトが必要な場合があります。このようなデモンストレーション・プログラムを作成し、SQL スクリプトを実行するには、コマンド行に make マクロ引数 RUNSQL=run を追加する必要があります。たとえば、calldemo デモンストレーション・プログラムを作成し、必要な \$ORACLE_HOME/precomp/demo/sql/calldemo.sql スクリプトを実行する場合は、次のように入力します。

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

また、Object デモンストレーション・プログラムを作成し、必要な 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 object_samples RUNSQL=run
```

必要に応じて SQL スクリプトを手動で実行することもできます。

ユーザー・プログラム

Make ファイル `$ORACLE_HOME/precomp/demo/proc/demo_proc.mk` は、ユーザー・プログラムを作成する場合に使用します。ユーザー・プログラムを `demo_proc.mk` にリンクする場合の構文は、次のとおりです。

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

たとえば、Pro*C/C++ のソース `myprog.pc` からプログラム `myprog` を作成する場合、ソースおよび実行ファイルの形式によって、次のいずれかを入力します。

C ソースの場合で、クライアント共有ライブラリに動的にリンクする場合

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

C ソースの場合で、静的にリンクする場合

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

C++ ソースの場合で、クライアント共有ライブラリに動的にリンクする場合

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

C++ ソースの場合で、静的にリンクする場合

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

DG/UX Intel、NCR MP-RAS または Solaris Intel で共有ライブラリを使用する場合の詳細は、DG、NCR または Sun Microsystems からの各プラットフォーム用のドキュメントを参照してください。

Pro*COBOL (DG/UX Intel、 SCO UnixWare および NCR MP-RAS の場合)

今回のリリースの Pro*COBOL には、次の 2 つのバージョンがあります。

- Pro*COBOL 8.0.6
- Pro*COBOL 1.8.28

表 5-3 に、この 2 つのバージョンのネーミング方法の違いを示します。

表 5-3 Pro*COBOL のネーミング方法の違い

	Pro*COBOL 8.0.6	Pro*COBOL 1.8.28
実行ファイル	procob	procobl8
デモ・ディレクトリ	procob2	procob
Make ファイル (MicroFocus COBOL)	demo_procob.mk	demo_procobl8.mk

Pro*COBOL では、静的リンク、動的リンク、または動的読み込みプログラムをサポートしています。動的リンク・プログラムでは、Oracle クライアント共有ライブラリ `$ORACLE_HOME/lib/libclntsh.so` が使用されます。動的読み込みプログラムでは、`rtsora` 実行ファイルが使用されます。

Pro*COBOL 8.0.6 の追加情報は、README ファイル

`$ORACLE_HOME/precomp/doc/procob2/readme.doc` を参照してください。

Pro*COBOL 1.8.28 の追加情報は、README ファイル

`$ORACLE_HOME/precomp/doc/prolx/readme.txt` を参照してください。

Pro*COBOL の管理

システム構成ファイル

Pro*COBOL 8.0.6 のシステム構成ファイルは、

`$ORACLE_HOME/precomp/admin/pcbcfg.cfg` です。

Pro*COBOL 1.8.28 のシステム構成ファイルは、

`$ORACLE_HOME/precomp/admin/pcccob.cfg` です。

環境変数

MicroFocus COBOL コンパイラ

MicroFocus COBOL コンパイラには、環境変数 COBDIR および LD_LIBRARY_PATH を設定する必要があります。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。次に例を示します。

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

LD_LIBRARY_PATH には、\$COBDIR/coblib ディレクトリを設定します。たとえば、次のように入力して \$COBDIR/coblib を LD_LIBRARY_PATH に追加します。

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

LD_LIBRARY_PATH に \$COBDIR/coblib が設定されていないと、プログラムをコンパイルしたときに次のエラーが発生します。

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

Pro*COBOL の使用

Pro*COBOL を使用する前に、適切なバージョンの COBOL 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。必要なバージョンについては、『Oracle8 for Intel UNIX (DG/UX Intel, NCR MP-RAS, SCO UnixWare, Solaris Intel) インストレーション・ガイド』の第 1 章を参照してください。

Oracle ランタイム・システム

Oracle では、動的読み込み Pro*COBOL プログラムを実行するために、rtsora という専用のランタイム・システムが提供されています。rtsora ランタイム・システムは、動的読み込み Pro*COBOL プログラムを実行する際に、MicroFocus の cobrun ランタイム・システムのかわりに使用します。cobrun で Pro*COBOL プログラムを実行すると、次のエラーが発生します。

```
$ 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
```

デモンストレーション・プログラム

デモンストレーション・プログラムは、Pro*COBOL プリコンパイラのさまざまな機能を参照するために提供されています。すべてのプログラムは、Pro*COBOL のバージョンによって、`$ORACLE_HOME/precomp/demo/procob` または `$ORACLE_HOME/precomp/demo/procob2` に保存されています。すべてのプログラムでは、`$ORACLE_HOME/sqlplus/demo/demobld.sql` で作成したデモンストレーション表は SCOTT スキーマにあり、パスワードは TIGER であると仮定しています。SQL*Plus を使用してデモンストレーション・プログラムを作成する場合の詳細は、[4-3 ページの「デモンストレーション表」](#)を参照してください。

参照： デモンストレーション・プログラムの詳細は、『Oracle Pro*COBOL プリコンパイラ・プログラマーズ・ガイド』を参照してください。

デモンストレーション Make ファイルは、サンプル・プログラムを作成する場合に必要です。Pro*COBOL 8.0.6 のデモンストレーション Make ファイルは、`$ORACLE_HOME/precomp/demo/procob2/demo_procob.mk` です。Pro*COBOL 1.8.28 のデモンストレーション Make ファイルは、`$ORACLE_HOME/precomp/demo/procob/demo_procob18.mk` です。たとえば、Pro*COBOL 8.0.6 のデモンストレーション・プログラム `sample1` をプリコンパイル、コンパイルおよびリンクするには、次のコマンドを使用します。

```
$ cd $ORACLE_HOME/precomp/demo/procob2
$ make -f demo_procob.mk sample1
```

また、次の例は構文が明示的になっているだけで、同様の結果になります。

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

デフォルトでは、すべてのプログラムは、クライアント共有ライブラリ `$ORACLE_HOME/lib/libclntsh.so` に動的にリンクされます。

すべての Pro*COBOL デモンストレーション・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 samples
```

rtsora で使用する動的読み込みプログラム `sample1.gnt` を作成するには、次のように入力します。

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

その後、次のように入力して、rtsora でプログラムを実行します。

```
$ rtsora sample1.gnt
```

デモンストレーション・プログラムによっては、実行する際に

\$ORACLE_HOME/precomp/demo/sql にある SQL スクリプトが必要な場合があります。このようなデモンストレーション・プログラムを作成し、SQL スクリプトを実行するには、コマンド行に make マクロ引数 RUNSQL=run を追加する必要があります。

たとえば、sample9 デモンストレーション・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場合は、次のように入力します。

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

必要に応じて SQL スクリプトを手動で実行することもできます。

ユーザー・プログラム

デモンストレーション Make ファイルは、ユーザー・プログラムを作成する場合に使用します。ご使用の Pro*COBOL のバージョンおよび COBOL コンパイラに応じて、適切な Make ファイルを使用してください。ユーザー・プログラムをデモンストレーション Make ファイルにリンクする構文は、次のとおりです。

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

たとえば、Pro*COBOL のソース myprog.pco からプログラム myprog を作成する場合、実行ファイルの形式および共有ライブラリの使用方法によって、次のいずれかのコマンドを実行します。

クライアント共有ライブラリに動的にリンクする実行ファイルの場合

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

クライアント共有ライブラリを使用せずに静的にリンクする実行ファイルの場合

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

rtsora で使用できる動的読み込みモジュールの場合

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

FORMAT プリコンパイラ

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。
FORMAT=ANSI (デフォルト) を指定した場合、カラム 1 ~ 6 はオプションの順序番号、カラム 7 はコメントまたは継続行を示す標識です。さらに、カラム 8 ~ 11 は段落の名前で、カラム 12 ~ 72 が文となります。

FORMAT=TERMINAL を指定した場合、カラム 1 ~ 6 は削除され、カラム 7 が左端のカラムになります。

Pro*FORTRAN (DG/UX Intel の場合)

Pro*FORTRAN 1.8.28 の追加情報は、README ファイル
\$ORACLE_HOME/precomp/doc/prolx/readme.txt を参照してください。

Pro*FORTRAN の管理

システム構成ファイル

Pro*FORTRAN のシステム構成ファイルは、
\$ORACLE_HOME/precomp/admin/pccfor.cfg です。

Pro*FORTRAN の使用

Pro*FORTRAN を使用する前に、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。必要なバージョンについては、『Oracle8 for Intel UNIX (DG/UX Intel, NCR MP-RAS, SCO UnixWare, Solaris Intel) インストレーション・ガイド』の第 1 章を参照してください。

デモンストレーション・プログラム

Pro*FORTRAN プリコンパイラのさまざまな機能を参照するために、デモンストレーション・プログラムが提供されています。すべてのプログラムは、
\$ORACLE_HOME/precomp/demo/profor にあります。また、すべてのデモンストレーション・プログラムでは、\$ORACLE_HOME/sqlplus/demo/demobld.sql 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、パスワードは TIGER であると仮定しています。

SQL*Plus を使用してデモンストレーション・プログラムを作成する場合の詳細は、[4-3 ページの「デモンストレーション表」](#)を参照してください。

参照： デモンストレーション・プログラムの詳細は、『Oracle プリコンパイラ・ガイド Pro*FORTRAN サプリメント』を参照してください。

Make ファイル `$ORACLE_HOME/precomp/demo/profor/demo_profor.mk` は、デモンストレーション・プログラムの作成に必要です。たとえば、`sample1` というデモンストレーション・プログラムをプリコンパイル、コンパイルまたはリンクする場合は、次のように入力します。

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

また、次の例は構文が明示的になっているだけで、同様の結果になります。

```
$ make -f demo_profor.mk build FORS=sample1.pfo EXE=sample1 OBJS=sample1.o
```

デフォルトでは、すべてのプログラムは、クライアント共有ライブラリ `$ORACLE_HOME/lib/libclntsh.so` に動的にリンクされます。

すべての Pro*FORTRAN デモンストレーション・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_profor.mk samples
```

デモンストレーション・プログラムによっては、実行する際に `$ORACLE_HOME/precomp/demo/sql` にある SQL スクリプトが必要な場合があります。このようなデモンストレーション・プログラムを作成し、SQL スクリプトを実行するには、コマンド行に make マクロ引数 `RUNSQL=run` を追加する必要があります。たとえば、`sample11` デモンストレーション・プログラムを作成し、必要な `$ORACLE_HOME/precomp/demo/sql/sample11.sql` スクリプトを実行する場合は、次のように入力します。

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

必要に応じて SQL スクリプトを手動で実行することもできます。

ユーザー・プログラム

Make ファイル `$ORACLE_HOME/precomp/demo/profor/demo_profor.mk` は、ユーザー・プログラムを作成する場合に使用します。ユーザー・プログラムを `demo_profor.mk` にリンクする構文は、次のとおりです。

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

たとえば、Pro*FORTRAN のソース `myprog.pfo` からプログラム `myprog` を作成する場合、実行ファイルの形式によって、次のいずれかのコマンドを実行します。

クライアント共有ライブラリに動的にリンクする実行ファイルの場合

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

静的にリンクする実行ファイルの場合

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


Oracle コール・インタフェース

Oracle コール・インタフェースの使用

Oracle コール・インタフェース (OCI) を使用する前に、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。必要なバージョンについては、『Oracle8 for Intel UNIX (DG/UX Intel, NCR MP-RAS, SCO UnixWare, Solaris Intel) インストール・ガイド』の第 1 章を参照してください。

デモンストレーション・プログラム

デモンストレーション・プログラムは、OCI のさまざまな機能を参照するために提供されています。デモンストレーション・プログラムには、C および C++ の 2 種類があります。すべてのデモンストレーション・プログラムは、`$ORACLE_HOME/rdbms/demo` にあります。ほとんどのデモンストレーション・プログラムでは、`$ORACLE_HOME/sqlplus/demo/demobld.sql` 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、パスワードは TIGER であると仮定しています。

SQL*Plus を使用してデモンストレーション・プログラムを作成する場合の詳細は、[4-3 ページの「デモンストレーション表」](#)を参照してください。

参照： デモンストレーション・プログラムの詳細は『Oracle コール・インタフェース・プログラマーズ・ガイド』を、各プログラムの詳細はプログラム・ソースを参照してください。

Make ファイル `$ORACLE_HOME/rdbms/demo/demo_rdbms.mk` は、デモンストレーション・プログラムを作成する場合に必要です。たとえば、`cdemo1` というデモンストレーション・プログラムを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_rdbms.mk cdemo1
```

また、次の例は構文が明示的になっているだけで、同様の結果になります。

```
$ make -f demo_rdbms.mk build OBJS=cdemo1.o EXE=cdemo1
```

デフォルトでは、すべてのプログラムは、クライアント共有ライブラリ
\$ORACLE_HOME/lib/libclntsh.so に動的にリンクされます。

すべての OCI の C デモンストレーション・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_rdbms.mk demos
```

すべての OCI の C++ デモンストレーション・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_rdbms.mk c++demos
```

注意： Solaris Intel の場合、C++ プログラムのリンク中に、次のエラー・メッセージが表示される場合があります。

```
ld: fatal: library -lsunmath: not found
ld: fatal: library -lC: not found
ld: fatal: library -lC_mtstubs: not found
ld: fatal: library -lcx: not found
```

この場合は、出力されたライブラリが存在するディレクトリを
LD_LIBRARY_PATH の設定に追加してください。

たとえば、Solaris コンパイラ 5.0 を使用している場合、次のように入力して、/opt/SUNWsp/SC5.0/lib ディレクトリを LD_LIBRARY_PATH の設定に追加します。

```
$ setenv LD_LIBRARY_PATH \
${LD_LIBRARY_PATH}:/opt/SUNWsp/SC5.0/lib
```

デモンストレーション・プログラムによっては、プログラムを実行する前に、
\$ORACLE_HOME/rdbms/demo にある SQL スクリプトを実行する必要があります。ほとんどの場合、SQL スクリプト名はプログラム名と同じで、拡張子は .sql です。たとえば、プログラム oci02 の SQL スクリプトは、oci02.sql です。

プログラムの先頭にコメントがある場合は、そのコメントから必要な SQL スクリプトを判断します。

ユーザー・プログラム

Make ファイル `$ORACLE_HOME/rdbms/demo/demo_rdbms.mk` は、ユーザー・プログラムを作成する場合に必要です。ユーザー・プログラムを `demo_rdbms.mk` にリンクする構文は、次のとおりです。

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

たとえば、C のソース `myprog.c` からプログラム `myprog` を作成する場合、実行ファイルの形式によって、次のいずれかのコマンドを実行します。

C ソースの場合で、クライアント共有ライブラリに動的にリンクする場合

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

C ソースの場合で、静的にリンクする場合

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

C++ のソース `myprog.cc` からプログラム `myprog` を作成する場合は、次のいずれかのコマンドを実行します。

C++ ソースの場合で、クライアント共有ライブラリに動的にリンクする場合

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

C++ ソースの場合で、静的にリンクする場合

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

Oracle プリコンパイラ、Oracle コール・インタフェースのリンクおよび Make ファイル

カスタム Make ファイル

この章で説明している各製品には、オラクル社から提供されている Make ファイル `demo_product.mk` を使用してユーザー・プログラムをリンクすることをお勧めします。この Make ファイルを修正する必要がある場合、またはカスタム Make ファイルを使用する場合は、次のことに注意してください。

- 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を解決するために、Oracle ライブラリはリンク行に 2 回以上追加されます。これには、次の 2 つの理由があります。
 1. Oracle ライブラリは、相互に参照し合います。つまり、ライブラリ A の関数はライブラリ B の関数を呼び出し、ライブラリ B の関数はライブラリ A の関数を呼び出します。
 2. Intel UNIX リンカーは、1 パスのリンカーです。つまり、リンカーは、リンク行に出現した時点で 1 回だけライブラリを検索します。
- リンク行に独自のライブラリを追加する場合は、リンク行の最初または最後に追加します。
ユーザー・ライブラリは、Oracle ライブラリの間には置かないでください。
- `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、マクロおよび接尾辞の処理について、ご使用のシステムで提供されている `make` ユーティリティ (`/usr/ccs/bin/make`) との違いに注意してください。Oracle の Make ファイルは、すでにテスト済みで、ご使用のシステムの `make` ユーティリティでサポートされています。
- 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、現在のリリースで提供されている Make ファイル `ddemo_product.mk` を必ず使用してください。

未定義シンボル

プログラムのリンク時に共通するエラーに、次のような未定義シンボルがあります。

```
$ 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
```

オラクル社が提供している `symfind` という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・ファイルの場所を調べるのに役立ちます。次に、シンボル `sqlcex` が定義されている場所を調べる `symfind` の出力例を示します。

```
$ symfind sqlcex
```

```
SymFind - Find Symbol <sqlcex> in <**>.a, .o, .so
```

```
-----
```

```
Command:          /u01/app/oracle/product/8.0.6/bin/symfind sqlcex
Local Directory:   /u01/app/oracle/product/8.0.6
Output File:       (none)
Note:              I do not traverse symbolic links
                   Use '-v' option to show any symbolic links
```

```
Locating Archive and Object files ...
```

```
[11645] |      467572|      44|FUNC|GLOB|0    |8        |sqlcex
*****
./lib/libclntsh.so
[35]    |            0|      44|FUNC|GLOB|0    |5        |sqlcex
*****
./lib/libsql.a
```

Oracle ライブラリを使用した静的リンクおよび動的リンク

プリコンパイラおよび OCI アプリケーションは、静的および動的に Oracle ライブラリにリンクできます。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つの実行ファイル・プログラムにリンクされます。そのため、アプリケーションの実行ファイルのサイズは非常に大きくなります。

動的リンクの場合、実行コードはある部分は実行ファイル・プログラムに存在し、また別の部分は実行時にアプリケーションと動的にリンクされるライブラリに存在します。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呼びます。動的リンクには、主に次の利点があります。

1. 必要なディスク領域が少なく済む
異なるアプリケーション、または同一アプリケーションからの異なる呼出しであっても、同一の共有ライブラリまたは動的ライブラリを使用できます。その結果、必要なディスク領域を削減できます。

2. 必要なメイン・メモリーが少なくて済む
異なるアプリケーションで、同一の共有ライブラリ・イメージまたは動的ライブラリ・イメージ（メモリ内コピー）を共有できます。つまり、1つのライブラリをメイン・メモリーに1回だけ読み込めば、複数のアプリケーションでそのライブラリを使用できます。その結果、必要なメイン・メモリーを削減できます。

Oracle 共有ライブラリ

Oracle 共有ライブラリは、`$ORACLE_HOME/lib/libclntsh.so` です。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、デフォルトで Oracle 共有ライブラリが使用されます。

環境変数 `LD_LIBRARY_PATH` は、プロセスの起動時にランタイム・ローダーが Oracle 共有ライブラリを検索できるように設定する必要があります。実行ファイルの起動時に次のエラーが発生する場合は、`LD_LIBRARY_PATH` に、Oracle 共有ライブラリがあるディレクトリを設定してください。

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

次のように入力して、`LD_LIBRARY_PATH` を設定します。

```
% setenv LD_LIBRARY_PATH $ORACLE_HOME/lib
```

Oracle 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。Oracle 共有ライブラリを再作成する必要がある場合は、Oracle 共有ライブラリを使用している SQL*Plus、Recovery Manager などのすべてのクライアント・アプリケーションを終了し、`oracle` ユーザーでログインして、次のように入力します。

```
% cd $ORACLE_HOME/rdbms/lib
% make -f ins_rdbms.mk client_sharedlib
```

シグナル・ハンドラの使用法

ここでは、2 タスク通信で Oracle8 が使用するシグナル、および独自のシグナル・ハンドラを設定する方法について説明します。

シグナル

シグナルは、データベースに接続するとユーザー・プロセスに作成され、切断すると削除されます。

表 5-4 に、Oracle8 の 2 タスク通信で使用するシグナルを示します。

表 5-4 2 タスク通信のためのシグナル

SIGCONT	アウトオブバンド・ブレークをユーザー・プロセスから Oracle プロセスへ送信する場合に、パイプ 2 タスク・ドライバが使用します。
SIGINT	ユーザーの割り込み要求を検出する場合に、すべての 2 タスク・ドライバが使用します。SIGINT は、Oracle プロセスではなく、ユーザー・プロセスが受け取ります。
SIGPIPE	通信チャネルでファイルの終わりを検出するために、パイプ・ドライバが使用します。パイプへの書き込み時に、読み込みプロセスが存在していなければ、SIGPIPE シグナルが書き込みプロセスに送信されます。SIGPIPE は、Oracle プロセスおよびユーザー・プロセスが受け取ります。
SIGCLD	パイプ・ドライバが使用します。SIGCLD は SIGPIPE と似ていますが、ユーザー・プロセスに適用され、Oracle プロセスには適用されません。Oracle プロセスが異常終了すると、UNIX カーネルはユーザー・プロセスに SIGCLD を送ります（サーバー・プロセスが異常終了しているかどうかを調べるには、シグナル・ハンドラで wait() を使用します）。SIGCLD は、Oracle プロセスではなく、ユーザー・プロセスが受け取ります。
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、このシグナルが送られます。SIGTERM は、ユーザー・プロセスではなく、Oracle プロセスが受け取ります。
SIGIO	ネットワーク・イベントの着信を示すために、Oracle Net8 のプロトコル・アダプタが使用します。
SIGURG	ユーザー・プロセスから Oracle プロセスへアウトオブバンド・ブレークを送信する場合に、Oracle Net8 TCP/IP ドライバが使用します。

ここに示されているシグナルは、Pro*C またはその他のプリコンパイラ・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 1 つのシグナル・ハンドラを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。その他のシグナルには、必要なだけのシグナル・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。

シグナル・ルーチンの例

次に、独自のシグナル・ルーチンおよび受取りルーチンを設定する方法を示します。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。

```
/* user side interrupt set */
word osnsui( /*_ word *handlp, void (*astp), char * ctx, _*/)
/*
** osnsui: Operating System dependent Network Set
**User-side
```

```

** Interrupt. Add an interrupt handling procedure
**astp.
** Whenever a user interrupt(such as a ^C) occurs,
**call astp
** with argument ctx. Put in *handlp handle for this
**handler so that it may be cleared with osncui.
** Note that there may be many handlers; each should
** be cleared using osncui. An error code is
** returned if an error occurs.
*/

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

次に、アプリケーション・プログラムで `osnsui()` および `osncui()` を使用する場合は、テンプレートを示します。

```

/*
** My own user interrupt handler.
*/
void sig_handler()
{
    ...
}

main(argc, argv)
int arc;
char **argv;
{
    int handle, err;
    ...

    /* set up my user interrupt handler */
    if (err = osnsui(&handle, sig_handler, (char *) 0))
    {
        /* if the return value is non-zero, an error has occurred
        Do something appropriate here. */
        ...
    }
}
```



```
...
/* clear my interrupt handler */
if (err = osncui(handle))
{
/* if the return value is non-zero, an error has occurred
Do something appropriate here. */
...
}
...
}
```

XA 機能

TP モニター XA アプリケーションを作成する場合、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、リンク行で Oracle のクライアント 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`）を使用する場合のみ必要です。

Oracle8 Server では、Oracle7 リリース 7.1.6 の XA コールをサポートしていません（リリース 7.3 の XA コールはサポートしています）。そのため、リリース 7.1.6 の XA コールを使用する TP モニター XA アプリケーションは、Oracle8 の XA ライブラリにリンクする必要があります。Oracle8 の XA コールは、共有ライブラリ `$ORACLE_HOME/lib/libclntsh.so` および静的ライブラリ `$ORACLE_HOME/lib/libclient.a` の両方で定義されています。

Oracle Net8 の構成

- 参照ドキュメント
- 主な Net8 製品および特徴
- Oracle Net8 Protocol Adapter
- BEQ Protocol Adapter
- IPC Protocol Adapter
- RAW Protocol Adapter
- TCP/IP Protocol Adapter
- SPX/IPX Protocol Adapter (DG/UX Intel、SCO UnixWare および Solaris Intel の場合)
- Oracle Enterprise Manager (OEM) Intelligent Agent
- Oracle Advanced Networking Option

参照ドキュメント

Oracle Net8 の特徴の詳細は、次のドキュメントを参照してください。

- 『Oracle Net8 管理者ガイド』
- 『Oracle Networking Quick Reference Card for Net8』
- 『Oracle Advanced Networking Option 管理者ガイド』
- 『Oracle Security Server ガイド』
- 『Oracle Cryptographic Toolkit Programmer's Guide』

README ファイルへの補足情報

[表 6-1](#) に、各バンドル製品の README ファイルの保存場所を示します。README ファイルには、前回のリリースからの変更情報が記載されています。

表 6-1 Oracle 製品の README ファイルの保存場所

製品	README ファイル
Net8	\$ORACLE_HOME/network/doc/README.Net8
Advanced Networking Option	\$ORACLE_HOME/network/doc/README.ANO
Oracle Intelligent Agent	\$ORACLE_HOME/network/doc/README.oemagent
Oracle Security Server	\$ORACLE_HOME/network/doc/README.OSS
Oracle Names Server	\$ORACLE_HOME/network/install/names/doc/README.doc
Oracle SPX/IPX Protocol Adapter	\$ORACLE_HOME/network/install/spxpa/doc/README.doc
Oracle TCP/IP Protocol Adapter	\$ORACLE_HOME/network/install/tcppa/doc/README.doc

主な Net8 製品および特徴

参照： サンプル・ファイルの詳細は、『Oracle Net8 管理者ガイド』を参照してください。

Net8 ファイルおよびユーティリティ

Net8 の構成ファイルの保存場所

Oracle Net8 および Connection Manager のグローバル・ファイルは、デフォルトでは /etc ディレクトリ (DG/UX Intel の場合) または /var/opt/oracle ディレクトリ (NCR MP-RAS、SCO UnixWare および Solaris Intel の場合) に保存されています。

Oracle Net8 および Connection Manager は、構成ファイルを次の順に検索します。

1. 環境変数 TNS_ADMIN に設定したディレクトリ (設定した場合)
2. DG/UX Intel の場合は /etc ディレクトリ、NCR MP-RAS、SCO UnixWare および Solaris Intel の場合は /var/opt/oracle ディレクトリ
3. \$ORACLE_HOME/network/admin

ファイルがデフォルトのディレクトリにない場合、すべてのネットワーク・ユーザーの起動ファイルの環境変数 TNS_ADMIN で、異なるディレクトリを設定します。

C シェルの場合、次のように入力します。

```
% setenv TNS_ADMIN new_default
```

システム・レベルの構成ファイルごとに、ローカルのプライベート構成ファイル (ユーザーのホーム・ディレクトリに保存されている) があります。プライベート・ファイルの設定は、システム・レベル・ファイルの設定を上書きします。sqlnet.ora のプライベート構成ファイルは \$HOME/.sqlnet.ora です。tnsnames.ora のプライベート構成ファイルは、\$HOME/.tnsnames.ora です。これらのファイルの構文は、対応するシステム・ファイルの構文と同じです。

サンプル構成ファイル

cman.ora、listner.ora、names.ora、sqlnet.ora および tnsnames.ora のサンプル構成ファイルは、\$ORACLE_HOME/network/admin/samples に保存されています。

アダプタ・ユーティリティ

インストールされている Oracle Net8 アダプタを表示するには、次のように入力します。

```
% adapters
```

指定した実行ファイルにリンクされているアダプタを表示するには、次のように入力します。

```
% adapters executable
```

たとえば、次のコマンドを実行すると、Oracle 実行ファイルにリンクされているアダプタが表示されます。

```
% adapters oracle
Protocol Adapters linked with oracle are:
    BEQ Protocol Adapter
    IPC Protocol Adapter
    TCP/IP Protocol Adapter
    RAW Protocol Adapter
Net8 Naming Adapters linked with oracle are:
    Oracle TNS Naming Adapter
    Oracle Naming Adapter
Advanced Networking Option/Network Security products linked with oracle are:
    Oracle Security Server Authentication Adapter
```

Oracle Connection Manager

参照： Oracle Connection Manager の詳細は、『Oracle Net8 管理者ガイド』を参照してください。

マルチスレッド・サーバー

参照： マルチスレッド・サーバーの詳細は、『Oracle8 Server 概要』および『Oracle8 Server 管理者ガイド』を参照してください。

Oracle Names

参照： Oracle Names の詳細は、『Oracle Net8 管理者ガイド』を参照してください。

Net8 Assistant

注意： 今回のリリースの NCR MP-RAS では、Net8 Assistant はサポートされていません。

Net8 Assistant (`$ORACLE_HOME/bin/net8asst.sh`) を使用する場合、DG/UX Intel、Solaris Intel および SCO UnixWare では Java 1.1.7 が必要です。Net8 Assistant を使用するには、Java がシステムにインストールされている必要があります。

Net8 Assistant のコマンド・スクリプトを実行すると、その他の Java 製品がシステムにインストールされていても、JRE 1.1.7 (`$ORACLE_HOME/network/jre11/bin/java`) とともに提供されている Java のコマンド・スクリプトが実行されます。

DG/UX Intel で Net8 Assistant を使用するには、環境変数 `JAVA_HOME`、`LD_LIBRARY_PATH` および `CLASSPATH` を正しく設定する必要があります。Solaris Intel および SCO UnixWare では、`JAVA_HOME` の設定のみが必要です。

次に、DG/UX Intel の例を示します。

JDK が `/usr/opt/jdk` にインストールされていて、環境変数を `ksh` で設定する場合は、次のように入力します。

```
export JAVA_HOME=/usr/opt/jdk
export CLASSPATH=$JAVA_HOME/lib/classes.zip:$JAVA_HOME/lib/rt.jar
export LD_LIBRARY_PATH=$JAVA_HOME/lib/dgux_x86/native_threads
```

次に、Solaris Intel の例を示します。

JDK が `/usr/java` にインストールされ、環境変数を `ksh` で設定する場合は、次のように入力します。

```
export JAVA_HOME=/usr/java
```

参照： Net8 Assistant の詳細は、『Oracle Net8 管理者ガイド』を参照してください。

Oracle Net8 Protocol Adapter

Intel UNIX では、Net8 バージョン 8.0.6 に対して次のプロトコル・アダプタがサポートされています。

- BEQ Protocol Adapter
- IPC Protocol Adapter
- RAW Protocol Adapter
- TCP/IP Protocol Adapter
- SPX/IPX Protocol Adapter

注意： SPX/IPX は、DG/UX Intel、Solaris Intel および SCO UnixWare でサポートされています。今回のリリースの NCR MP-RAS では、サポートされていません。

TCP/IP または SPX/IPX Protocol Adapter をインストールする前に、適切なオペレーティング・システム・ソフトウェアがインストールおよび構成されていることを確認してください。インストールおよび構成の要件については、『Oracle8 for Intel UNIX (DG/UX Intel, NCR MP-RAS, SCO UnixWare, Solaris Intel) インストレーション・ガイド』を参照してください。BEQ および IPC Protocol Adapter に固有のオペレーティング・システム要件はありません。

IPC、TCP/IP または SPX/IPX Protocol Adapter には、それぞれのプロトコル固有の ADDRESS 指定があります。ADDRESS 指定は、Net8 構成ファイルおよび `init.ora` ファイルのデータベース初期化パラメータ `MTS_LISTENER_ADDRESS` を設定する場合に必要です。詳細は、この章の各プロトコル・アダプタの項を参照してください。

表 6-2 に、各プロトコル・アダプタの ADDRESS 指定を示します。

表 6-2 ADDRESS 指定

プロトコル・ アダプタ ADDRESS 指定	
BEQ	(ADDRESS = (PROTOCOL = BEQ) (PROGRAM = ORACLE_HOME/bin/oracle) (ARGV0 = oracleORACLE_SID) (ARGS = '(DESCRIPTION=(LOCAL=YES)(ADDRESS=(PROTOCOL=BEQ)))') (ENVS = 'ORACLE_HOME=ORACLE_HOME,ORACLE_SID=ORACLE_SID'))
IPC	(ADDRESS = (PROTOCOL=IPC) (KEY=key))
RAW	N/A
TCP/IP	(ADDRESS = (PROTOCOL=TCP) (HOST=hostname) (PORT=port_id))
SPX/IPX	(ADDRESS = (PROTOCOL=SPX) (SERVICE=servicename))

BEQ Protocol Adapter

BEQ Protocol Adapter の概要

BEQ Protocol Adapter は、通信メカニズムであり、また、プロセス起動メカニズムでもあります。サービス名を指定（コマンド行またはログイン画面でユーザーが直接指定するか、または TWO_TASK などの環境変数で間接的に指定する）しなかった場合、BEQ Protocol Adapter が使用されます。その場合、専用のサーバーが使用され、マルチスレッド・サーバーは使用されません。この専用サーバーは、BEQ Protocol Adapter によって自動的に起動された後、サーバー・プロセスが起動されて既存の SGA に割り当てられるのを待ちます。サーバー・プロセスが正常に起動されると、BEQ Protocol Adapter で、UNIX パイプを介してプロセス間通信ができるようになります。

BEQ Protocol Adapter が動作するためにネットワーク・リスナーは不要です。これは、アダプタがクライアント・ツールにリンクされていて、外部から操作しなくてもそのサーバー・プロセスを直接起動するためです。ただし、BEQ Protocol Adapter を使用できるのは、クライアント・プログラムおよび Oracle8 Server が同一のマシンに常駐する場合だけです。BEQ Protocol Adapter は常にインストールされ、すべてのクライアント・ツールおよび Oracle8 Server にリンクされています。

BEQ の ADDRESS 指定

ADDRESS のキーワードとその値の組合せが BEQ Protocol Adapter の接続パラメータです。パラメータの入力順に決まりはありません。

```
(ADDRESS =  
  (PROTOCOL = BEQ)  
  (PROGRAM = ORACLE_HOME/bin/oracle)  
  (ARGV0 = oracleORACLE_SID)  
  (ARGS = ' (DESCRIPTION=(LOCAL=YES) (ADDRESS=(PROTOCOL=BEQ))) ' )  
  (ENVS = 'ORACLE_HOME=ORACLE_HOME, ORACLE_SID=ORACLE_SID' )  
)
```

表 6-3 に、BEQ Protocol Adapter の接続パラメータの構文を示します。

表 6-3 BEQ Protocol Adapter の接続パラメータの構文

PROTOCOL	使用するアダプタを指定します。 値は beq で、大文字または小文字のどちらかで指定します。
PROGRAM	Oracle 実行ファイルのフルパス名を指定します。
ARGV0	ps コマンドで表示されるプロセスの名前を指定します。 推奨値は、oracleORACLE_SID です。
ARGS	' (DESCRIPTION=(LOCAL=YES) (ADDRESS=(PROTOCOL=BEQ))) '
ENVS	環境を指定します。ORACLE_HOME には、接続先データベースの ORACLE_HOME ディレクトリをフルパスで指定します。ORACLE_SID には、 接続先データベースのシステム識別子を指定します。

次に、BEQ ADDRESS の例を示します。

例 6-1 クライアントを指定するための BEQ ADDRESS

```
(ADDRESS =
  (PROTOCOL = BEQ)
  (PROGRAM = /u01/app/oracle/product/8.0.6/bin/oracle)
  (ARGV0 = oracleV806)
  (ARGS = ' (DESCRIPTION=(LOCAL=YES) (ADDRESS=(PROTOCOL=BEQ))) ' )
  (ENVS = 'ORACLE_HOME=/u01/app/oracle/product/8.0.6,ORACLE_SID=V806' )
)
```

通常、ADDRESS の指定は、接続記述子、構成ファイルなどを構成している内容の一部です。

IPC Protocol Adapter

IPC Protocol Adapter の概要

IPC Protocol Adapter は、クライアント・プログラムおよび Oracle8 Server が同一マシンに常駐する場合にのみ使用できるという点が、BEQ Protocol Adapter と似ています。IPC Protocol Adapter と BEQ Protocol Adapter の違いは、IPC Protocol Adapter が専用サーバーおよびマルチスレッド・サーバーで使用できることです。IPC Protocol Adapter が動作するには、ネットワーク・リスナーが必要です。IPC Protocol Adapter は常にインストールされ、すべてのクライアント・ツールおよび Oracle8 Server にリンクされています。

IPC Protocol Adapter では、Oracle7 リリース 7.1 以降、UNIX システムでの UNIX Domain Socket (IPC) ファイルの保存場所が変わりました。そのため、Oracle7 リリース 7.1 を Oracle8 と同じマシンにインストールし、2 つのインスタンス間で IPC 接続しようとする と失敗します。この問題を解決するには、IPC ファイルが以前に保存されていたディレクトリ (/var/tmp/o) および現在保存されているディレクトリ (/var/tmp/.oracle) 間にシンボリック・リンクを作成します。

IPC の ADDRESS 指定

ADDRESS のキーワードとその値の組合せが IPC Protocol Adapter の接続パラメータです。パラメータの入力順に決まりはありません。

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

表 6-4 に、IPC Protocol Adapter の接続パラメータの構文を示します。

表 6-4 IPC Protocol Adapter の接続パラメータの構文

PROTOCOL	使用するアダプタを指定します。 値は ipc で、大文字または小文字のどちらかで指定します。
KEY	データベースのサービス名またはデータベースの識別子 (SID) を指定します。

次に、IPC ADDRESS の例を示します。

例 6-2 クライアントを指定する IPC ADDRESS

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

通常、ADDRESS の指定は、接続記述子、構成ファイルなどを構成している内容の一部です。

RAW Protocol Adapter

データがクライアントとサーバー間で転送されると、Net8 は、そのヘッダー情報を各パケット（ネットワークを介して送信される情報のひとまとまり）に追加します。Raw Transport 機能によって、ネットワークを介する各パケットのヘッダー情報の送信を最小限に抑えることができます。

接続後、2 種類の情報（データおよびブレイク・ハンドリング）がネットワーク上に送信されます。接続パケットには、適切に接続するための Net8 のヘッダー情報が必要です。しかし接続後は、データを送受信するすべてのデータ・パケットや、接続を切断または再設定するパケットから Net8 のヘッダー情報は削除され、Net8 の NT および Oracle Protocol Adapter レイヤーを通過して、直接オペレーティング・システムに渡されます。データが流れるプロトコル・スタック・レイヤーの数が減り、ネットワーク上に送信されるデータのバイト数が減るため、接続のパフォーマンスが向上します。

この機能は、ユーザーが意識しなくても、必要な場合に有効になります。つまり、ヘッダー情報を送信しなくてもよい場合、ヘッダーは削除されます。たとえば、暗号化および認証では、各情報のパケットと一緒に特定の情報を送信する必要があります。そのため、この場合は、Raw Transport は有効になりません。

この機能を使用する場合、特別な構成はありません。Net8 では、ユーザーが意識しなくても、Raw Transport モードに切り替える必要があるかどうか判断され、必要な場合には切り替えられます。

TCP/IP Protocol Adapter

TCP/IP Protocol Adapter の概要

Oracle Net8 リスナー・ポートを定義するネットワーク上の各ノードの `/etc/services` ファイルで、Oracle Net8 リスナー用のポートを予約してください。ほとんどの場合、ポートは 1521 です。リスナー名とポート番号という形式で入力します。たとえば、次のように入力します。

```
listener      1521/tcp
```

この場合、*listener* は `listener.ora` で定義されているリスナーの名前を示します。
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。

TCP/IP の ADDRESS 指定

ADDRESS のキーワードとその値の組合せが TCP/IP Protocol Adapter の接続パラメータです。パラメータの入力順に決まりはありません。

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

表 6-5 に、TCP/IP Protocol Adapter の接続パラメータの構文を示します。

表 6-5 TCP/IP Protocol Adapter の接続パラメータの構文

PROTOCOL	使用するアダプタを指定します。 値は大文字でも小文字でもかまいません。デフォルトは <code>tcp</code> です。
HOST	ホスト名またはホスト IP アドレスを指定します。
PORT	TCP/IP のポートを指定します。 <code>/etc/services</code> ファイルで指定されている番号または名前を指定します。推奨値は、1521 です。

次に、MADRID ホストにクライアントを指定する TCP/IP ADDRESS の例を示します。

例 6-3 クライアントを指定する TCP/IP の ADDRESS

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

最後のフィールドは、PORT=listener というように名前を指定することもできます。通常、ADDRESS の指定は、接続記述子、構成ファイルなどを構成している内容の一部です。

SPX/IPX Protocol Adapter (DG/UX Intel、SCO UnixWare および Solaris Intel の場合)

注意： NCR MP-RAS では、SPX/IPX Protocol Adapter はサポートされていません。

SPX/IPX Protocol Adapter の help コマンドは、[表 6-6](#) を参照してください。

ntisbsdmd 同報通信デーモン

クライアントは名前を使用し、その名前を SPX アドレスに変換してサーバーを識別し、サーバーと通信します。Netware バインダリは、変換メカニズムを提供するディレクトリ・サービスです。サーバーがバインダリに登録されると、そのアドレスのバインダリを定期的に通知します。これは、サーバー通知プロトコル (SAP) を使用して行われます。

サーバーは、IPX データグラムで 60 秒ごとに SAP パケットを同報通信します。この SAP パケットには、関連するすべてのアドレス情報が含まれています。その後、どのクライアントからも、必要なサーバーのアドレスに最も近いサーバーに対して問合せできます。

Oracle SPX/IPX Protocol Adapter は、\$ORACLE_HOME/bin の ntisbsdmd 同報通信デーモンを使用して同報通信を行います。ntisbsdmd は、ntspcxctl ユーティリティによって起動および停止されます。

ntspcxctl ユーティリティ

ntspcxctl ユーティリティには、名前の登録と取消し、およびバインダリの問合せを実行する機能が含まれています。また、同報通信デーモンを起動または停止する場合に使用します。リスナーは、このデーモンを使用して、使用中のサービス名を自動登録します。

例 6-4 に、ntspcxctl ユーティリティのユーザーが複数の場合の例を示します。

例 6-4 ntspcxctl ユーティリティの使用

ntspcxctl ユーティリティは、コマンド行からコマンドを読み取ります。パラメータを指定していない場合、それらのパラメータを指定するためのプロンプトが表示されます。

ntspcxctl を起動するには、次のように入力します。

```
$ ntspcxctl
```

次のように表示されます。

```
ntspcxctl for Intel SVR4 UNIX: Version 8.0.6.0.0
Production on Fri Jul 3 11:42:00 1999
```

同報通信デーモンを起動するには、次のように入力します。

```
ntspcxctl> startup
```

次のように表示されます。

```
ntisbsdsm started at Fri Jul 3 11:43:47 1999
```

デーモンがすでに起動されている場合は、システム・メッセージが表示されます。

同報通信デーモンは、マシンの起動時に常に自動的に起動されるように設定する必要があります。/etc/inittab ファイルにエントリを追加して、デーモンの起動を自動化します。たとえば、システムの起動時に ntisbsdsm を起動するには、/etc/inittab に次の行を追加します。

```
ntspcxctl:2:once:/u/oracle/bin/ntisbsdsm &
```

この場合、/u/oracle は \$ORACLE_HOME へのフルパス名を示します。

テスト用の名前を登録するには、register および名前を入力します。

```
ntspcxctl> register stet
```

これによって ntisbsdsm が所有するソケットが作成され、登録されます。

次のメッセージが表示されます。

```
Name YYY successfully registered
YYY address 00eee045:000000000001:4454
```


ntisbsdsm の状態を表示するには、次のように入力します。

```
ntspxctl> status
```

次のメッセージが表示されます。

```
ntisbsdsm started at Fri Jul 3 11:43:47 1999
Tracing is off
Pid: 14784 YYY
```

次のように入力すると、状態をより簡潔に表示できます。

```
ntspxctl> summary
```

次のメッセージが表示されます。

```
ntisbsdsm started at Fri Jul 3 11:43:47 1999
Tracing is off
1 names are registered
```

SPX/IPX Protocol Adapter のコマンドの要約

表 6-6 に、SPX/IPX Protocol Adapter の help コマンドを示します。

表 6-6 help コマンド

register <i>name</i>	エントリを登録します。
remove <i>name</i>	エントリを取り消します。
shutdown [<i>force</i>]	ntisbsdsm を停止します。
startup	状態の要約を表示します。
traceon	トレースをアクティブにします。
traceoff	トレースを非アクティブにします。
status	状態の詳細情報を表示します。
getname <i>name</i> <i>hex_number</i>	ネーム・サービスの問合せを実行します。
exit	プログラムを終了します。
help [<i>command</i>]	コマンド情報を印刷します。
!	シェルを取り消します。

getname コマンド

getname コマンドは、Novell システムに名前を要求する場合に使用します。同報通信デーモンは必要ありません。

次のように入力します。

```
getname name servicetype
```

次のメッセージが表示されます。

```
getname name servicetype (address number_of_hops)
```

表 6-7 に、getname コマンドの構文を示します。

表 6-7 getname コマンドの構文

name	入力した名前です。
servicetype	Novell で割り当てられた番号です。Oracle の番号は 103 です。
address	入力した名前のアドレスです。
number_of_hops	宛先までのホップ数 (16 進数) です。値が 10 の場合は、名前が登録解除されたことを表します。SAP 問合せがサポートされていない場合、値は 0000 です。

付けることができるすべての名前を表示するには、次のように入力します。

```
getname * *
```

例 6-5 に、getname コマンドで名前を付ける例を示します。

例 6-5 getname コマンドの使用

```
ntspxctl> getname YYY *
YYY servertype x0103 address 00eee045:0000000000001:
4465 hops 0000
ntspxctl> getname * 103
LSNR servertype x0103 address 00eee053:0000000000001:
502c hops 0000
IBM6 servertype x0103 address 00eee058:0000000000001:
507f hops 0000
DESK servertype x0004 address 00eee055:0000000000001:
5451 hops 0000
DESK servertype x0107 address 00eee055:0000000000001:
5104 hops 0000
CXY4 servertype x009e address 00eee055:0000000000001:
5063 hops 0000
IBM2 servertype x0004 address 00eee057:0000000000001:
5451 hops 0000
```

ntisbsdsm を停止するには、次のように入力します。

```
ntspxctl> shutdown
```

このデーモンは、名前がまだ登録されている場合は停止しません。次のメッセージが表示されます。

```
1 names are registered
ntisbsdsm not stopped
```

名前を取り消すには、remove および名前を入力します。名前が YYY の場合の例を次に示します。

```
ntspxctl> remove YYY
```

次のメッセージが表示されます。

```
Name xxxremoved.
ntspxctl> shutdown
ntisbsdsm stopped
```

強制終了するには、次のように入力します。

```
ntspxctl> shutdown force
```

次のメッセージが表示されます。

```
ntisbsdsm stopped
```

SPX/IPX の ADDRESS 指定

SPX/IPX Protocol および Oracle SPX/IPX がシステムにインストール済みの場合、TNS 接続記述子を指定した SPX/IPX パラメータを使用して、SPX/IPX コミュニティ・ノードを識別できます。

ADDRESS のキーワードとその値の組合せが、SPX/IPX Protocol Adapter パラメータです。

```
(ADDRESS=
  (PROTOCOL=SPX)
  (SERVICE=service_name)
)
```

表 6-8 に、SPX/IPX Protocol Adapter 接続の構文を示します。

表 6-8 SPX/IPX Protocol Adapter 接続の構文

PROTOCOL	アダプタ名を指定します。SPX/IPX の場合、値は <code>spx</code> です。
SERVICE	ネットワーク上のアプリケーションを示す一意の名前を 30 文字以下で指定します。サービスには起動中に名前が付けられ、ネットワーク全体で使用できます。クライアントは、ネットワーク・ディレクトリのバイナリを検索を使用してこのサービスを参照します。

例 6-6 に、リモート・サーバー上にサービス MAILDB1 を指定する SPX/IPX ADDRESS を示します。

例 6-6 SPX/IPX Protocol Adapter 接続

```
(ADDRESS=
  (PROTOCOL=SPX)
  (SERVICE=MAILDB1)
)
```

通常、ADDRESS の指定は、接続記述子、構成ファイルなどを構成している内容の一部です。

Oracle Enterprise Manager (OEM) Intelligent Agent

エージェント・サービスの展開および自動構成

参照： 詳細は、『Oracle Enterprise Manager 構成ガイド』を参照してください。

tcl スクリプトのデバッグ

tcl スクリプトのデバッグ用に、実行ファイル `oratclsh` が用意されています。`oratclsh` を実行する前に、環境変数 `TCL_LIBRARY` に `$ORACLE_HOME/network/agent/tcl` を設定してください。

参照： 詳細は、『Oracle Enterprise Manager アプリケーション開発者ガイド』を参照してください。

Oracle Advanced Networking Option

.bak ファイル

Oracle Advanced Networking Option のインストール中に、3 つの .bak ファイル (`naet.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` に保存されています。これらのファイルは、Oracle Advanced Networking Option の削除（デインストール）中の再リンクに必要なので、削除しないでください。

セキュリティおよびシングル・サインオン

参照： セキュリティおよびシングル・サインオンの構成の詳細は、『Oracle Advanced Networking Option 管理者ガイド』を参照してください。

DCE 統合

参照： DCE 統合の詳細は、『Oracle Advanced Networking Option 管理者ガイド』を参照してください。

Intel UNIX での Oracle データ・カートリッジ・デモの実行

- データ・カートリッジ共通の要件
- Oracle8 Time Series Cartridge
- Oracle8 Visual Information Retrieval Cartridge
- Oracle8 Image Cartridge

データ・カートリッジ共通の要件

データ・カートリッジでは、インストール後にはサンプル・アプリケーションを組み込むために C コンパイラが必要です。

参照： Oracle データ・カートリッジの詳細は、『Oracle Enterprise Manager 構成ガイド』および各カートリッジのドキュメントを参照してください。

Oracle8 Time Series Cartridge

参照： 詳細は、『Oracle Time Series カートリッジ・ユーザーズ・ガイド』を参照してください。

Time Series Cartridge デモのインストール

Time Series Cartridge デモのファイルは、`$ORACLE_HOME/ord/ts/demo` に保存されています。このディレクトリの各サブディレクトリに、それぞれのデモが含まれています。

- `usage` デモでは、Time Series オブジェクトの組み込み方法および Time Series 機能をコールする方法を示します。最初に `usage` デモを実行して、他のデモで使用する Time Series スキーマのサンプルを作成します。
- `extend` デモでは、カスタマイズされた Time Series 機能の書き込みおよび追加方法を示します。
- `proc` デモでは、Pro*C を使用してカートリッジが保存されているデータへのアクセス方法を説明します。
- `oci` デモでは、Oracle コール・インタフェースを通してデータにアクセスする方法を説明します。
- `dev2k` デモには、カートリッジを使用してデータを取り出す Developer 2000 形式が含まれています。

各デモの詳細は、デモ・ディレクトリにある `README` ファイルを参照してください。

Oracle8 Visual Information Retrieval Cartridge

参照： 詳細は、『Oracle Visual Information Retrieval Cartridge ユーザーズ・ガイド』を参照してください。

デモの作成

1. 次のように入力して、Visual Information Retrieval (VIR) デモ・ディレクトリを作成します。

```
% svrmgrl
SVRMGR> connect internal;
SVRMGR> create or replace directory virdemodir
2> as '$ORACLE_HOME/ord/vir/demo';
```

2. 次のように入力して、ディレクトリの権限を PUBLIC に与えます。

```
SVRMGR> grant read on directory virdemodir to public with grant option;
SVRMGR> exit;
```

3. 次のように入力して、virdemo プログラムを作成します。

```
% cd $ORACLE_HOME/ord/vir/demo
% make -f demo_ordvir.mk virdemo
```

Oracle8 Image Cartridge

参照： 詳細は、『Oracle Image Cartridge ユーザーズ・ガイド』を参照してください。

デモの作成

1. 次のように入力して、Image Cartridge のデモ・ディレクトリを作成します。

```
% svrmgrl
SVRMGR> connect internal;
SVRMGR> create or replace directory imgdemodir
      2> as '$ORACLE_HOME/ord/img/demo';
```

2. 次のように入力して、ディレクトリの権限を PUBLIC に与えます。

```
SVRMGR> grant read on directory imgdemodir to public with grant option;
SVRMGR> exit;
```

3. 次のように入力して、imgdemo プログラムを作成します。

```
% cd $ORACLE_HOME/ord/img/demo
% make -f demo_ording.mk imgdemo
```

記号

\$
 使用, 2-24
?
 使用例, 2-10
@
 SORACLE_SID, 2-10
_editor
 設定, 4-6

数字

2 タスク
 アーキテクチャ
 セキュリティ, 2-21
 シグナル, 5-18

A

admswap (DG/UX Intel), 3-10
Advanced Networking Option, 6-19
 .bak ファイル, 6-19
afiedt.buf, 4-7

B

BEQ Protocol Adapter, 6-8
 ADDRESS, 6-8
 概要, 6-8
 接続パラメータの構文, 6-9
Bourne シェル・スクリプトのトレース, 2-9
BSD, 2-12
BSD-UNIX ファイル・システム・タイプ, 3-28

C

C
 Pro*C/C++, 5-4
CATPROC.SQL, 1-17
CDE ツール, 2-9
chmod コマンド
 例, 2-23
COBOL
 Pro*COBOL, 5-7
CONNECT INTERNAL
 セキュリティ, 2-23
COPY コマンド
 SQL*Plus, 4-8
coraenv, 2-9
CPU
 Parallel Server のチューニング, 3-39
CPU の使用状況
 シングルタスク・アーキテクチャ, 3-31

D

daemon ユーザー
 セキュリティ, 2-25
DBA_GROUP
 /etc/listener.ora ファイル, 2-25
DBA_GROUP_sid
 /etc/listener.ora ファイル, 2-25
dba グループ
 再リンク, 2-22
 メンバー, 2-21
DBA グループ ID
 キーワード, 2-25
 デフォルト以外の名前, 2-25

DBWR
チューニング, 3-24
DCE 統合, 6-20
DISPLAY
環境変数, 2-11

E

echo コマンド, 2-5
expst コマンド, 3-31

F

FORMAT プリコンパイラ
Pro*COBOL, 5-10, 5-11

G

GC パラメータ
GC_DB_LOCKS, 3-40
GC_FILES_TO_LOCKS, 3-40
getname コマンド, 6-16
glogin.sql, 4-2

H

HOME, 2-11

I

Image Cartridge, 7-4
デモの作成, 7-4
impst コマンド, 3-31
I/O
DBWR チューニング, 3-24
Parallel Server のチューニング, 3-38
チューニング, 3-22
ディスク要求キュー, 3-26
iostat, 3-9
IPC Protocol Adapter, 6-10
ADDRESS, 6-10
概要, 6-10
ipcs コマンド, 3-16
ireclen, 5-3

J

JAVA_HOME, 2-11

L

LANGUAGE
環境変数, 2-11
ld, 2-11
LD_LIBRARY_PATH
環境変数, 2-11
LDOPTS
環境変数, 2-11
LDPATH
環境変数, 2-11
logical reads, 3-43
login.sql, 4-2
LONG 列
チューニング, 3-13
LPDEST
環境変数, 2-11

M

Make, 5-6
MAXDATAFILES パラメータ, 2-35
MAXINSTANCES パラメータ, 2-35
MAXLOGFILES パラメータ, 2-35
MAXLOGHISTORY パラメータ, 2-35
MAXLOGMEMBERS パラメータ, 2-35
MicroFocus COBOL コンパイラ, 5-8
mpstat, 3-12

N

Net8
Advanced Networking Option, 6-19
IPC Protocol Adapter, 6-10
Net8 Assistant, 6-5
Oracle Connection Manager, 6-4
Oracle Enterprise Manager Intelligent Agent, 6-19
Oracle Names, 6-5
RAW Protocol Adapter, 6-11
README ファイル, 6-2
SPX/IPX Protocol Adapter, 6-13
TCP/IP Protocol Adapter, 6-12
アダプタ・ユーティリティ, 6-4

- 自動ログイン, 2-24
- 製品および特徴, 6-3
- ファイルおよびユーティリティ, 6-3
- プロトコル・アダプタ, 6-6
- マルチスレッド・サーバー, 6-4
- Net8 Assistant, 6-5
- Net8 の構成ファイル
 - 保存場所, 6-3
- NLS_LANG
 - 環境変数, 2-8
- ntisbsdmd 同報通信デーモン, 6-13
- ntspxcctl ユーティリティ, 6-14

O

- ORA_NLS
 - 環境変数, 2-8
- Oracle, 5-8
 - メモリーの使用量, 2-14
- Oracle Connection Manager, 6-4
- Oracle Enterprise Manager Intelligent Agent, 6-19
 - tcl スクリプトのデバッグ, 6-19
- Oracle Names, 6-5
- Oracle Parallel Server
 - 監視, 3-38
- Oracle Server
 - アカウント, 2-20
- ORACLE_HELP
 - 環境変数, 2-8
- ORACLE_HOME
 - ? の使用, 2-10
 - 環境変数, 2-8
- ORACLE_SID
 - 環境変数, 2-9
 - プロンプトの抑制, 2-6
- ORACLE_TERM
 - 環境変数, 2-9
- Oracle8 で使用する UNIX 環境変数
 - DISPLAY, 2-11
 - HOME, 2-11
 - LANG, 2-11
 - LD_LIBRARY_PATH, 2-11
 - LDOPTS, 2-11
 - LDPATH, 2-11
 - LPDEST, 2-11
 - PATH, 2-12
 - PRINTER, 2-12
 - SHELL, 2-12
 - TERM, 2-12
 - TMPDIR, 2-12
 - XENVIRONMENT, 2-12
- oracle アカウント
 - 認可の設定, 2-23
- Oracle 環境変数
 - EPC_DISABLED, 2-8
 - NLS_LANG, 2-8
 - ORA_NLS33, 2-8
 - ORACLE_BASE, 2-8
 - ORACLE_HELP, 2-8
 - ORACLE_HOME, 2-8
 - ORACLE_PATH, 2-8
 - ORACLE_SID, 2-9
 - ORACLE_TERM, 2-9
 - ORACLE_TERMINAL, 2-9
 - ORACLE_TRACE, 2-9
 - ORAENV_ASK, 2-9
 - TNS_ADMIN, 2-10
 - TWO_TASK, 2-10
- oracle グループ
 - 権限と実行プログラム, 2-21
- Oracle コール・インタフェース, 5-13
 - 使用, 5-13
 - デモンストレーション・プログラム, 5-13
 - ユーザー・プログラム, 5-15
- oracle ソフトウェア所有者, 2-20
 - 特殊アカウント, 2-20
- Oracle のシステム ID, 2-9
- Oracle プリコンパイラ、OCI のリンクおよび Make ファイル, 5-16
 - カスタム Make ファイル, 5-16
 - 未定義シンボル, 5-16
- Oracle ライブラリ
 - Oracle 共有ライブラリ, 5-18
 - 静的リンクおよび動的リンク, 5-17
- Oracle ランタイム・システム
 - Pro*COBOL, 5-8
- ORAENV_ASK, 2-9
 - 設定, 2-6
- oraenv ファイル
 - 説明, 2-5
 - データベース間の移動, 2-6
- orainst, 2-9
- orapwd コマンド, 2-26
- orecld, 5-3

P

Parallel Server

- 監視, 3-38
- リソースの競合, 3-38

PATH, 2-12

PL/SQL

- 大きいブロック, 3-20
- デモンストレーション
ロード, 2-30

PRINTER, 2-12

Pro*C/C++

- Make ファイル, 5-5, 5-6
- 管理, 5-4
- シグナル, 5-19
- システム構成ファイル, 5-4
- 使用, 5-4
- デモンストレーション・プログラム, 5-4
- ユーザー・プログラム, 5-6

Pro*COBOL, 5-7

- FORMAT プリコンパイラ, 5-10, 5-11
- Oracle ランタイム・システム, 5-8
- 環境変数, 5-7, 5-8
- 管理, 5-7
- システム構成ファイル, 5-7
- デモンストレーション・プログラム, 5-8, 5-9
- ネーミング方法の違い, 5-7
- ユーザー・プログラム, 5-10

PRODUCT_USER_PROFILE 表

- SQL*Plus, 4-3

pstat コマンド, 3-15

pupbld.sql, 4-3

R

RAW Protocol Adapter, 6-11

RAW デバイス, 2-37

- Oracle8 Parallel Server のインストール・デモンストレーション, 2-38
- RAW ディスク・パーティションの可用性, 2-38
- ガイドライン, 2-38
- 欠点, 2-37
- 使用基準, 2-37
- 設定, 2-38
- バッファ・キャッシュ・サイズ, 3-42

RAW パーティション, 3-28

README ファイル

- Net8, 6-2

REDO バッファ

- チューニング, 3-18

REDO ログ

- RAW デバイス, 3-23
- チューニング, 3-23
- バッファ・サイズのチューニング, 3-18
- バッファ・ラッチの競合, 3-35

root

- ユーザー, 2-20

S

s5 ファイル・システム・タイプ, 3-28

sar, 3-7

sar コマンド, 3-15, 3-43

Server Manager

- SHOW PARAMETERS, 2-15
- コマンド, 2-22
- セキュリティ, 2-22

SGA

- サイズの見積り, 3-16
- 再配置, 2-19
- チューニング, 3-16
- 物理メモリーへのロック, 3-17

SHELL, 2-12

SHUTDOWN コマンド, 2-23

- セキュリティ, 2-23

SIGCLD 2 タスク・シグナル, 5-19

SIGINT 2 タスク・シグナル, 5-19

SIGIO 2 タスク・シグナル, 5-19

SIGPIPE 2 タスク・シグナル, 5-19

SIGTERM 2 タスク・シグナル, 5-19

SIGURG 2 タスク・シグナル, 5-19

Site Profile

- SQL*Plus, 4-2

SPOOL コマンド

- SQL*Plus, 4-8
- 使用, 4-8

SPX/IPX Protocol Adapter, 6-13

- ADDRESS, 6-18
- getname コマンド, 6-16
- ntisbsdsm 同報通信デーモン, 6-13
- ntspxtcl ユーティリティ, 6-14
- コマンドの要約, 6-15

SQL

- SQL 文の再使用, 3-14
- 管理, 4-2

SQL*DBA , 2-9
 SHOW PARAMETERS , 2-15
SQL*Plus
 COPY コマンド , 4-8
 PRODUCT_USER_PROFILE 表 , 4-3
 Site Profile , 4-2
 SPOOL コマンド , 4-8
 UNIX コマンド , 4-7
 User Profile , 4-2
 エディタ , 4-6
 環境変数 , 4-7
 システム・エディタ , 4-6
 使用 , 4-6
 制限事項 , 4-8
 設定ファイルのサンプル , 4-2
 デフォルト・エディタ , 4-6
 デモンストレーション表 , 4-3
 ファイルの設定 , 4-2
 ヘルプ機能 , 4-4
 割込み , 4-8
SQL*Plus の使用 , 4-6
SQL*Plus への割込み , 4-8
STARTUP コマンド
 セキュリティ , 2-23
swap , 3-10
swap コマンド , 3-15
SYSDATE
 TZ , 2-13
System V , 2-11
SYSTEM アカウント
 権限 , 2-20
SYS アカウント
 権限 , 2-20

T

TCP/IP Protocol Adapter , 6-12
 ADDRESS , 6-12
 概要 , 6-12
TERM
 環境変数 , 2-12
Time Series Cartridge , 7-2
 dev2k デモ , 7-2
 extend デモ , 7-2
 oci デモ , 7-2
 proc デモ , 7-2
 usage デモ , 7-2

Time Series Cartridge のデモ
 インストール , 7-2
TMPDIR , 2-12
TNS_ADMIN
 環境変数 , 2-10
TNS リスナー
 Oracle TCP/IP Protocol Adapter 用の構成 , 6-12
Toolkit II リソース・ファイル , 2-9
tstshm コマンド , 3-16
TWO_TASK
 環境変数 , 2-10
TZ
 および SYSDATE , 2-13
 環境変数 , 2-13

U

ufs ファイル・システム・タイプ , 3-28
UNIX
 セキュリティ , 2-21
UNIX カーネル
 チューニング , 3-37
UNIX コマンド
 SQL*Plus からの実行 , 4-7
UNIX ファイル・システム・タイプ , 3-28
User Profile
 SQL*Plus , 4-2
utltdstat コマンド , 3-14
utlstat コマンド , 3-14

V

V\$SESSION_WAIT 表 , 3-32
V\$SGASTAT 表 , 3-19
V\$SYSTEM_EVENT 表 , 3-31
Veritas ファイル・システム・タイプ , 3-28
Visual Information Retrieval Cartridge , 7-3
 デモの作成 , 7-3
vmstat , 3-6
vmstat コマンド , 3-15
vxfv ファイル・システム・タイプ , 3-28

X

XA 機能 , 5-21
XENVIRONMENT
 環境変数 , 2-12

X-Window , 2-11 , 2-12

あ

アクセス

チューニング , 3-14

アダプタ・ユーティリティ , 6-4

い

インポート

チューニング , 3-31

う

受取りルーチン

例 , 5-19

え

エクステントの断片化 , 3-27

エクスポート

チューニング , 3-31

エディタ

SQL*Plus , 4-6

順序の設定 , 4-6

お

オペレーティング・システム・コマンド

SQL*Plus からの実行 , 4-7

か

カーソル

セッション・キャッシュのチューニング , 3-21

カートリッジのデモ

Image Cartridge , 7-4

Time Series 用のインストール , 7-2

Visual Information Retrieval Cartridge , 7-3

カーネル

UNIX パラメータのチューニング , 3-37

チューニング , 3-16

各国語サポート (NLS)

環境変数 , 2-8

環境変数 , 5-8

DISPLAY , 2-11

HOME , 2-11

LANG , 2-11

LANGUAGE , 2-11

LD_LIBRARY_PATH , 2-11

LDOPTS , 2-11

LDPATH , 2-11

LPDEST , 2-11

MicroFocus COBOL コンパイラ , 5-8

NLS_LANG , 2-8

ORA_NLS , 2-8

ORACLE_HELP , 2-8

ORACLE_HOME , 2-8

ORACLE_SID , 2-9

ORACLE_TERM , 2-9

ORACLE_TERMINAL , 2-9

ORACLE_TRACE , 2-9

ORAENV_ASK , 2-9

PATH , 2-12

PRINTER , 2-12

Pro*COBOL , 5-7 , 5-8

SHELL , 2-12

SQL*Plus の設定 , 4-7

TERM , 2-12

TMPDIR , 2-12

TNS_ADMIN , 2-10 , 6-3

TWO_TASK , 2-10

TZ , 2-13

XENVIRONMENT , 2-12

感嘆符

SQL*Plus プロンプト , 4-7

管理

SQL , 4-2

き

キーワード

DBA グループ ID , 2-25

疑問符

使用例 , 2-10

キャッシュ

SGA のチューニング , 3-17

サイズ

チューニング , 3-42

ヒット率 , 3-43

共有プールのサイズ

チューニング , 3-19

共通の環境

- oraenv ファイル, 2-5
- 設定, 2-5

共有

- オブジェクト・ライブラリ, 2-11
- ライブラリ・ローダー, 2-11

共有メモリー

- SGA, 2-17
- SGA のチューニング, 3-16
- 見積り, 3-16

共有オブジェクト・ライブラリ, 2-11

緊密共有メモリー, 2-19

く

クラスタ

- サイズの見積り, 2-14

グループ

- サンプル・スクリプト, 2-30

け

権限

- dba グループ, 2-21
- 付与, 2-22

言語, 2-8

こ

構成ファイル

- Net8, 6-3
- プリコンパイラ, 5-3

コマンド

- expst, 3-31
- impst, 3-31
- ipcs, 3-16
- orapwd, 2-26
- pstat, 3-15
- sar, 3-15
- swap, 3-15
- tstshm, 3-16
- utltdbstat, 3-14
- utlstat, 3-14
- vmstat, 3-15

コマンド・インタプリタ, 2-12

さ

索引

- チューニング, 3-22, 3-42

索引サイズ

- 計算, 2-14

サブシェル

- SQL*Plus での作成, 4-7

参照ドキュメント, xi

し

シグナル

- 2 タスク, 5-18
- ハンドラの作成, 5-18

シグナル・ハンドラ

- シグナル, 5-18

- 使用, 5-18

シグナル・ルーチン

- 例, 5-19

システム構成ファイル

- Pro*C/C++, 5-4
- Pro*COBOL, 5-7

システム時刻

- 設定, 2-13

システム・エディタ

- SQL*Plus, 4-6
- 順序の設定, 4-6
- デフォルト設定, 4-7

システム・グローバル領域 (SGA)

- 再配置, 2-19
- 要件, 2-17

実行プログラム, 2-12

自動ログイン

- listener.ora ファイル, 2-24
- remote_os_roles, 2-28
- UNIX 以外のシステム, 2-24

シャドウ・プロセス

- セキュリティ, 2-21

初期化パラメータ

- BACKGROUND_DUMP_DEST, 2-16
- BITMAP_MERGE_AREA_SIZE, 2-16
- COMMIT_POINT_STRENGTH, 2-16
- CONTROL_FILES, 2-16
- CREATE_BITMAP_AREA_SIZE, 2-16
- DB_BLOCK_BUFFERS, 2-16
- DB_BLOCK_SIZE, 2-16

DB_FILE_DIRECT_IO_COUNT , 2-16
DB_FILE_MULTIBLOCK_READ_COUNT , 2-16
DB_FILES , 2-16
DISTRIBUTED_TRANSACTIONS , 2-16
HASH_AREA_SIZE , 2-16
HASH_MULTIBLOCK_IO_COUNT , 2-16
LOCK_SGA , 2-16
LOCK_SGA_AREAS , 2-16
LOG_ARCHIVE_BUFFER_SIZE , 2-16
LOG_ARCHIVE_BUFFERS , 2-16
LOG_ARCHIVE_DEST , 2-16
LOG_ARCHIVE_FORMAT , 2-16
LOG_BUFFER , 2-16
LOG_CHECKPOINT_INTERVAL , 2-16
LOG_SMALL_ENTRY_MAX_SIZE , 2-16
MTS_LISTENER_ADDRESS , 2-17
MTS_MAX_DISPATCHERS , 2-16
MTS_MAX_SERVERS , 2-16
MTS_SERVERS , 2-17
NLS_LANGUAGE , 2-17
NLS_TERRITORY , 2-17
OBJECT_CACHE_MAX_SIZE_PERCENT , 2-17
OBJECT_CACHE_OPTIMAL_SIZE , 2-17
OPEN_CURSORS , 2-17
OS_AUTHENT_PREFIX , 2-17
PROCESSES , 2-17
REMOTE_OS_AUTHENT , 2-24
SHARED_POOL_SIZE , 2-17
SHOW PARAMETERS コマンド , 2-15
SORT_AREA_SIZE , 2-17
SORT_READ_FAC , 2-17
SORT_SPACEMAP_SIZE , 2-17
USER_DUMP_DEST , 2-17
デフォルト , 2-15

す

スーパーユーザー , 2-20
スピン・カウント
チューニング , 3-36
スワップ領域
チューニング , 3-14 , 3-15

せ

制限
リソース , 2-15

制限事項 (SQL*Plus) , 4-8
COPY コマンド , 4-8
ウィンドウのサイズ変更 , 4-9
リターン・コード , 4-9
静的リンクおよび動的リンク
Oracle ライブラリ , 5-17
セキュリティ
2 タスク・アーキテクチャ , 2-21
CONNECT INTERNAL , 2-23
Server Manager アクセス , 2-22
SHUTDOWN コマンド , 2-23
STARTUP コマンド , 2-23
グループ・アカウント , 2-21
権限の割当て , 2-21
デフォルト・グループ名 , 2-22
ネットワーク , 2-24
ファイル所有権 , 2-21
セッション・キャッシュ・カーソル
チューニング , 3-21
設定
RAW デバイス , 2-38

た

タイム・ゾーン
TZ での設定 , 2-13
断片化
チューニング , 3-27

ち

チューニング
I/O ボトルネック , 3-22
Oracle リソース , 3-31
Parallel Server , 3-38
RAW パーティション , 3-28
アプリケーションの設計 , 3-13
カーネル・パラメータ , 3-37
競合の分割 , 3-31
クライアント / サーバー構成 , 3-31
索引の競合の回避 , 3-42
使用状況のパターン , 3-30
スピン・カウント , 3-36
断片化の低減 , 3-27
ディスク I/O のローカライズ , 3-38
トレース・ファイルおよびアラート・ファイル ,
2-36

表のストライプ化, 3-30
ホット・ファイルの分散, 3-26
マルチ・プロセッサ・システム, 3-36

つ

ツール

iostat, 3-9
mpstat, 3-12
sar, 3-7
swap, 3-10
vmstat, 3-6

て

ディスク

パフォーマンスの監視, 3-22
割当て制限, 2-15

ディスク I/O

Parallel Server のチューニング, 3-38
チューニング, 3-22
非同期 I/O, 3-25
要求キュー, 3-26

データベース

管理者
実行プログラムへの権限, 2-21
バッファのチューニング, 3-17
ファイル
セキュリティ, 2-23
認可, 2-23

データベース I/O

DBWR チューニング, 3-24

データベースのバッファ

チューニング, 3-28

データベースの変更, 2-6

データベース・ライター

チューニング, 3-24

データ・カートリッジ

デモの実行, 7-1

データ・ディクショナリ・キャッシュ

チューニング, 3-19

デバック・プログラム, 5-3

デフォルト

ディレクトリ, 2-11, 2-12
プリンタ, 2-11
リンカー・オプション, 2-11

デフォルト設定

システム・エディタ, 4-7

デフォルト・プリンタ, 2-12

デモ

Image Cartridge 用の作成, 7-4

Time Series Cartridge 用のインストール, 7-2

Visual Information Retrieval Cartridge 用の作成,
7-3

デモンストレーション

プリコンパイラ, 2-31

プロシージャ・オプション、PL/SQL, 2-30

デモンストレーション表

SQL*Plus, 4-3

削除, 4-4

手動で作成, 4-4

デモンストレーション・プログラム

Oracle コール・インタフェース, 5-13

Pro*C/C++, 5-4

Pro*COBOL, 5-8

テンポラリ・ディスク・ファイル, 2-12

と

問合せ

パラレルのチューニング, 3-35

動的リンクおよび静的リンク

Oracle ライブラリ, 5-17

特殊アカウント, 2-20

特殊グループ

root, 2-21

トレース・ファイルおよびアラート・ファイル

アラート・ファイル, 2-36

使用, 2-36

トレース・ファイル名, 2-36

に

入出力, 2-11

ね

ネットワーク

DBA 権限, 2-24

セキュリティ, 2-23

パスワード, 2-23

は

- 配布ソフトウェア, 2-8
- バインディング、プロセス, 3-30
- パスワード
 - リモート, 2-26
- パスワード・ファイル
 - 共有, 2-27
- バッファ
 - チューニング, 3-17
 - データベース, 3-28
- バッファ・キャッシュ・サイズ
 - チューニング, 3-42
- バッファ・マネージャ, 3-14
- パフォーマンス
 - Parallel Server, 3-38
 - 表のストライプ化, 3-30
- パラレル問合せ
 - チューニング, 3-35

ひ

- ヒット率
 - バッファ・キャッシュ, 3-17
- 非同期 I/O, 3-25
 - 使用, 3-25
- 表
 - ストライプ化, 3-30
 - データを別々のファイルに格納, 3-30
- 表のストライプ化, 3-24
 - チューニング, 3-26
- 表領域
 - 断片化, 3-27

ふ

- ファイル
 - トレース・ファイル, 2-36
- ファイルの設定
 - SQL*Plus, 4-2
- ファイル名
 - SQL*Plus でのデフォルト拡張子, 4-8
- ファイル・システム・タイプ, 3-28
- プールのサイズ
 - チューニング, 3-19
- 複数のシグナル・ハンドラ, 5-19

複数のデータベース

- 共有パスワード・ファイル, 2-27
- プリコンバイラ
 - ireclen および oreclen の値, 5-3
 - 値, 5-3
 - 大文字から小文字への変換, 5-3
 - 概要, 5-2
 - シグナル, 5-19
 - 実行ファイルの再リンク, 5-2
 - デモンストレーションの実行, 2-31
 - ベンダー提供のデバッグ・プログラム, 5-3
- プリコンバイラ構成ファイル, 5-3
- プリコンバイラ実行ファイルの再リンク, 5-2
- プロセッサのバインディング, 3-30
- プロトコル・アダプタ, 6-6
 - ADDRESS の指定, 6-7

へ

- ページング領域
 - チューニング, 3-14, 3-15
- ヘルプ機能
 - SQL*Plus, 4-4
- ヘルプ・ファイル, 2-8

ほ

- ホーム・ディレクトリ, 2-11

ま

- マルチスレッド・サーバー, 6-4

め

- メモリー
 - SGA のチューニング, 3-16
 - 仮想, 2-13
 - 共有, 2-17
 - 共有の見積り, 3-16
 - 使用量の見積り, 2-13
 - チューニング, 3-14

ゆ

- ユーザー, 5-10
 - サンプル・スクリプト, 2-30

ユーザー・プログラム

Oracle コール・インタフェース, 5-15

Pro*C/C++, 5-6

Pro*COBOL, 5-10

ユーザー割込みハンドラ, 5-19

優先順位、プロセスの, 3-30

ら

ライターの動作

チューニング, 3-24

ライブラリ・キャッシュ

チューニング, 3-20

ラッチ

競合, 3-34

スピン・カウンットのチューニング, 3-36

り

リソース定義, 2-12

リソースの制限, 2-15

率

キャッシュ・ヒット, 3-43

リモート

接続

INTERNAL として, 2-28

OPERATOR として, 2-28

パラメータ

REMOTE_OS_AUTHENT, 2-24

ログイン

セキュリティ, 2-25

リモート DBA アクセスの制御

REMOTE_DBA_OPS_ALLOWED, 2-25

REMOTE_DBA_OPS_DENIED, 2-25

リモート接続パラメータ

OS_AUTHENT_PREFIX, 2-28

REMOTE_OS_AUTHENT, 2-28

REMOTE_OS_ROLES, 2-28

リモート・ログインの制御, 2-25

OPS_DOLLAR_LOGIN_DENIED, 2-25

リンク

シングルタスク, 3-31

ろ

ロールバック・セグメント

競合, 3-34

ログイン・ホーム・ディレクトリ

管理, 2-29

サンプル・スクリプト, 2-29

ロック

Parallel Server のチューニング, 3-41

論理ボリューム

チューニング, 3-24

論理ボリューム・マネージャ, 3-24

れ

列

チューニング, 3-13

