

Oracle8 Server 管理者ガイド

リリース 8.0

1998 年 2 月

部品番号 A56816-1

Oracle8 Server 管理者ガイド

部品番号 A56816-1

第 1 版：1998 年 2 月

原本名：Oracle8 Administrator's Guide, Release 8.0

原本部品番号：A58397-01

原本著者：Joyce Fee

協力者：John Bellemore, Atif Chaudhry, Sandra Cheevers, Connie Dialeris, John Frazzini, Mike Hartstein, Bhaskar Himatsingka, Alex Ho, Wei Huang, Ken Jacobs, Robert Jenkins, Val Kane, Andre Kruglikov, Bill Lee, Nina Lewis, Phil Locke, Diana Lorentz, Ekrem Soylemez, Jags Srinivasan, Ashwini Surpur, Alex Tsukerman

グラフィック・デザイナー：Valarie Moore

Copyright© 1997, 1998, 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。

本書の情報は、予告なしに変更されることがあります。本書に問題を見つけたら、当社にコメントをお送りください。オラクル社は本書の無謬性を保証しません。

危険な用途への使用について

当社製品は、原子力、航空産業、大量輸送、又は医療の分野など、本質的に危険が伴うアプリケーションを用途として特に開発されておられません。当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いただきたく、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、当社および開発元である米国 Oracle Corporation（その関連会社も含みます。）は一切責任を負いかねます。

ORACLE は、Oracle Corporation の登録商標です。

本文中の他社の商品名は、それぞれ各社の商標または登録商標です。

目次

はじめに	xix
------------	-----

第 1 部 基本データベース管理

1 Oracle データベース管理者

Oracle ユーザーのタイプ	1-2
データベース管理者	1-2
セキュリティ管理者	1-3
アプリケーション開発者	1-3
アプリケーション管理者	1-3
データベース・ユーザー	1-3
ネットワーク管理者	1-3
データベース管理者のセキュリティと権限	1-4
データベース管理者のオペレーティング・システム・アカウント	1-4
データベース管理者のユーザー名	1-4
DBA ロール	1-5
データベース管理者の認証	1-5
認証方法の選択	1-6
オペレーティング・システム認証を使用する	1-7
OSOPER と OSDBA	1-7
認証パスワード・ファイルを使用する	1-8
パスワード・ファイル管理	1-8
ORAPWD を使用する	1-9
REMOTE_LOGIN_PASSWORDFILE を設定する	1-10
ユーザーをパスワード・ファイルに追加する	1-11

管理者権限で接続する	1-13
パスワード・ファイルのメンテナンス	1-14
データベース管理者ユーティリティ	1-16
Enterprise Manager	1-16
SQL*Loader	1-16
エクスポートとインポート	1-16
データベース管理者の初期優先順位	1-17
ステップ 1: Oracle ソフトウェアをインストールする	1-17
ステップ 2: データベース・サーバーのハードウェアを評価する	1-18
ステップ 3: データベースを計画する	1-18
ステップ 4: データベースを作成し、オープンする	1-19
ステップ 5: データベース設計をインプリメントする	1-19
ステップ 6: データベースをバックアップする	1-19
ステップ 7: システム・ユーザーを登録する	1-20
ステップ 8: データベース・パフォーマンスを調整する	1-20
Oracle ソフトウェア・リリースを識別	1-20
リリース番号の形式	1-20
他の Oracle ソフトウェアのバージョン	1-21
現行のリリース番号を調べる	1-22

2 Oracle データベースの作成

データベースを作成する前の考慮点	2-2
作成の前提条件	2-3
初期データベースを使用する	2-3
旧リリースのデータベースを移行する	2-3
Oracle データベースの作成	2-4
Oracle データベースを作成する手順	2-4
データベースの作成例	2-7
データベース作成の問題を解決する	2-8
データベースを削除する	2-8
パラメータ	2-8
DB_NAME および DB_DOMAIN	2-9
CONTROL_FILES	2-9
DB_BLOCK_SIZE	2-10
DB_BLOCK_BUFFERS	2-11

PROCESSES	2-11
ROLLBACK_SEGMENTS	2-11
ライセンスに関するパラメータ	2-12
LICENSE_MAX_SESSIONS_ と LICENSE_SESSIONS WARNING	2-12
LICENSE_MAX_USERS	2-13
データベースを作成した後の考慮点	2-13
初期のチューニング・ガイドライン	2-14
ロールバック・セグメントを割り当てる	2-14
DB_BLOCK_LRU_LATCHES の数値を選択する	2-14
I/O を分散させる	2-15

3 起動と停止

起動手順	3-2
インスタンス起動の準備	3-2
インスタンス起動の方法	3-3
データベースの可用性の変更	3-6
インスタンスにデータベースをマウントする方法	3-6
クローズされたデータベースをオープンする方法	3-7
オープン状態のデータベースへのアクセスを制限する方法	3-7
停止手順	3-8
通常の条件下でデータベースを停止する方法	3-10
データベースを即時停止する方法	3-11
シャットダウン・トランザクション	3-11
インスタンスを中断する方法	3-12
パラメータ・ファイルの使用	3-12
サンプル・パラメータ・ファイル	3-13
パラメータ・ファイルの数	3-13
分散環境におけるパラメータ・ファイルの位置	3-13

第 2 部 Oracle Server の構成

4 Oracle プロセスの管理

専用サーバー・プロセス用に Oracle を構成	4-2
専用サーバー・プロセスに接続する場合	4-3
マルチスレッド・サーバー・プロセス用に Oracle を構成	4-3

SHARED_POOL_SIZE: 共有サーバーの共有プールに追加領域を割り当てる	4-5
MTS_LISTENER_ADDRESS: リスナー・プロセスのアドレスを設定する	4-5
MTS_SERVICE: ディスパッチャに対してサービス名を指定する	4-6
MTS_DISPATCHERS: ディスパッチャの最初の数を設定する	4-6
MTS_MAX_DISPATCHERS: ディスパッチャの最大数を設定する	4-8
MTS_SERVERS: 共有サーバー・プロセスの初期数を設定する	4-8
MTS_MAX_SERVERS: 共有サーバー・プロセスの最大数を設定する	4-9
サーバー・プロセスの変更	4-9
共有サーバー・プロセスの最小数を変更する	4-9
ディスパッチャ・プロセスを追加、削除する	4-9
Oracle プロセスを追跡	4-10
Oracle インスタンスのプロセスを監視する	4-10
トレース・ファイル、ALERT ファイル、バックグラウンド・プロセス	4-13
チェックポイント・プロセスを起動する	4-15
並列問合せオプションのプロセスの管理	4-16
問合せサーバーを管理する	4-16
問合せサーバー・プロセスの数の変化	4-16
外部プロシージャのプロセスの管理	4-17
セッションの停止	4-18
停止するセッションを識別する	4-19
アクティブなセッションを停止する	4-19
アクティブでないセッションを停止する	4-20

5 オンライン REDO ログの管理

オンライン REDO ログの計画	5-2
オンライン REDO ログを多重化する	5-2
オンライン REDO ログ・メンバーを別々のディスクに配置する	5-3
オンライン REDO ログ・メンバーのサイズを設定する	5-3
適切な数のオンライン REDO ログ・ファイルを選択する	5-4
オンライン REDO ログ・グループおよびメンバーの作成	5-5
オンライン REDO ログ・グループを作る	5-5
オンライン REDO ログ・メンバーを作る	5-6
オンライン REDO ログ・メンバーを改名および再配置	5-6
オンライン REDO ログ・グループの削除	5-8
オンライン REDO ログ・メンバーの削除	5-9

チェックポイントとログ・スイッチの制御	5-10
データベースのチェックポイント間隔を設定する	5-10
ログ・スイッチを強制する	5-12
ログ・スイッチなしで高速データベース・チェックポイントを強制する	5-12
REDO ログ・ファイル内のブロックの検証	5-13
オンライン REDO ログ・ファイルの消去	5-13
制限	5-14
オンライン REDO ログ・ファイルに関する情報のリスト	5-14

6 制御ファイルの管理

制御ファイルのガイドライン	6-2
制御ファイルを命名する.....	6-2
制御ファイルを異なるディスク上に多重化する.....	6-2
制御ファイルを正しく配置する.....	6-3
制御ファイルのサイズを管理する.....	6-3
制御ファイルの作成	6-3
初期制御ファイルを作成する.....	6-4
制御ファイルの追加コピーの作成と制御ファイルの改名 / 再配置.....	6-4
新しい制御ファイル.....	6-5
新しい制御ファイルを作成する	6-6
制御ファイルの作成後の問題解決	6-7
欠落したまたは余分なファイルをチェックする.....	6-7
CREATE CONTROLFILE でのエラーを処理する	6-8
制御ファイルの削除	6-8

7 ジョブ・キューの管理

SNP バックグラウンド・プロセス	7-2
複数の SNP プロセス	7-3
SNP プロセスを起動する	7-3
ジョブ・キューの管理	7-4
DBMS_JOB パッケージ	7-4
ジョブをジョブ・キューに送る	7-5
ジョブの実行方法.....	7-9
ジョブ・キューからジョブを削除する.....	7-11
ジョブを変更する	7-11

中断されたジョブ	7-12
ジョブを強制的に実行する	7-13
ジョブを終了する	7-15
ジョブ・キューについての情報の表示	7-15

第3部 Oracle Server の構成

8 表領域の管理

表領域を管理するためのガイドライン	8-2
複数の表領域を使用する	8-2
表領域の記憶領域パラメータを使用する	8-2
ユーザーに表領域割当て制限を割り当てる	8-3
表領域の作成	8-3
一時表領域を作成する	8-4
表領域割当ての管理	8-5
表領域に対する記憶領域設定を変更する	8-5
空き領域を合わせる	8-6
表領域の可用性の変更	8-7
表領域をオンラインにする	8-7
表領域をオフラインにする	8-8
表領域を読み専用にする方法	8-10
前提条件	8-10
読み専用表領域を書込み可能にする	8-11
WORM デバイスに読み専用表領域を作成する	8-11
表領域の削除	8-12
表領域についての情報の表示	8-13

9 データ・ファイルの管理

データ・ファイルを管理するためのガイドライン	9-2
データ・ファイルの数	9-2
データ・ファイルのサイズ設定	9-4
適切なデータ・ファイルの配置	9-4
REDO ログ・ファイルと分離したデータ・ファイルの格納	9-4
データ・ファイルの作成、表領域への追加	9-4
データ・ファイルのサイズを変更	9-5

データ・ファイルの自動拡張機能を使用可能および使用禁止にする	9-5
手動でデータ・ファイルのサイズを変更する	9-6
データ・ファイルの可用性の変更	9-7
ARCHIVELOG モードでデータ・ファイルをオンラインにする	9-8
NOARCHIVELOG モードでデータ・ファイルをオフラインにする	9-8
データ・ファイルの改名、再配置	9-8
単一の表領域のデータ・ファイルを改名、再配置する	9-9
複数の表領域のデータ・ファイルを改名、再配置する	9-10
データ・ファイル内のデータ・ブロックの検証	9-12
データ・ファイルの情報の表示	9-12

10 スキーマ・オブジェクトを管理するためのガイドライン

データ・ブロックの領域管理	10-2
PCTFREE パラメータ	10-2
PCTUSED パラメータ	10-4
関連する PCTUSED と PCTFREE の値を選択する	10-6
記憶領域パラメータの設定	10-7
指定可能な記憶領域パラメータ	10-7
INITRANS と MAXTRANS を設定する	10-9
表領域内のセグメントのデフォルト記憶領域パラメータを設定する	10-9
データ・セグメントの記憶領域パラメータを設定する	10-10
索引セグメントの記憶領域パラメータを設定する	10-10
LOB セグメントの記憶領域パラメータを設定する	10-10
記憶領域パラメータの値を変更する	10-10
記憶領域パラメータの優先順位を理解する	10-11
領域の割当ての解除	10-12
高水位標を表示する	10-13
領域割当て解除文を発行する	10-13
データ型の使用領域の理解	10-16
Oracle のデータ型のまとめ	10-18

11 パーティション表と索引の管理

パーティション表と索引とは何か	11-2
パーティションの作成	11-2
パーティションのメンテナンス	11-3

パーティションを移動する	11-4
パーティションを追加する	11-4
パーティションを削除する	11-5
パーティションを切り捨てる	11-7
パーティションを分割する	11-9
パーティションをマージする	11-10
表のパーティションの交換	11-11
索引のパーティションの再構築	11-14
履歴表での時間枠の移動	11-14
複数ステップのメンテナンス操作中のアプリケーション静止	11-15

12 表の管理

表を管理するためのガイドライン	12-2
作成前の表の設計	12-2
データ・ブロック領域の使用方法を指定する	12-3
トランザクション・エントリ・パラメータを指定する	12-3
各表の位置を指定する	12-3
表作成を並行化する	12-4
回復不能表作成に関する考慮事項	12-4
表のサイズの見積もりと記憶領域パラメータの設定	12-5
大規模な表の計画	12-5
表の制限	12-6
表の作成	12-6
表の変更	12-7
表の記憶領域を手動で割り当て	12-8
表の削除	12-8
索引構成表	12-9
索引構成表とは何か	12-9
索引構成表を作成する	12-11
索引構成表をメンテナンスする	12-14
シナリオ : 索引構成表で ORDER BY 句を使用する場合	12-14
シナリオ : キー列を更新する	12-15
索引構成表を標準的な表に変換する	12-15

13 ビュー、順序、シノニムの管理

ビューの管理	13-2
ビューを作成する.....	13-2
結合ビューを変更する.....	13-4
ビューを置き換える.....	13-8
ビューを削除する.....	13-9
順序の管理	13-9
順序を作成する.....	13-9
順序を変更する.....	13-10
順序に影響を及ぼす初期化パラメータ.....	13-10
順序を削除する.....	13-10
シノニムの管理	13-11
シノニムを作成する.....	13-11
シノニムを削除する.....	13-11

14 索引の管理

索引を管理するためのガイドライン	14-2
表データ挿入後に索引を作成する.....	14-2
表あたりの索引の数を制限する.....	14-3
トランザクション・エントリ・パラメータを指定する.....	14-3
索引ブロックの領域使用を指定する.....	14-3
各索引の表領域を指定する.....	14-4
索引作成を並行化する.....	14-4
回復不能索引作成に関する考慮事項.....	14-4
索引サイズの見積もりと記憶領域パラメータの設定.....	14-5
制約を使用禁止または削除する前の検討.....	14-6
索引の作成	14-6
制約に対応付けられる索引を作成する.....	14-6
索引を明示的に作成する.....	14-7
既存の索引を作成しなおす.....	14-7
索引の変更	14-8
索引の領域使用を監視	14-8
索引の削除	14-9

15 クラスタの管理

クラスタを管理するためのガイドライン	15-2
適切な表のクラスタ化	15-4
クラスタ・キーに対する適切な列の選択	15-4
データ・ブロック領域使用の指定	15-5
平均クラスタ・キーとその対応行が必要とする領域の指定	15-5
各クラスタとクラスタ索引の位置の指定	15-5
クラスタ・サイズの見積もりおよび記憶領域パラメータの設定	15-6
クラスタの作成	15-6
クラスタ化された表を作成する	15-7
クラスタ索引を作成する	15-7
クラスタの変更	15-8
クラスタ化した表とクラスタ索引を変更する	15-9
クラスタの削除	15-9
クラスタ化した表を削除する	15-10
クラスタ索引を削除する	15-10

16 ハッシュ・クラスタの管理

ハッシュ・クラスタを管理するためのガイドライン	16-2
ハッシングの長所	16-2
ハッシングの短所	16-3
ハッシュ・クラスタが必要とするサイズを見積もり、記憶領域パラメータを設定する	16-3
ハッシュ・クラスタを作成する	16-4
ハッシュ・クラスタ内の領域使用を制御する	16-5
ハッシュ・クラスタの変更	16-8
ハッシュ・クラスタの削除	16-8

17 スキーマ・オブジェクトの一般的な管理

一度の操作で複数の表やビューを作成	17-2
オブジェクトの改名	17-2
表、索引、クラスタの分析	17-3
表、索引、クラスタの統計を使用する	17-4
表、索引、クラスタの妥当性を検査する	17-8
表とクラスタの連鎖された行をリストする	17-8
表とクラスタの削除	17-9

トリガーの使用可能、使用禁止	17-11
トリガーを使用可能にする.....	17-11
トリガーを使用禁止にする.....	17-11
整合性制約の管理	17-12
整合性制約の状態.....	17-13
制約チェックを延期する.....	17-15
対応付けられた索引をもつ制約を管理する.....	17-17
定義時に整合性制約を使用禁止、または妥当性検査なしで使用可能、使用可能にする.....	17-17
既存の整合性制約を使用可能、使用禁止にする.....	17-18
整合性制約を削除する.....	17-20
制約例外をレポートする.....	17-20
オブジェクトの依存性の管理	17-22
ビューを手動で再コンパイルする.....	17-23
プロシージャとファンクションを手動で再コンパイルする.....	17-23
パッケージを手動で再コンパイルする.....	17-24
オブジェクト名の名前の変換の管理	17-24
データ・ディクショナリの記憶領域パラメータの変更	17-25
データ・ディクショナリの構造.....	17-25
データ・ディクショナリの記憶領域の変更が必要なエラー.....	17-27
スキーマ・オブジェクトについての情報を表示	17-28
ディクショナリの記憶領域の Oracle パッケージ.....	17-29
例 1: スキーマ・オブジェクトをタイプ別に表示する.....	17-29
例 2: 例情報を表示する.....	17-30
例 3: ビューとシノニムの依存性を表示する.....	17-30
例 4: 一般的なセグメント情報を表示する.....	17-31
例 5: 一般的なエクステント情報を表示する.....	17-31
例 6: データベースの空き領域（エクステント）を表示する.....	17-31
例 7: 追加のエクステントを割り当てることのできないセグメントを表示する.....	17-32

第 4 部 データベース・セキュリティ

18 ロールバック・セグメントの管理

ロールバック・セグメントを管理するためのガイドライン	18-2
複数ロールバック・セグメントを使用する.....	18-2
パブリックとプライベートのロールバック・セグメントの選択.....	18-3

自動的に獲得するロールバック・セグメントの指定.....	18-3
ロールバック・セグメント・サイズの設定.....	18-4
等しいサイズのエクステントを数多く持つロールバック・セグメントを作る.....	18-5
各ロールバック・セグメントに対するエクステントの最適数の設定.....	18-6
ロールバック・セグメントに対する記憶位置の設定.....	18-7
ロールバック・セグメントの作成	18-8
新しいロールバック・セグメントをオンラインにする.....	18-8
ロールバック・セグメントの記憶領域パラメータを指定	18-8
ロールバック・セグメントを作成するときに記憶領域パラメータを設定する.....	18-8
ロールバック・セグメントの記憶領域パラメータを変更する.....	18-9
ロールバック・セグメントのフォーマットを変更する.....	18-10
ロールバック・セグメントを手動で縮小する.....	18-10
ロールバック・セグメントのオンライン、オフライン	18-10
ロールバック・セグメントをオンラインにする.....	18-11
ロールバック・セグメントをオフラインにする.....	18-12
ロールバック・セグメントにトランザクションを明示的に割り当て	18-13
ロールバック・セグメントの削除	18-14
ロールバック・セグメント情報を監視	18-15
ロールバック・セグメント情報を表示する.....	18-15

19 セキュリティ方針の設定

システム・セキュリティ方針	19-2
データベース・ユーザー管理.....	19-2
ユーザー認証.....	19-2
オペレーティング・システムのセキュリティ.....	19-3
データ・セキュリティ方針	19-3
ユーザー・セキュリティ方針	19-4
一般的なユーザー・セキュリティ.....	19-4
エンド・ユーザーのセキュリティ.....	19-5
管理者のセキュリティ.....	19-7
アプリケーションの開発者のセキュリティ.....	19-8
アプリケーション管理者のセキュリティ.....	19-10
パスワード管理方針	19-11
アカウント・ロック.....	19-11
パスワード・エイジングおよび時間切れ.....	19-12

パスワード履歴.....	19-13
パスワードの複雑さの検証.....	19-13
監査方針	19-17

20 ユーザーとリソースの管理

セッションとユーザーのライセンス	20-2
同時使用ライセンス.....	20-2
接続権限.....	20-3
セッションの最大数を設定する.....	20-4
セッション警告制限を設定する.....	20-4
データベースの稼働中に同時使用制限を変更する.....	20-4
名前付きユーザー制限.....	20-5
ライセンス制限と現行値を参照する.....	20-6
ユーザー認証	20-7
データベース認証.....	20-7
外部認証.....	20-8
企業認証.....	20-10
Oracle ユーザー	20-10
ユーザーを作成する.....	20-10
ユーザーを変更する.....	20-14
ユーザーを削除する.....	20-16
プロファイルでリソースを管理	20-17
プロファイルを作成する.....	20-17
プロファイルを割り当てる.....	20-18
プロファイルを変更する.....	20-18
複合制限を使用する.....	20-19
プロファイルを削除する.....	20-20
リソース制限を使用可能、使用禁止にする.....	20-20
データベース・ユーザーとプロファイルに関する情報のリスト	20-21
ユーザーとプロファイルに関する情報を記述する例.....	20-22
例	20-25

21 ユーザー権限とロールの管理

ユーザー権限の識別	21-2
システム権限.....	21-2

オブジェクト権限.....	21-9
ユーザー・ロールの管理	21-11
ロールを作成する.....	21-11
事前定義済みのロール.....	21-12
ロールの認可.....	21-13
ロールを削除する.....	21-15
ユーザー権限とロールの付与	21-16
システム権限とロールを付与する.....	21-16
オブジェクト権限とロールを付与する.....	21-17
列に権限を付与する.....	21-18
ユーザー権限とロールの取り消し	21-18
システム権限とロールを取り消す.....	21-18
オブジェクト権限とロールを取り消す.....	21-19
権限の取消しによる影響.....	21-20
ユーザー・グループ PUBLIC に対する付与と取消し.....	21-21
オペレーティング・システムまたはネットワークを使ってロールを付与	21-22
オペレーティング・システムのロール識別機能を使用する.....	21-23
オペレーティング・システムのロール管理機能を使用する.....	21-24
OS_ROLES=TRUE の場合にロールを付与する、取り消す.....	21-24
OS_ROLES=TRUE の場合にロールを使用可能、使用禁止にする.....	21-25
オペレーティング・システムによるロール管理でネットワーク接続を使用する.....	21-25
権限とロールの情報を記述	21-25
権限とロールの情報を記述する例.....	21-26

22 データベース使用の監査

監査機能のガイドライン	22-2
データベースまたはオペレーティング・システムで監査する.....	22-2
監査済み情報を監理しやすい状態に維持する.....	22-2
データベース監査証跡ビューの作成、削除	22-4
監査証跡ビューを作成する.....	22-4
監査証跡ビューを削除する.....	22-5
監査証跡情報の管理	22-5
デフォルトで監査されるイベント.....	22-6
監査オプションを設定する.....	22-7
データベース監査を使用可能、使用禁止にする.....	22-13

監査証跡の成長とサイズを制御する	22-14
監査証跡を保護する	22-16
データベース監査証跡情報の参照	22-16
アクティブな文監査オプションを記述する	22-18
特定のオブジェクトのアクティブな権限監査オプションを記述する	22-18
特定のオブジェクトのアクティブなオブジェクト監査オプションをリストする	22-18
デフォルトのオブジェクト監査オプションを記述する	22-19
監査レコードを記述する	22-19
AUDIT SESSION オプションの監査レコードを記述する	22-19
データベース・トリガーによる監査	22-20

23 REDO 情報のアーカイブ

NOARCHIVELOG モードと ARCHIVELOG モードの選択	23-2
データベースを NOARCHIVELOG モードで稼働する	23-2
データベースを ARCHIVELOG モードで稼働する	23-2
アーカイブの切り替え	23-4
初期データベース・アーカイブ・モードを設定する	23-4
データベース・アーカイブ・モードを変更する	23-4
自動アーカイブを使用可能にする	23-6
自動アーカイブを使用禁止にする	23-7
手動アーカイブを実行する	23-8
アーカイブの調整	23-8
システム・パフォーマンスへの影響を最小にする	23-9
アーカイブを高速にする	23-9
アーカイブ状態を表示	23-10
アーカイブ済み REDO ログのファイル名フォーマットと宛先を指定	23-11

A スキーマ・オブジェクトの領域の見積もり

クラスタ化されてない表に必要な領域の見積もり	A-2
索引の領域の見積もり	A-5
クラスタで必要な領域の見積もり	A-10
ハッシュ・クラスタで必要となる領域の見積もり	A-15

索引

はじめに

このマニュアルは、Oracle データベース・システムの運用を管理するユーザーを対象として記述しています。このようなユーザーは、「データベース管理者 (DBA)」と呼ばれ、Oracle データベース・システムの円滑な運用を保証し、その使用状況を監視する責任を負うものとみなされます。データベース管理者の責務については、第 1 章で説明します。

注意： このマニュアルは、Oracle8 と Oracle8 Enterprise Edition 製品の特徴と機能を説明しています。Oracle8 と Oracle8 Enterprise Edition の基本機能は同じです。ただし、最新の機能のいくつかは、Enterprise Edition だけで使用可能であり、オプションのものもあります。たとえば、自動の表領域 Point-in-Time 回復 (Recovery Manager を使用して) を実行するには、Enterprise Edition が必要です。

Oracle8 と Oracle8 Enterprise Edition の相違点、使用可能な機能とオプションの詳細は、『Oracle8 と Oracle8 Enterprise Edition の解説』を参照してください。

対象読者

このマニュアルの読者は、リレーショナル・データベースの概念に精通しているものと想定しています。また、Oracle を稼働しているオペレーティング・システム環境にも精通している必要があります。

前提条件として、すべての読者は、『Oracle8 Server 概要』の第 1 章「Oracle Server の技術的な基礎」に記載されている内容を理解しておく必要があります。この章では、このマニュアルを通じて使われる概念と用語について幅広く説明しています。

インストールと移行に関する情報が必要な読者

管理者は、Oracle Server ソフトウェアのインストールや、既存の Oracle データベースの新しい形式への移行作業（たとえば、バージョン 7 データベースを Oracle8 形式に移行する作業）にかかわることがあります。このマニュアルでは、インストールやシステムの移行については説明しません。

主に必要な情報がインストールに関する情報である場合は、使っているオペレーティング・システム固有の Oracle のマニュアルを参照してください。

主に必要な情報がデータベースまたはアプリケーションの移行に関する情報である場合は、『Oracle8 Server 移行ガイド』を参照してください。

アプリケーションの設計に関する情報が必要な読者

このマニュアルには管理者だけでなく、Oracle に精通したユーザーと上級のデータベース・アプリケーション設計者にとっても有効な情報が記載されています。

ただし、データベース・アプリケーションの開発者は、『Oracle8 Server アプリケーション開発者ガイド』と、Oracle データベース・アプリケーションを開発するために使うツールまたは言語製品のマニュアルも参照してください。

このマニュアルの使用方法

このマニュアルのすべての読者は、『Oracle8 Server 概要』の第 1 章「Oracle Server の技術的な基礎」を必ず読んでいなければなりません。そこに記述された Oracle に関する概念と用語の概要は、このマニュアルの詳細な情報の基礎となるものです。『Oracle8 Server 概要』の他の章では、Oracle のアーキテクチャと各機能について説明し、各機能の動作について詳しく述べています。

このマニュアルの構成

このマニュアルは、次の部と章で構成されています。

第 1 部：基本データベース管理

- | | |
|-------------------------|--|
| 第 1 章「Oracle データベース管理者」 | ソフトウェアのインストール、データベースの計画などのデータベース管理者が実行する一般的な作業の概要を説明します。 |
| 第 2 章「Oracle データベースの作成」 | データベースの作成について、最も考慮すべき点について説明します。データベースを計画するときに参照してください。 |
| 第 3 章「起動と停止」 | データベースの起動、またはその可用性の変更、停止を行うときに参照してください。起動と停止に関連したパラメータ・ファイルについても説明しています。 |

第 2 部：Oracle Server の構成

- | | |
|-------------------------|--|
| 第 4 章「Oracle プロセスの管理」 | 専用サーバー・プロセスおよびマルチスレッド・サーバー・プロセスなどの様々な Oracle プロセスの識別に役立ちます。プロセスの構成および変更、追跡、管理を行うときに参照してください。 |
| 第 5 章「オンライン REDO ログの管理」 | オンライン REDO ログの管理のすべての作業（計画または作成、改名、削除、消去など）について説明します。 |
| 第 6 章「制御ファイルの管理」 | 制御ファイルの管理のすべての作業（命名および作成、トラブルシューティング、削除）について説明します。 |
| 第 7 章「ジョブ・キューの管理」 | ジョブ・キューを扱う前に参照してください。ジョブ・キューの送信および削除、変更、調整などすべての作業について説明しています。 |

第 3 部：データベース記憶領域

- | | |
|---------------|---|
| 第 8 章「表領域の管理」 | 表領域の管理についてガイドラインを示し、表領域の作成および管理、変更、削除の方法を説明します。 |
|---------------|---|

第 9 章「データ・ファイルの管理」	データ・ファイルの管理についてガイドラインを示し、データ・ファイルに関する情報の作成および変更、改名、表示の方法を説明します。
第 10 章「スキーマ・オブジェクトを管理するためのガイドライン」	記憶領域パラメータの設定、領域の割当て解除および管理などの一般的な作業について説明します。
第 11 章「パーティション表と索引の管理」	パーティション表（および索引）について、およびその作成方法と管理方法を説明します。
第 12 章「表の管理」	一般的な表の管理のガイドライン、および表の作成、変更、メンテナンス、削除についても説明します。
第 13 章「ビュー、順序、シノニムの管理」	ビュー、順序、シノニムの管理のすべての作業について説明します。
第 14 章「索引の管理」	作成および変更、監視、削除などの索引についての一般的なガイドラインを示します。
第 15 章「クラスタの管理」	クラスタの作成および変更、削除などについての一般的なガイドラインを示します。
第 16 章「ハッシュ・クラスタの管理」	ハッシュ・クラスタの変更または削除についての一般的なガイドラインを示します。
第 17 章「スキーマ・オブジェクトの一般的な管理」	スキーマの管理について、第 10 章よりも専門的な作業について説明しています。表の分析、および表とクラスタの切捨て、データベース・トリガー、整合性制約、オブジェクトの依存性について説明します。いくつか例も記載されています。

第 4 部：データベース・セキュリティ

第 18 章「ロールバック・セグメントの管理」	ロールバック・セグメントを管理するためのガイドラインを示します。
第 19 章「セキュリティ方針の設定」	パスワードの管理に関連した特定の作業だけでなく、システム、データとユーザーのセキュリティ方針などのデータベース・セキュリティのすべての作業について説明します。

第 20 章「ユーザーとリソースの管理」	セッションとユーザーのライセンス、ユーザー認証について説明し、ユーザーとリソースの管理に関連した作業の特定の例を提供します。
第 21 章「ユーザー権限とロール管理」	ユーザー権限とロールの管理についてすべての点についての情報があります。権限とロールの付与および取消しの方法について説明します。
第 22 章「データベース使用の監査」	監査情報の作成および管理、参照方法について説明します。
第 23 章「REDO 情報のアーカイブ」	アーカイブ・モードおよびアーカイブのチューニング、表示方法について説明します。

付録

付録 A「スキーマ・オブジェクトの領域の見積もり」	スキーマ・オブジェクトに必要な領域の見積もりに関して特定の複数の計算式があります。
---------------------------	---

このマニュアルで使用する表記規則

ここでは、このマニュアルで使う表記上の規則について説明します。

- 本文
- 構文図および表記法
- コードの例

本文

ここでは、このマニュアルのテキストで使用する表記規則について説明します。

大文字

アルファベットの大文字は、コマンドのキーワード、オブジェクト名、パラメータ、ファイル名、などを明示するために使われています。次に例を示します。

「プライベート・ロールバック・セグメントを作る場合、ロールバック・セグメントの名前は、パラメータ・ファイルの ROLLBACK_SEGMENT パラメータに指定しなければなりません。」

イタリック

イタリック表記は、マニュアル名の表示、または強調を示す場合に使います。

構文図および表記法

このマニュアルの構文図と表記法では、SQL コマンドの構文、関数、ヒント、その他の要素を示しています。以降では、それらに基づき、構文図と例の読み方および SQL 文の書き方について説明します。

キーワード

キーワードは、SQL 言語で特別な意味があるワードです。このマニュアルの構文図では、キーワードを大文字で示します。大文字または小文字のどちらでも使える場合以外は、構文図が示すとおり SQL 文にキーワードを入力する必要があります。たとえば、CREATE TABLE 文を開始するには、CREATE TABLE 構文図が示すように CREATE キーワードを使用する必要があります。

パラメータ

パラメータは、構文図でブレース・ホルダーの役割をします。パラメータは小文字で示されます。パラメータは、通常、データベース・オブジェクトの名前またはオラクルのデータ型名、式です。構文図にパラメータがある場合、SQL 文では、オブジェクトまたは式の部分に適切な語句を代入します。たとえば、CREATE TABLE 文を記述するには、構文図の *table* パラメータのかわりに、EMP のような作成する表の名前を使用します。(パラメータ名が本文内でイタリックで示されていることに注意してください)。

次のリストでは、このマニュアルの構文図にあるパラメータと文中でその部分に代入される値の例を示します。

パラメータ	説明	例
<i>table</i>	代入値は、パラメータに指定されたタイプのオブジェクト名でなければなりません。	emp
<i>'text'</i>	代入値は、引用符でくくられた文字リテラルでなければなりません。	'Employee Records'
<i>condition</i>	代入値は TRUE または FALSE を評価する条件でなければなりません。	ename > 'A'
<i>date</i> <i>d</i>	代入値は、日付定数または DATE データ型の式でなければなりません。	TO_DATE ('01-Jan-1996', DD-MON-YYYY')
<i>expr</i>	代入値は、どのデータ型の式でも使用できます。	sal + 1000
<i>integer</i>	代入値は整数でなければなりません。	72
<i>rowid</i>	代入値は、データ型 ROWID の式でなければなりません。	00000462.0001.0001
<i>subquery</i>	代入値は、他の SQL 文に含まれる SELECT 文でなければなりません。	SELECT ename FROM emp
<i>statement_name</i> <i>block_name</i>	代入値は、SQL 文または PL/SQL ブロックの識別子でなければなりません。	s1 b1

コードの例

SQL、SQL*Plus の各コマンドや文は、下記のようにクーリエ・フォントで示され、本文のパラグラフとは区別して表記されています。

```
INSERT INTO emp (empno, ename) VALUES (1000, 'JFEE');  
ALTER TABLESPACE users ADD DATAFILE 'users2.ora' SIZE 50K;
```

例文には、コンマや引用符などの句読点が含まれている場合があります。例文に示した句読点はすべて必須です。例文はすべて、セミコロンで終了します。使うアプリケーションにより、文を終了する際にセミコロンやその他の終了記号が必要である場合とそうでない場合があります。

例文では、大文字によって Oracle SQL におけるキーワードを明示します。ただし、文を発行するときはキーワードが大 / 小文字区別を行わないことに注意してください。

例文では、小文字によって例文のコンテキストにだけ適用されたワードを示します。たとえば、小文字のワードは表、列、ファイルの名前を示す場合があります。

Enterprise Manager インタフェースの例

このマニュアルでは、Oracle データベースの管理のための主要なユーティリティである Enterprise Manager のダイアログ・ボックスとメニューの例を示しています。Server Manager の文字モードの画面の図もあります。ただし、使っているシステムのユーザー・インタフェースによって実際の画面の外観が異なることがあります。

詳細は、『Oracle Enterprise Manager 管理者ガイド』を参照してください。

第 1 部

基本データベース管理

Oracle データベース管理者

この章では、Oracle Server を管理する人、すなわちデータベース管理者の責任について説明します。

トピックは次のとおりです。

- Oracle ユーザーのタイプ
- データベース管理者のセキュリティと権限
- データベース管理者の認証
- パスワード・ファイル管理
- データベース管理者ユーティリティ
- データベース管理者の初期優先順位
- Oracle ソフトウェア・リリースを識別

Oracle ユーザーのタイプ

ユーザーのタイプやその責任は、サイトによって異なることがあります。たとえば、大規模なデータベースの場合は、データベース管理者が果たす義務を何人かで分担することがあります。

このセクションのトピックは次のとおりです。

- データベース管理者
- セキュリティ管理者
- アプリケーション開発者
- アプリケーション管理者
- データベース・ユーザー
- ネットワーク管理者

データベース管理者

Oracle データベース・システムは、非常に大規模で多数のユーザーがいるため、システムの管理は何人かの担当者または複数の担当者グループで行う必要があります。このような管理上の責任者をデータベース管理者（DBA）と呼びます。どのデータベースでも、管理の義務を果たす担当者が少なくとも 1 名は必要です。

データベース管理者の担当する作業は、次のとおりです。

- Oracle Server とアプリケーション Tools をインストールおよびアップグレードする。
- データベース・システムにシステム記憶を割り当て、将来の記憶要件を計画する。
- アプリケーション開発者がアプリケーションを設計した後、初期データベースの記憶構造（表領域）を作る。
- アプリケーション開発者がアプリケーションを設計した後、初期オブジェクト（表、ビュー、索引）を作る。
- アプリケーション開発者から得た情報に基づいてデータベース構造を修正する。
- ユーザーを登録し、システム・セキュリティをメンテナンスする。
- Oracle のライセンス契約に従っていることを保証する。
- データベースに対するユーザー・アクセスを制御、監視する。
- データベースのパフォーマンスを監視、最適化する。
- データベース情報のバックアップおよびリカバリの計画を立てる。
- テープ上のアーカイブ済みデータをメンテナンスする。
- データベースをバックアップ、復元する。
- 技術サポートについてオラクル社に連絡する。

セキュリティ管理者

場合によっては、データベースに 1 名以上のセキュリティ管理者が必要です。セキュリティ管理者は、主にユーザーの登録、データベースに対するユーザー・アクセスの制御と監視、およびシステム・セキュリティのメンテナンスに関連します。したがって、サイトごとにセキュリティ管理者がいる場合、データベース管理者はこれらの業務に対する責任はありません。

アプリケーション開発者

アプリケーション開発者は、データベース・アプリケーションを設計し、インプリメントします。アプリケーション開発者の担当する作業は、次のとおりです。

- データベース・アプリケーションを設計、開発する。
- アプリケーションのためのデータベース構造を設計する。
- アプリケーションに対する記憶要件を見積もる。
- アプリケーションのためのデータベース構造の変更を指定する。
- データベース管理者にこれらの情報を伝える。
- 開発中にアプリケーションを調整する。
- 開発中にアプリケーションのセキュリティ手段を確立する。

アプリケーション管理者

Oracle では、サイトごとに 1 名以上のアプリケーション管理者が必要な場合があります。アプリケーション管理者は、特定のアプリケーションの管理上の要求に責任があります。

データベース・ユーザー

データベース・ユーザーは、アプリケーションまたはユーティリティを介してデータベースと対話します。一般ユーザーの責任で行われる作業は、次のとおりです。

- 許された範囲でデータを入力および修正、削除する。
- データの報告書を作る。

ネットワーク管理者

サイトによっては、1 名以上のネットワーク管理者が存在することがあります。ネットワーク管理者は、Net8 などの Oracle ネットワーキング製品の管理を担当することがあります。

関連項目：『Oracle8 Server 分散システム』の「ネットワーク管理」

データベース管理者のセキュリティと権限

Oracle で管理業務を遂行するには、データベースとデータベースの稼働するサーバーのオペレーティング・システムの両方で特別な権限が必要です。データベース管理者のアカウントへのアクセスは厳しく管理する必要があります。

このセクションのトピックは次のとおりです。

- データベース管理者のオペレーティング・システム・アカウント
- データベース管理者のユーザー名
- DBA ロール

データベース管理者のオペレーティング・システム・アカウント

データベースに対する管理上の多くの業務を果たすには、オペレーティング・システム・コマンドを実行できなければなりません。Oracle が稼働するオペレーティング・システムによって異なりますが、オペレーティング・システムにアクセスするには、オペレーティング・システム・アカウント (ID) が必要になります。その場合、そのオペレーティング・システム・アカウントに対しては、多くのデータベース・ユーザーが Oracle ソフトウェアのインストールを実行するなどの場合に比べて、より大きなオペレーティング・システム権限やアクセス権が必要になります。Oracle ファイルを自分のアカウント内に格納する必要はありませんが、それらに対するアクセス権は必要です。

また、Enterprise Manager では、オペレーティング・システム・アカウントまたは ID をなんらかの方法で識別して、オペレーティング・システムによって権限が与えられた Enterprise Manager コマンドが使えなければなりません。

関連項目：データベース管理者のアカウントを識別する方法は、オペレーティング・システムによって異なります。詳細は、オペレーティングシステム固有の Oracle マニュアルを参照してください。

データベース管理者のユーザー名

データベースとともに 2 つのユーザー・アカウントが自動的に作られ、DBA ロールを付与されます。このユーザー・アカウントは次の 2 つです。

- SYS (初期パスワード: CHANGE_ON_INSTALL)
- SYSTEM (初期パスワード: MANAGER)

これらの 2 つのユーザー名について、次に説明します。

注意： データ・ディクショナリ表への不当なアクセスを防ぐために、Oracle データベースを作った直後、SYS および SYSTEM のユーザーに対するパスワードを変更しなければなりません。

日常的な管理業務を遂行するときに使う管理者ユーザー名を追加で作ることもできます。

SYS

データベースの作成時に、ユーザー SYS (パスワード CHANGE_ON_INSTALL で識別される) が自動的に登録され、DBA ロールが付与されます。

データベースのデータ・ディクショナリの実表とビューはすべて、スキーマ SYS に格納されます。この実表とビューは、Oracle の操作に非常に重要なものです。データ・ディクショナリの整合性を維持するため、SYS スキーマ内の表は Oracle によってしか操作されません。つまりユーザーやデータベース管理者は絶対に修正してはならず、ユーザー SYS のスキーマ内に表を作ってはなりません。(ただし、必要に応じてデータ・ディクショナリ設定の記憶領域パラメータを変更することはできます)。

ほとんどのデータベース・ユーザーに対し、SYS アカウントを使って接続できないようにする必要があります。管理者はこのアカウントを使ってデータベースに接続できますが、その場合も Oracle の技術サポートやマニュアルによって指示されたときだけに限定してください。

SYSTEM

データベースの作成時には、ユーザー SYSTEM (パスワード MANAGER によって識別される) も自動的に作成され、データベースに関するシステム権限がすべて付与されます。

ユーザー名 SYSTEM により、管理情報を表示する追加表およびビュー、さらに Oracle Tools が使用する内部表およびビューが作られます。個々のユーザーに関係する表は、SYSTEM スキーマ内に作ってはなりません。

DBA ロール

事前定義済みの "DBA" という名前のロールは、あらゆる Oracle データベースで自動的に作られます。このロールには、すべてのデータベース・システム権限が含まれています。これは非常に強力な権限であり、専任のデータベース管理者だけに付与してください。

データベース管理者の認証

データベース管理者は、データベースの起動や停止などの特殊な操作を実行する必要があります。このような操作は、通常のデータベース・ユーザーは実行すべきではないため、データベース管理者のユーザー名には安全性の高い認証方式が必要です。

このセクションのトピックは次のとおりです。

- 認証方法の選択
- オペレーティング・システム認証を使用する
- OSOPER と OSDBA
- 認証パスワード・ファイルを使用する

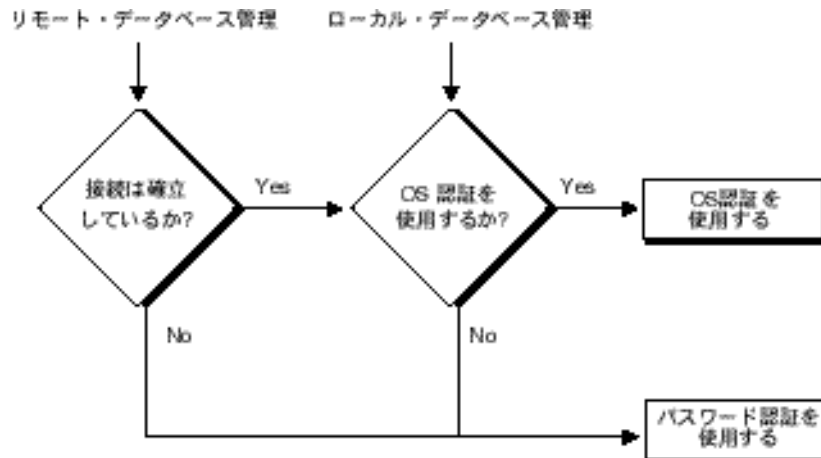
認証方法の選択

次に示すデータベース管理者の認証方法は、Oracle の旧バージョンで提供された CONNECT INTERNAL 構文にかわるものです（CONNECT INTERNAL は、引き続きサポートされますが、それは旧バージョンとの互換性を保つためです）。

- オペレーティング・システム認証
- パスワード・ファイル

データベースが存在する同一のマシン上でデータベースをローカルで管理するか、または単一のリモート・クライアントから複数の異なるデータベースを管理するかによって、オペレーティング・システム認証か、またはパスワード・ファイルのどちらかを選択できます。図 1-1 は、データベース管理者の認証方式の選択肢を示したものです。

図 1-1 データベース管理者の認証方式



ほとんどのオペレーティング・システムでは、データベース管理者の OS 認証のため、データベース管理者の OS ユーザー名を特殊なグループ（UNIX システムでは DBA グループ）に入れたり、その OS ユーザー名に特殊なプロセス権を与えたりします。

データベースは、パスワード・ファイルを使って管理者権限を付与されているデータベース・ユーザー名を追跡管理します。

関連項目：『Oracle8 Server 概要』の「ユーザー認証」

オペレーティング・システム認証を使用する

この方法を選択すると、データベース管理操作を実行するユーザーをオペレーティング・システムによって認証させることができます。

1. オペレーティング・システムが認証するユーザーを設定する。
2. 初期化パラメータ `REMOTE_LOGIN_PASSWORD` が `NONE` に設定されていることを確認する。これが、このパラメータのデフォルト値です。
3. 認証されたユーザーは、次のどちらかのコマンドを入力すると、ローカル・データベースに接続したり、安全性の高い接続によってリモート・データベースに接続できます。

```
CONNECT / AS SYSOPER  
CONNECT / AS SYSDBA
```

旧リリースの Oracle で `INTERNAL` として正常に接続する場合は、ステップ 3 に示す新しい構文を使うことにより、引き続き正常に接続できます。

注意： OS 認証を使って `SYSOPER` または `SYSDBA` として接続するときは、`SYSOPER` または `SYSDBA` のシステム権限を付与されている必要はありません。オペレーティング・システム・レベルで適切な `OSDBA` または `OSOPER` ロールを付与されているかどうかをサーバーが検証します。

OSOPER と OSDBA

オペレーティング・システム認証を使うときは、`OSOPER` と `OSDBA` という 2 つの特殊なオペレーティング・システム・ロールによってデータベース管理者のログインが制御されます。

OSOPER	ユーザーは、 <code>STARTUP</code> および <code>SHUTDOWN</code> 、 <code>ALTER DATABASE OPEN/MOUNT</code> 、 <code>ALTER DATABASE BACKUP</code> 、 <code>ARCHIVE LOG</code> 、 <code>RECOVER</code> の実行を許可されます。このロールは <code>RESTRICTED SESSION</code> 権限を持っています。
OSDBA	<code>ADMIN OPTION</code> 付きのすべてのシステム権限と <code>OSOPER</code> ロールを持っています。 <code>CREATE DATABASE</code> と時間ベース回復を許可します。

`OSOPER` と `OSDBA` の名前と機能は、使っているオペレーティング・システムによって異なる可能性があります。

`OSOPER` ロールと `OSDBA` ロールは、オペレーティング・システムだけからユーザーに付与されます。これらのロールは、`GRANT` 文では付与できません。また、これらのロールは取消しも削除もできません。ユーザーが管理者権限でログインするとき、`REMOTE_LOGIN_PASSWORDFILE` が `NONE` に設定されていると、Oracle は、オペレーティング・システムと通信して最初に `OSDBA` を使用可能にしようとし、それが失敗した場合は次に `OSOPER` を使用可能にしようとします。そして両方とも成功しない場合、その接続は失

敗に終わります。オペレーティング・システムからこれらの権限を付与する方法は、オペレーティング・システムによって異なります。

リモート・データベース管理の場合は、Net8 マニュアルを参照して、安全性の高い接続を使っているかどうかを判断してください。TCP/IP や DECnet などの最も一般的な接続プロトコルは、Net8 のどのバージョンを使っても安全性の高い接続ではありません。

関連項目：データベース管理者のオペレーティング・システム認証の詳細は、オペレーティング・システム固有の Oracle マニュアルを参照してください。

認証パスワード・ファイルを使用する

ユーザーがデータベース管理を実行することをパスワード・ファイルを使って認証する必要があると判断した場合は、次に示すステップを実行しなければなりません。各ステップについては、この章の後の項で詳しく説明します。

1. ORAPWD ユーティリティを使ってパスワード・ファイルを作る。

```
ORAPWD FILE=filename PASSWORD=password ENTRIES=max_users
```

2. REMOTE_LOGIN_PASSWORDFILE 初期化パラメータを EXCLUSIVE に設定する。

3. SQL を使ってパスワード・ファイルにユーザーを追加して、次の例のようにデータベース管理を実行する必要がある各ユーザーに適切な権限を付与する。

```
GRANT SYSDBA TO scott  
GRANT SYSOPER TO scott
```

SYSDBA 権限は、OSDBA と同じ操作の実行を許可します。同様に、SYSOPER 権限は OSOPER と同じ操作の実行を許可します。

4. 権限を付与されたユーザーは、次のコマンドを使ってデータベースに接続できます。

```
CONNECT scott/tiger@acct.hq.com AS SYSDBA
```

パスワード・ファイル管理

パスワード・ファイルは、パスワード・ファイル作成ユーティリティ ORAPWD を使って作ります。特定のオペレーティング・システムでは、このファイルを標準インストール作業の一部として作成できます。

この部分のトピックは次のとおりです。

- ORAPWD を使用する
- REMOTE_LOGIN_PASSWORDFILE を設定する
- ユーザーをパスワード・ファイルに追加する
- 管理者権限で接続する
- パスワード・ファイルのメンテナンス

関連項目：インストーラ・ユーティリティでパスワード・ファイルをインストールする方法の詳細は、使っているオペレーティング・システム固有の Oracle のマニュアルを参照してください。

ORAPWD を使用する

パラメータをまったく指定せずにパスワード・ファイル作成ユーティリティを起動すると、次の出力例のようなコマンドの正しい使用方法を示すメッセージが表示されます。

```
orapwd
Usage: orapwd file=<fname> password=<password> entries=<users>
ここで
file - パスワード・ファイルの名前 (mand),
password - SYS と INTERNAL のパスワード (mand),
entries - 各種の DBAs と OPERs の最大数 (opt),
等号 (=) 文字のまわりにスペースはありません。
```

たとえば、次のコマンドを使うと、ACCT.PWD という名前のファイルが作られ、30 人までのユーザーに権限を与えて、それぞれ異なるパスワードが使えるようにできます。このファイルは、最初はパスワード SECRET で、INTERNAL または SYS として接続するユーザー用に作られます。

```
ORAPWD FILE=acct.pwd PASSWORD=secret ENTRIES=30
```

次に、ORAPWD ユーティリティのパラメータについて説明します。

FILE

このパラメータは、作成するパスワード・ファイルの名前を設定します。ファイルには完全なパス名を指定する必要があります。このファイルの内容は暗号化されているため、ユーザーには読めないファイルです。これは必須パラメータです。

パスワード・ファイルで許されているファイル名のタイプは、オペレーティング・システムにより異なります。プラットフォームの中には、パスワード・ファイルが特定のフォーマット（たとえば、orapw<SID>）であり、特定のディレクトリに設定しなければならないものがあります。また、環境変数を使用してパスワード・ファイルの名前と場所を指定できるプラットフォームもあります。プラットフォームで許されている名前と場所については、オペレーティング・システム固有の Oracle マニュアルを参照してください。

Oracle Parallel Server を使って Oracle の複数インスタンスを稼働している場合は、各インスタンスの環境変数が同じパスワード・ファイルを指し示していなければなりません。

警告： パスワード・ファイルやパスワード・ファイルの位置を特定する環境変数の保護は、システムのセキュリティにとって非常に重要です。パスワード・ファイルや環境変数にアクセスできるユーザーは、接続に対するセキュリティを脅かす可能性を潜在的に持っています

PASSWORD

このパラメータは、INTERNAL および SYS 用のパスワードを設定します。データベースに接続した後、ALTER USER コマンドを発行してパスワードを変更すると、データ・ディクショナリに格納されているパスワードとパスワード・ファイルに格納されているパスワードの両方が更新されます。INTERNAL ユーザーは、下位互換性を保つ目的でサポートされています。これは必須パラメータです。

ENTRIES

このパラメータは、パスワード・ファイルに指定できるエントリの最大数を設定します。これは、データベースに SYSDBA または SYSOPER として接続できる異なるユーザーの最大数に対応します。パスワード・ファイルにユーザーを追加したり、パスワード・ファイルからユーザーを削除したりするときに、このエントリは繰り返し使われます。このパスワード・ファイルをいずれ EXCLUSIVE にする必要がある場合には、このパラメータは必須です。

警告： この制限を超える必要がある場合は、新しいパスワード・ファイルを作らなければなりません。必要になると思われる数よりも大きい数を選択するのが最も安全です。

関連項目：パスワード・ファイルの正確な名前、またはオペレーティング・システムにこの名前を指定する環境変数の名前は、使用しているオペレーティング・システム固有の Oracle のマニュアルを参照してください。

REMOTE_LOGIN_PASSWORDFILE を設定する

パスワード・ファイルの作成に加えて、初期化パラメータ REMOTE_LOGIN_PASSWORDFILE を適切な値に設定する必要があります。認識される値を次に示します。

注意： インスタンスまたはデータベースを起動する場合は、必ず Enterprise Manager を使用してください。データベース名とパラメータ・ファイルを指定して、インスタンスの設定を初期化します。Net8 を使って、リモート・データベースの完全修飾名を指定します。ただし、初期化パラメータ・ファイルと構成ファイルなどの関連ファイルが必ずクライアント・マシン上に必要です。つまり、Enterprise Manager を稼働しているマシン上にパラメータ・ファイルが必要です。

NONE

このパラメータを NONE に設定すると、Oracle はパスワード・ファイルがない場合と同じ動作をします。つまり、保護のない接続では権限付きの接続は実現できません。NONE はこのパラメータのデフォルト値です。

EXCLUSIVE

EXCLUSIVE パスワード・ファイルは、1つのデータベースでしか使えません。EXCLUSIVE ファイルだけに SYS と INTERNAL 以外のユーザー名を登録できます。EXCLUSIVE パスワード・ファイルを使うと、個々のユーザーにシステム権限 SYSDBA と SYSOPER を付与し、SYSDBA、SYSOPER 権限で接続させることができます。

SHARED

SHARED パスワード・ファイルは、複数のデータベースで使えます。ただし、SHARED パスワード・ファイルによって認識されるユーザーは SYS と INTERNAL だけです。SHARED パスワード・ファイルではユーザーの追加ができません。SYSDBA または SYSOPER システム権限を必要とするすべてのユーザーは、必ず、SYS とパスワードを使って接続してください。このオプションは、1人のデータベース管理者が複数のデータベースを管理する場合に役に立ちます。

提案： 最も高いレベルのセキュリティを実現するには、パスワード・ファイルを作った後、ただちに REMOTE_LOGIN_PASSWORDFILE ファイルの初期化パラメータを EXCLUSIVE に設定してください。

ユーザーをパスワード・ファイルに追加する

ユーザーに SYSDBA または SYSOPER 権限を付与すると、そのユーザーの名前と権限情報がパスワード・ファイルに追加されます。サーバーに EXCLUSIVE パスワード・ファイルがない場合、つまり、初期化パラメータ REMOTE_LOGIN_PASSWORDFILE が NONE か SHARED である場合は、これらの権限を付与しようとすると、エラー・メッセージが表示されます。

ユーザーがこの2つの権限のうち1つでも持っている間は、そのユーザーの名前がパスワード・ファイルに残っています。ユーザーのこの2つの権限のうちの最後の1つを取り消すと、そのユーザーはパスワード・ファイルから削除されます。

パスワード・ファイルを作って新規ユーザーを追加する手順

1. パスワード・ファイルの作成は、次の指示に従ってください。
2. REMOTE_LOGIN_PASSWORDFILE 初期化パラメータを EXCLUSIVE に設定する。
3. 次の例に示すように、SYSDBA 権限で接続する。

```
CONNECT SYS/change_on_install AS SYSDBA
```

4. 必要であれば、インスタンスを起動してデータベースを作ります。または既存のデータベースをマウントしてオープンする。
5. 必要に応じてユーザーを登録する。必要であれば、データベース管理者自身および適切なユーザーに SYSOPER 権限または SYSDBA 権限を付与します。

- これで、ユーザーはパスワード・ファイルに追加され、(SYS ではなく) ユーザー名とパスワードを使って SYSOPER または SYSDBA としてデータベースに接続できます。OS 認証されたユーザーが OS 認証の基準を満たしていれば、パスワード・ファイルを使うことによって OS 認証されたユーザーの接続が妨げられることはありません。

SYSOPER 権限と SYSDBA 権限の付与と取消し

サーバーで EXCLUSIVE パスワード・ファイルを使っている場合は、GRANT コマンドを使って、次の例に示すようにユーザーに SYSDBA または SYSOPER のシステム権限を付与してください。

```
GRANT SYSDBA TO scott
```

ユーザーのシステム権限 SYSDBA または SYSOPER を取り消すには、次の例のように REVOKE コマンドを使います。

```
REVOKE SYSDBA FROM scott
```

SYSDBA と SYSOPER は、最も強力なデータベース権限であるため、ADMIN OPTION は使われません。現在 SYSDBA (または INTERNAL) として接続しているユーザーだけが、別のユーザーにシステム権限 SYSDBA または SYSOPER を付与できます。これは、REVOKE にも当てはまります。この 2 つの権限は、ロールには付与できません。ロールはデータベースの起動後までは使用可能にならないためです。SYSDBA と SYSOPER のデータベース権限とオペレーティング・システム・ロールを混同しないでください。これらはまったく別の機能です。

関連項目：システム権限の詳細は、第 21 章「ユーザー権限とロールの管理」を参照してください。

パスワード・ファイル・メンバーのリスト

V\$PWFIL_USERS ビューは、1 つのデータベースについてどのユーザーにシステム権限 SYSDBA および SYSOPER が付与されているかを判断するために使います。このビューで表示される列は、次のとおりです。

USERNAME

パスワード・ファイルで認識されるユーザーの名前。

SYSDBA

この列の値が TRUE の場合、そのユーザーは SYSDBA システム権限でログインできます。

SYSOPER

この列の値が TRUE の場合、そのユーザーは SYSOPER システム権限でログインできます。

管理者権限で接続する

ユーザー名とパスワードを使って SYSOPER または SYSDBA 権限で接続した場合は、一般的にユーザー名に対応付けられているスキーマではなく、デフォルトのスキーマ SYS で接続していることになります。

Enterprise Manager の CONNECT コマンドの AS SYSDBA 句または AS SYSOPER 句を使って、管理者権限で接続してください。

管理者権限で接続する例

たとえば、ユーザー SCOTT が次のコマンドを発行したとします。

```
CONNECT scott/tiger
CREATE TABLE scott_test(name VARCHAR2(20));
```

後で、SCOTT が次のコマンドを発行するとします。

```
CONNECT scott/tiger AS SYSDBA
SELECT * FROM scott_test;
```

この場合、SCOTT_TEST が存在しないというエラーを受け取ります。このエラーは、表は SCOTT スキーマ内に作られています、SCOTT が現在デフォルトで SYS スキーマを参照しているために起こります。

保護のないリモート接続

保護のない接続で権限を持つユーザーとして Oracle に接続するには、次の条件を満たしていなければなりません。

- 接続するサーバーにパスワード・ファイルがあること。
- システム権限 SYSOPER または SYSDBA を付与されていること。
- ユーザー名とパスワードを使って接続すること。

ローカル接続および安全性の高いリモート接続

ローカル接続または安全性の高いリモート接続で、権限を持つユーザーとして Oracle に接続するには、次のどちらかの条件を満たしていなければなりません。

- 前述の保護のない接続の項で概説した基準を満たす場合は、パスワード・ファイルを使って接続できる。
- サーバーがパスワード・ファイルを使わない場合、あるいは SYSOPER 権限または SYSDBA 権限を付与されていないため、これらがパスワード・ファイルにない場合は、権限付き接続のために、オペレーティング・システムによってオペレーティング・システム・ユーザー名を認証する必要がある。この認証の形式は、オペレーティング・システムによって異なります。

オペレーティング・システム認証の詳細は、使用しているオペレーティング・システム固有の Oracle のマニュアルを参照してください。

関連項目 :1-8 ページの「パスワード・ファイル管理」

パスワード・ファイルのメンテナンス

ここでは、パスワード・ファイルの拡張および移動、削除について、またパスワード・ファイルの状態を変更しない方法について説明します。

パスワード・ファイルのユーザー数を増やす

ユーザーにシステム権限 SYSDBA または SYSOPER を付与しようとしたときに、ファイルが満杯 (ORA-1996) というエラーが発行された場合は、よりサイズの大きいパスワード・ファイルを作って、ユーザーにもう 1 度権限を付与する必要があります。

パスワード・ファイルを置換する手順

1. V\$PWFFILE_USERS ビューに問い合せて、どのユーザーが SYSDBA 権限または SYSOPER 権限を持っているかを確認する。
2. データベースを停止する。
3. 既存のパスワード・ファイルを削除する。
4. ORAPWD ユーティリティを使ってパスワード・ファイルを新しく作るため、1-9 ページの「ORAPWD を使用する」の指示のとおりにする。ENTRIES パラメータは、必ず十分な大きさの数値に設定してください。
5. 1-11 ページの「パスワード・ファイルを作って新規ユーザーを追加する手順」にある指示に従う。

パスワード・ファイルの位置を移動する

パスワード・ファイルを作った後、希望の位置に移動できます。パスワード・ファイルの位置を移動した後、該当する環境変数を新しいパス名に再設定してください。オペレーティング・システムがあらかじめ定義されたパス名を使う場合は、パスワード・ファイルの位置は変更できません。

パスワード・ファイルを削除する

ユーザーの認証にパスワード・ファイルを使う必要がなくなったと判断した場合、パスワード・ファイルを削除して、REMOTE_LOGIN_PASSWORDFILE 初期化パラメータを NONE にリセットできます。このファイルを削除した後は、オペレーティング・システムによって認証されるユーザーしかデータベース管理操作ができません。

警告： `REMOTE_LOGIN_PASSWORDFILE=EXCLUSIVE`（または `SHARED`）を使ってデータベースまたはインスタンスをマウントしている場合は、パスワード・ファイルを削除したり修正したりしないでください。そのようにすると、パスワード・ファイルを使ってリモートに再接続できなくなります。かわりのパスワード・ファイルを作っても、タイムスタンプとチェックサムが正しくないため、新しいパスワード・ファイルは使えません。

パスワード・ファイルの状態を変更する

パスワード・ファイルの状態は、パスワード・ファイルに格納されます。パスワード・ファイルを初めて作ったときのデフォルトの状態は `SHARED` です。パスワード・ファイルの状態は、`REMOTE_LOGIN_PASSWORDFILE` パラメータを設定すると変更できます。`STARTUP` 文でインスタンスを起動すると、クライアント・マシン上に格納された初期化パラメータ・ファイルからこのパラメータの値が検索されます。データベースをマウントすると、このパラメータの値とパスワード・ファイルに格納された値とが比較されます。これらの値が一致しない場合は、ファイル内の値が上書きされます。

警告： `EXCLUSIVE` パスワード・ファイルを誤って `SHARED` に変更しないように注意してください。複数のクライアントから `STARTUP` 文でインスタンスを起動できるようにするには、各クライアントに初期化パラメータ・ファイルが存在し、各クライアントのファイルの `REMOTE_LOGIN_PASSWORDFILEAND` パラメータの値が一致していなければなりません。それ以外の場合は、パスワード・ファイルの状態は、インスタンスを起動した場所によって変わります。

データベース管理者ユーティリティ

Oracle Server には、そのメンテナンスと制御に役立つ便利なユーティリティがいくつか用意されています。

トピックは、次のとおりです。

- Enterprise Manager
- SQL*Loader
- エクスポートとインポート

Enterprise Manager

Enterprise Manager を使うと、Oracle データベースを監視し制御できます。このマニュアルに記述されている管理的な操作はすべて、Enterprise Manager を使って実行されます。Enterprise Manager は、GUI（グラフィック・ユーザー・インタフェース）とライン・モード・インタフェースの両方を備えています。

Enterprise Manager は、ANSI/ISO 標準 SQL コマンドのスーパーセットを使います。最も一般的な管理コマンドは、Enterprise Manager/GUI のメニューで使えます。頻繁に使わないコマンドは、Enterprise Manager の SQL ワークシートにタイプして実行できます。

関連項目：『Oracle Enterprise Manager 管理者ガイド』

SQL*Loader

SQL*Loader は、Oracle のデータベース管理者とユーザーの両方が使います。SQL*Loader は、オペレーティング・システムの標準ファイル（テキストまたは C データ形式のファイル）から Oracle データベース表にデータをロードします。

関連項目：『Oracle8 Server ユーティリティ』

エクスポートとインポート

Export ユーティリティと Import ユーティリティにより、Oracle データベースとの間で Oracle 形式の既存のデータを移動できます。たとえば、エクスポート・ファイルを使って、データベースのデータをアーカイブしたり、同一のオペレーティング・システムまたは異なるオペレーティング・システム上で稼働している複数の異なる Oracle データベース間でデータを移動できます。

関連項目：『Oracle8 Server ユーティリティ』

データベース管理者の初期優先順位

一般に、データベース管理者は、データベース・システムを編成し、稼働させるための一連の手順を実行し、メンテナンスしなければなりません。

どのタイプのコンピュータ・システムでも、Oracle Server およびデータベースを構成するには、次に示すステップを実行する必要があります。次の項で、各ステップについて説明します。

Oracle Server を構成する手順

- ステップ 1: Oracle ソフトウェアをインストールする
- ステップ 2: データベース・サーバーのハードウェアを評価する
- ステップ 3: データベースを計画する
- ステップ 4: データベースを作成し、オープンする
- ステップ 5: データベース設計をインプリメントする
- ステップ 6: データベースをバックアップする
- ステップ 7: システム・ユーザーを登録する
- ステップ 8: データベース・パフォーマンスを調整する

注意： 新しいリリースへ移行する場合は、既存の本番データベースのバックアップをとってからインストールしてください。すでに作られているデータベース保存方法の詳細は、『Oracle8 Server 移行ガイド』を参照してください。

ステップ 1: Oracle ソフトウェアをインストールする

データベース管理者は、Oracle Server ソフトウェアと、データベースにアクセスするすべてのフロント・エンド・ツールおよびデータベース・アプリケーションをインストールしなければなりません。分散処理環境のインストレーションによっては、データベースが中央のコンピュータによって制御され、データベースのツールとアプリケーションはリモート・マシン上で実行されることがあります。このような場合には、Oracle を実行するコンピュータにリモート・マシンを接続するために必要な Oracle Net8 ドライバもインストールしなければなりません。

関連項目： 詳細は、1-20 ページの「Oracle ソフトウェア・リリースを識別」を参照してください。

インストレーションの特定の要件や指示の詳細は、使っているオペレーティング・システム固有の Oracle のマニュアルと、フロント・エンド・ツールおよび Net8 ドライバのインストレーション・ガイドを参照してください。

ステップ 2: データベース・サーバーのハードウェアを評価する

インストール後、使用可能なコンピュータ・リソースを、Oracle とそのアプリケーションが最大限に活用するにはどうしたらよいかを評価します。この評価では、次のような情報を明らかにする必要があります。

- いくつかのディスク・ドライブを Oracle とそのデータベースに使えるか。
- いくつかの専用テープ・ドライブを Oracle とそのデータベースに使えるか（テープ・ドライブがある場合）。
- どのくらいのメモリーを Oracle インスタンスで使えるか（システムの構成マニュアルを参照）。

ステップ 3: データベースを計画する

データベース管理者は、次のことを計画しなければなりません。

- データベースの論理記憶構造
- 全体のデータベース設計
- データベースのバックアップ計画

重要なことは、データベースの論理記憶構造が、システムのパフォーマンスと種々のデータベース管理操作にどのように影響を及ぼすかについて計画することです。たとえば、いくつかのデータ・ファイルで表領域を構成するのか、データ・ファイルが物理的にどこ（どのディスク・ドライブ）に格納されるのか、どのような種類の情報がそれぞれの表領域に格納されるのかについて、表領域を作る前に把握しておくべきです。データベース全体の論理記憶構造を計画するとき、非常に重要なことは、実際にデータベースが作られ、稼働したときに、この構造によって生じる影響を考えておくことです。次の項目について、データベースの論理記憶構造が及ぼす影響を検討してください。

- Oracle を実行しているコンピュータのパフォーマンス
- データ・アクセス操作の間のデータベースのパフォーマンス
- データベースのバックアップおよびリカバリの手順の効率

また、データベースのオブジェクト間のリレーショナル設計と、各オブジェクトの記憶特性を計画してください。オブジェクトを作る前に、オブジェクトと各オブジェクトの物理記憶間の関連について計画を立てることによって、1つの単位としてのデータベースのパフォーマンスに直接的に影響を与えます。また、データベースの拡張計画についても必ず検討してください。

分散データベース環境では、この計画段階が非常に重要です。頻繁にアクセスされるデータの物理的な位置が、アプリケーションのパフォーマンスにかなり影響を及ぼす可能性があります。

また、この計画段階で、データベースのバックアップも計画してください。この計画を作った後で、バックアップの効率を向上させるために、計画したデータベースの論理記憶構造、またはデータベース設計を変更すべきことに気付くこともあります。

リレーショナル・データベース設計および分散データベース設計については、このマニュアルでは扱っていません。このような設計の問題について詳しく知りたい場合には、これらの問題を説明している、業界標準として一般に認められている資料を参照してください。

関連項目：データベースの論理記憶構造およびオブジェクト、整合性制約の作成の詳細は、第 9 章～第 17 章を参照してください。

ステップ 4: データベースを作成し、オープンする

データベース設計が完了すると、通常の使用のためにデータベースを作り、オープンできます。使っているオペレーティング・システムによっては、Oracle のインストール・プロシージャの中で初期データベースがすでに作られていることがあります。その場合には、インスタンスを起動し、初期データベースをマウントしてからオープンするだけです。

Oracle のインストール中にオペレーティング・システムが初期データベースを作るかどうかを判断するには、インストール・ガイドまたはユーザズ・ガイドを参照してください。インストール時にデータベースが作られない場合、または追加のデータベースを作る場合は、この手順について第 2 章を参照してください。

関連項目：データベースとインスタンスの起動と停止の手順の詳細は、第 3 章を参照してください。

ステップ 5: データベース設計をインプリメントする

データベースの作成および起動が終わったら、必要なすべてのロールバック・セグメントと表領域をすることによって、計画したデータベースの論理構造を作成できます。これが作られると、データベースにオブジェクトを作成できます。

関連項目：データベースの論理記憶構造およびオブジェクトを作るの詳細は、第 8 ～ 17 章を参照してください。

ステップ 6: データベースをバックアップする

データベース構造を作った後、追加の REDO ログ・ファイルを作り、最初の全体データベース・バックアップ（オンラインまたはオフライン）を実行し、その後の定期的なデータベース・バックアップをスケジュールすることによって、データベースのバックアップ計画を実行します。

関連項目：バックアップ操作をカスタマイズする方法と回復手順の実行方法の詳細は、『Oracle8 Server バックアップおよびリカバリ』を参照してください。

ステップ 7: システム・ユーザーを登録する

データベース構造のバックアップが終わったら、Oracle のライセンス契約に従ってデータベースのユーザーを登録し、登録したユーザーのロールを作り、各ユーザーに適切なロールを付与できます。

関連項目：ユーザー・アカウントやロールの作成手順およびライセンス契約の遵守の詳細は、第 18 章～第 20 章を参照してください。

ステップ 8: データベース・パフォーマンスを調整する

データベース・システムのパフォーマンスを最適化することは、データベース管理者の日常的な責任の 1 つです。

関連項目：データベースおよびアプリケーションの調整の詳細は、『Oracle8 Server チューニング』を参照してください。

Oracle ソフトウェア・リリースを識別

Oracle 製品の開発と変更は常に進行しているため、ある時点でユーザーに提供されている製品のリリース番号が複数存在する可能性があります。ソフトウェア製品を完全に見分けるには、3 つの番号が必要となります。

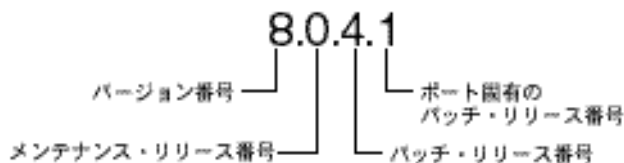
この部分のトピックは次のとおりです。

- リリース番号の形式
- 他の Oracle ソフトウェアのバージョン
- 現行のリリース番号を調べる

リリース番号の形式

たとえば、Oracle Server 配布テープに「リリース 8.0.4.1」というラベルが付いているとします。この番号の各部の意味は次のとおりです。

図 1-2 Oracle のリリース番号の例



バージョン番号

8 のようなバージョン番号が最も一般的な識別子です。バージョンは、通常は重要な新しい機能が含まれる、ソフトウェアの主なエディションを示すものです。

メンテナンス・リリース番号

メンテナンス・リリース番号は、一般的なバージョンの異なるリリース番号を示し、バージョン 8.0 でもそうであるように、0 から始まります。メンテナンス・リリース番号は、バグが修正されたり既存プログラムで新しい機能が利用できるようになったときに大きくなります。

パッチ・リリース番号

パッチ・リリース番号は、8.0.4 のような特定レベルのオブジェクト・コードを識別します。パッチ・リリースには、次のメンテナンス・リリースまで待つことができない重要なバグの修正が含まれています。メンテナンス・リリースの最初の提供版のパッチ番号は、常に 0 です。

ポート固有のパッチ・リリース番号

あるオペレーティング・システム上のソフトウェア製品の特定の緊急パッチ・リリースを識別するのに、8.0.4.1. または 8.0.4.1.3. のような、第 4 の番号（さらに場合によっては第 5 の番号）が使われることがあります。緊急のパッチは、通常広く配布することを意図しているわけではありませんが、通常は、このリリースでは特定の重大な問題が解決されています。

リリース番号の例

Oracle8 のリリース番号の例を次に示します。

8.0.0	Oracle8 の最初の配布
8.1.0	Oracle8 の最初のメンテナンス・リリース
8.2.0	Oracle8 の 2 番目のメンテナンス・リリース（全体では 3 番目のリリース）
8.2.2	2 番目のメンテナンス・リリース後の 2 番目のパッチ・リリース

他の Oracle ソフトウェアのバージョン

オラクル社は、新製品を発表したり、既存製品の機能を拡張しますので、個々の製品のバージョン番号は別々に増えます。したがって、Oracle Server リリース 8.0.12.2 システムを、Oracle Forms バージョン 4.0.3 および SQL*Plus バージョン 3.1.9、Pro*FORTRAN バージョン 1.5.2 とともに稼働させることもあります（これらの番号は具体例として使っているだけです）。

現行のリリース番号を調べる

現在使っている Oracle とそのコンポーネントのリリースを確認するには、次のようにして、データ・ディクショナリ・ビュー `PRODUCT_COMPONENT_VERSION` に問い合わせます（この情報は、Oracle の技術サポートに連絡しなければならないときに役立ちます）。

```
SVRMGR> SELECT * FROM product_component_version;
```

製品名	バージョン	ステータス
-----	-----	-----
CORE	3.4.1.0.0	Production
NLSRTL	3.1.3.0.0	Production
Oracle8 Server	3.2.1.0.0	Beta Release
PL/SQL	2.2.1.0.0	Beta
TNS for SunOS:	2.1.4.0.0	Production
5 行選択		

Oracle データベースの作成

この章では、Oracle データベースを作るために必要な手順を説明します。

- データベースを作成する前の考慮点
- Oracle データベースの作成
- パラメータ
- データベースを作成した後の考慮点
- 初期のチューニング・ガイドライン

データベースを作成する前の考慮点

この部分のトピックは次のとおりです。

- 作成の前提条件
- 初期データベースを使用する
- 旧リリースのデータベースを移行する

データベース作成では、いくつかのオペレーティング・システム・ファイルを準備して、それらが Oracle データベースとして作動するようにします。データベースは、データ・ファイルの数やアクセスするインスタンスの数にかかわらず、1 度だけ作ります。データベースの作成によって、既存のデータベース内の情報を消去し、同じ名前と物理構造を持つデータベースも作成できます。

データベース作成では、次のような操作を実行します。

- 新しいデータ・ファイルを作るか、または既存のデータ・ファイル内のデータを消去する。
- Oracle がデータベースをアクセスし使うために必要となる構造（データ・ディクショナリ）を作成する。
- データベースの制御ファイルと REDO ログ・ファイルを作成して初期化する。

データベースを作る前に、次の点を考慮してください。

- データベース表と索引を計画し、それらが必要とする領域を見積もる。
- オンライン REDO ログとアーカイブ済み REDO ログの構成（そしてこれらが必要とする領域）を含め、データベースを保護する方法を計画し、バックアップ計画を立てる。
- データベースのキャラクタ・セットを選択する。データベースの作成時には、データベースのキャラクタ・セットを指定する必要があります。データベースを作成した後では、再度作成しないと、キャラクタ・セットを変更できません。そのため、使用するキャラクタ・セットの選択は十分注意して行うことが重要です。データ・ディクショナリ内のデータも含め、すべての文字データは、そのデータベースのキャラクタ・セットに格納されます。ユーザーが別のキャラクタ・セットを使ってデータベースにアクセスする場合、データベースのキャラクタ・セットは、ユーザーの使用しているすべてのキャラクタ・セットと同じか、またはそのスーパーセットでなければなりません。

また、インスタンスの起動と停止、データベースのマウントとオープン、パラメータ・ファイルの使用に関する原則とオプションについても理解しておく必要があります。

関連項目：『Oracle8 Server リファレンス・マニュアル』の「各国語サポート」

表および索引、領域管理の詳細は、第 9 章～第 17 章を参照してください。

オンライン REDO ログおよびアーカイブ REDO ログの詳細は、それぞれ第 5 章と第 23 章を参照してください。

データベースのバックアップおよびリカバリの詳細は、『Oracle8 Server バックアップおよびリカバリ』を参照してください。

作成の前提条件

新しいデータベースを作成するには、次の項目が必要です。

- データベース管理者があらゆる操作を実行できるようなオペレーティング・システム権限
- Oracle インスタンスを起動するために十分なメモリー
- Oracle を実行するコンピュータ上に、設計したデータベースのための十分なディスク記憶領域

初期データベースを使用する

使用しているオペレーティング・システムによっては、Oracle のインストール・プロシージャの一部として自動的に初期データベースがすでに作成されていることがあります。この初期データベースを使用して、情報管理要件に一致するようにカスタマイズしたり、この初期データベースを廃棄して、新しいデータベースを作成して置き換えたりできます。

旧リリースのデータベースを移行する

旧リリースの Oracle を使っている場合、データベースを作ることが必要になるのは、まったく新しいデータベースを作るときだけです。それ以外の場合は、旧バージョンの Oracle が管理している既存の Oracle データベースを移行して、新しいバージョンの Oracle ソフトウェアで使用できます。

関連項目：既存のデータベースの移行方法の詳細は、『Oracle8 Server 移行ガイド』を参照してください。

既存のデータベースの移行方法の詳細は、使っているオペレーティング・システム固有の Oracle マニュアルを参照してください。

Oracle データベースの作成

この部分のトピックは次のとおりです。

- Oracle データベースを作成する手順
- データベースの作成例
- データベース作成の問題を解決する
- データベースを削除する

Oracle データベースを作成する手順

次に示す手順は、Oracle データベースを作る方法を示したもので、ここに記述されている順序で必ず実行してください。

新しいデータベースを作り、システムで使用可能にする手順

- 1 既存のデータベースをバックアップする。
- 2 パラメータ・ファイルを作成する。
- 3 新しいパラメータ・ファイルを編集する。
- 4 システムのインスタンス識別子を確認する。
- 5 Enterprise Manager を起動し、Oracle に管理者として接続する。
- 6 インスタンスを起動する。
- 7 データベースを作成する。
- 8 データベースをバックアップする。

関連項目：ここで説明する手順は、すべてのオペレーティング・システムでのデータベース作成についての一般的な情報です。使っているプラットフォーム上でのデータベースの作成の詳細は、オペレーティング・システム固有の Oracle マニュアルを参照してください。

ステップ 1: 既存のデータベースをバックアップする。 データベース作成は既存ファイルに影響を及ぼす可能性がありますので、新しいデータベースを作る前に、すべての既存データベースの完全なバックアップを作ることをご Oracle 社は強くお勧めします。バックアップには、パラメータ・ファイル、データ・ファイル、REDO ログ・ファイル、制御ファイルを含めなければなりません。

ステップ 2: パラメータ・ファイルを作成する。 Oracle データベースのインスタンス（システム・グローバル領域とバックグラウンド・プロセス）は、パラメータ・ファイルを使って起動されます。

システム上の各データベースには、そのデータベースにだけ対応する、カスタマイズしたパラメータ・ファイルを少なくとも 1 つ用意してください。複数のデータベースで同一ファイルを使わないでください。

これから作成するデータベースのパラメータ・ファイルを作成するには、オペレーティング・システムを使用して、オラクル社が配布メディア上に提供しているパラメータ・ファイルのコピーを作成してください。このコピーに新しいファイル名を付けます。その後、新しいデータベース用にこの新しいファイルを編集したり、カスタマイズしたりできます。

関連項目：パラメータ・ファイルのコピーの詳細は、オペレーティング・システム固有の Oracle のマニュアルを参照してください。

注意： 分散処理環境では、Enterprise Manager がネットワーク内のクライアント・マシンから実行されます。クライアント・マシンを使って Enterprise Manager を実行し、新しいデータベースを作成している場合は、新しいパラメータ・ファイル（この時点では Oracle を稼働しているコンピュータ上にある）をクライアント・ワークステーションにコピーする必要があります。この手順は、オペレーティング・システムによって異なります。ネットワーク内のコンピュータ間でファイルをコピーする方法の詳細は、オペレーティング・システム固有の Oracle マニュアルを参照してください。

ステップ 3: 新しいパラメータ・ファイルを編集する。 データベースを新しく作るには、新しいパラメータ・ファイルの中で次のパラメータを必ず検査し、編集してください。

パラメータ	次のものに関する説明
DB_NAME	2-9 ページ
DB_DOMAIN	2-9 ページ
CONTROL_FILES	2-9 ページ
DB_BLOCK_SIZE	2-10 ページ
DB_BLOCK_BUFFERS	2-11 ページ
PROCESSES	2-11 ページ
ROLLBACK_SEGMENTS	2-11 ページ

該当するライセンス・パラメータも必ず編集してください。

パラメータ	次のものに関する説明
LICENSE_MAX_SESSIONS	2-12 ページ
LICENSE_SESSION_WARNING	2-12 ページ
LICENSE_MAX_USERS	2-13 ページ

ステップ 4: システムのインスタンス識別子を確認する。他にデータベースがある場合、Oracle インスタンス識別子をチェックしてください。使用しているシステムで同時に稼働している他の Oracle インスタンスとの混同を避けるため、Oracle のインスタンス識別子は、データベース名（DB_NAME の値）と一致している必要があります。

詳細は、オペレーティング・システム固有の Oracle マニュアルを参照してください。

ステップ 5: Enterprise Manager を起動し、Oracle に管理者として接続する。Enterprise Manager を起動した後は、データベースに管理者として接続してください。

関連項目：Enterprise Manager の起動は、オペレーティング・システムによって異なります。詳細は、オペレーティング・システム固有の Oracle のマニュアルを参照してください。

ステップ 6: インスタンスを起動する。新しいデータベースで使うインスタンス（システム・グローバル領域とバックグラウンド・プロセス）を起動するには、Enterprise Manager の「データベース起動」ダイアログ・ボックスを使います。「データベース起動」ダイアログ・ボックスでは、「アンマウント起動」ラジオ・ボタンが選択されていることを確認してください。

「アンマウント起動」を選択した後、インスタンスが起動されます。この時点では、データベースが存在しません。SGA とバックグラウンド・プロセスだけが、新しいデータベースの作成に備えて起動します。

ステップ 7: データベースを作成する。新しいデータベースを作成するには、SQL コマンド CREATE DATABASE を使用します。この文には、データベースの指定、ファイルの最大数の設定、ファイルの指定とそのサイズの設定など、オプションでパラメータを設定できます。

CREATE DATABASE 文を実行すると、Oracle は次の操作を実行します。

- データベースにデータ・ファイルを作成する。
- データベースに制御ファイルを作成する。
- データベースに REDO ログ・ファイルを作成する。
- SYSTEM 表領域と SYSTEM ロールバック・セグメントを作成する。
- データ・ディクショナリを作成する。
- ユーザー SYS と SYSTEM を作成する。
- データの格納に使うキャラクタ・セットを指定する。
- データベースをマウントし、オープンする。

警告： 指定したデータベースおよび REDO ログ・ファイルが別のデータベースの各ファイルと矛盾しないことを確認してください。

ステップ 8: データベースをバックアップする。 メディア障害が発生した場合に回復するための完全なファイルのセットが確実に存在するように、データベースの完全なバックアップを作る必要があります。

関連項目：『Oracle8 Server バックアップおよびリカバリ』

パラメータ・ファイルの詳細は、3-12 ページの「パラメータ・ファイルの使用」を参照してください。

CREATE DATABASE コマンド、キャラクタ・セット、データベース作成の詳細は、『Oracle8 Server SQL リファレンス』を参照してください。

データベースの作成例

次に、CREATE DATABASE 文の例を示します。

```
CREATE DATABASE test
  DATAFILE 'test_system' SIZE 10M
  LOGFILE GROUP 1 ('test_log1a', 'test_log1b') SIZE 500K,
  GROUP 2 ('test_log2a', 'test_log2b') SIZE 500K;
```

この例で、MAXLOGFILES および MAXLOGMEMBERS、MAXDATAFILES、MAXLOGHISTORY、MAXINSTANCES の各オプションに対する値は、オペレーティング・システムに固有のデフォルト値を想定しています。データベースは、デフォルト・モードの NOARCHIVELOG と EXCLUSIVE でマウントされ、そしてオープンされます。

上の文の項目と情報では、次のような特性を持つデータベースが作られます。

- 新しいデータベースには TEST という名前が付けられる。
- 新しいデータベースの SYSTEM 表領域は、10MB の TEST_SYSTEM という名前の 1 つのデータ・ファイルから構成される。
- 新しいデータベースは、それぞれ 500KB のメンバーを 2 つ含む、2 つのオンライン REDO ログ・グループを持っている。
- 新しいデータベースは、パラメータ・ファイルで指定されている既存の制御ファイルを上書きしない。

注意： データベースを作成するとき、制限をいくつか設定できます。また、これらの制限のいくつかは、オペレーティング・システムの制限にかかわる条件となり、お互いに影響を及ぼす可能性があります。たとえば、MAXDATAFILES を設定した場合、最初はデータベースにデータ・ファイルが 1 つしかなくても、Oracle は MAXDATAFILES だけのファイル名を格納するために十分な領域を制御ファイル内に割り当てます。制御ファイルの最大サイズには制限があり、オペレーティング・システムによって異なりますから、CREATE DATABASE のパラメータすべてをその理論的な最大値に設定できない可能性があります。

関連項目：データベースを作るときに制限を設定する方法の詳細は、『Oracle8 Server SQL リファレンス』を参照してください。

オペレーティング・システムの制限の詳細は、オペレーティング・システム固有の Oracle マニュアルを参照してください。

データベース作成の問題を解決する

なんらかの理由でデータベース作成が失敗した場合は、インスタンスを停止し、CREATE DATABASE 文によって作られたファイルをすべて削除した後で、データベースを作成しなおしてください。

データベース作成の失敗の原因となったエラーを訂正した後で、「Oracle データベースの作成」のステップ 6 に戻ってください。

データベースを削除する

データベースを削除するには、データベースのデータ・ファイルおよび REDO ログ・ファイル、その他の関連ファイル（制御ファイル、パラメータ・ファイル、アーカイブ済みログ・ファイル）すべてを削除します。

データベースのデータ・ファイルと REDO ログ・ファイルの名前を表示するには、データ・ディクショナリ・ビュー V\$DBFILE と V\$LOGFILE に問い合わせます。

関連項目：これらのビューの詳細は、『Oracle8 Server リファレンス・マニュアル』を参照してください。

パラメータ

「Oracle データベースの作成」項のステップ 3 で説明するように、Oracle では最小のパラメータ・セットを変更することを提案しています。これらのパラメータについては、この後の部分で説明します。

- DB_NAME および DB_DOMAIN
- CONTROL_FILES
- DB_BLOCK_SIZE
- PROCESSES
- ROLLBACK_SEGMENTS
- ライセンスに関するパラメータ
- DB_BLOCK_BUFFERS
- LICENSE_MAX_SESSIONS_ と LICENSE_SESSIONS WARNING
- LICENSE_MAX_USERS

DB_NAME および DB_DOMAIN

データベースのグローバル・データベース名（ネットワーク構造内の名前と位置）は、データベースを作る前に、DB_NAME と DB_DOMAIN の両パラメータを設定することによって作られます。作った後では、データベースの名前は簡単には変更できません。DB_DOMAIN パラメータはネットワーク構造内のドメイン（論理的な位置）を示し、DB_NAME パラメータはデータベース名のローカル名構成要素を決定します。これら 2 つのパラメータの設定を組み合わせ、ネットワーク内で一意となるデータベース名を形成する必要があります。たとえば、TEST.US.ACME.COM というグローバル・データベース名を持つデータベースを作るには、次のように新しいパラメータ・ファイルのパラメータを編集します。

```
DB_NAME = TEST
DB_DOMAIN = US.ACME.COM
```

DB_NAME には、8 文字以内のテキスト文字列を設定しなければなりません。データベースの作成時に、DB_NAME に指定した名前は、データベースのデータ・ファイルおよび REDO ログ・ファイル、制御ファイルに記録されます。データベース・インスタンス起動時に（パラメータ・ファイル内の）DB_NAME パラメータの値と制御ファイル内のデータベース名が一致しないと、データベースは起動されません。

DB_DOMAIN は、データベースが作られるネットワーク定義域を指定するテキスト列です。通常、データベースを所有する組織の名前です。作成しようとしているデータベースが分散のデータベース・システムの一部である場合、データベースを作成する前に、この初期化パラメータに特に注意してください。

関連項目：分散データベースの詳細は、『Oracle8 Server 分散システム』を参照してください。

CONTROL_FILES

新しいパラメータ・ファイルに CONTROL_FILES パラメータを指定し、新しいデータベースで使う制御ファイル名（リスト）にその値を設定してください。データベース用の制御ファイルを作るときに Oracle が新しいオペレーティング・システム・ファイルを作るには、CONTROL_FILES パラメータに記述されているファイル名が現在システム上に存在するいずれのファイル名とも一致しないことを確認してください。データベース用の制御ファイルを作るときに Oracle が既存のファイルを再使用、または上書きするには、CONTROL_FILES パラメータに記述されているファイル名が、現在システム上に存在するファイル名と一致することを確認してください。

警告： このオプションを選択する場合には十分に注意してください。不注意で意図しなかったファイルを指定して、CREATE DATABASE 文を実行すると、そのファイルの内容は上書きされてしまいます。

CONTROL_FILES パラメータにファイル名を記述しないと、デフォルト・ファイル名が使われます。

データベースごとに、少なくとも 2 つの制御ファイルを別々の物理ディスク・ドライブに格納して使うことをオラクル社は強くお勧めします。したがって、新しいパラメータ・ファイルの CONTROL_FILES パラメータを指定するときには、次のガイドラインに従ってください。

- CONTROL_FILES パラメータに少なくとも 2 つのファイル名を記述する。
- ファイル名ごとに、異なるディスク・ドライブを参照するファイル名を完全に指定することによって、別々の物理ディスク・ドライブ上に各制御ファイルを配置する。

注意： 制御ファイルのファイル指定は、オペレーティング・システムによって異なります。使用しているオペレーティング・システムとは関係なく、制御ファイルには常にファイル名を省略せずに指定してください。

CREATE DATABASE 文を実行する（ステップ 7）場合、パラメータ・ファイルの CONTROL_FILES パラメータで表示されている制御ファイルが作成されます。

関連項目： CONTROL_FILES パラメータ用のデフォルトのファイル名は、オペレーティング・システムによって異なります。詳細は、オペレーティング・システム固有の Oracle マニュアルを参照してください。

DB_BLOCK_SIZE

Oracle server のデータ・ブロックのデフォルト・サイズは、オペレーティング・システムによって異なります。標準では、Oracle のデータ・ブロック・サイズは 2KB か 4KB のどちらかです。一般に、デフォルトのデータ・ブロック・サイズで十分です。ただし、場合によってはデータ・ブロック・サイズを大きくして、ディスクとメモリーの I/O（データのアクセスと記憶）の効率を向上させることができます。それは次のような場合です。

- Oracle が大容量メモリーと高速ディスク・ドライブを装備した大型コンピュータ・システム上にある場合。たとえば、莫大なハードウェア資源を有するメインフレーム・コンピュータによって制御されるデータベースは、通常 4KB 以上のデータ・ブロック・サイズを使用します。
- Oracle を稼働させるオペレーティング・システムが小さなオペレーティング・システムのブロック・サイズを使う場合。たとえば、オペレーティング・システムのブロック・サイズが 1KB で、データ・ブロック・サイズと一致する場合、Oracle は通常の処理で過度のディスク I/O を実行している可能性があります。この場合にパフォーマンスを最高にするために、データベース・ブロックは複数のオペレーティング・システム・ブロックから構成する必要があります。

各データベースのブロック・サイズは、初期化パラメータ DB_BLOCK_SIZE によってデータベース作成時に設定されます。データベースを作成した後は、データベースを作成し直す以外にブロック・サイズの変更はできません。データベースのブロック・サイズがオペレーティング・システムのブロック・サイズと異なる場合、データベースのデータ・ブロック・サイズは、オペレーティング・システムのブロック・サイズの倍数である必要があります。

たとえば、使っているオペレーティング・システムのブロック・サイズが 2KB(2048 バイト) である場合、DB_BLOCK_SIZE 初期化パラメータに対して次のように設定すると有効です。

```
DB_BLOCK_SIZE=4096
```

また、初期化パラメータ DB_BLOCK_SIZE は、システム・グローバル領域 (SGA) のバッファ・キャッシュ内のデータベース・バッファのサイズも決定します。

関連項目：デフォルトのブロック・サイズの詳細は、オペレーティング・システム固有の Oracle のマニュアルを参照してください。

DB_BLOCK_BUFFERS

このパラメータは、システム・グローバル領域 (SGA) 内のバッファ・キャッシュのバッファ数を決定します。このバッファ数がキャッシュのパフォーマンスに影響を及ぼします。キャッシュ・サイズを大きくすると、修正したデータをディスクに書き込む回数が少なくなります。しかし、キャッシュを大きくすることによって、メモリーを使いすぎて、ページングやスワッピングが発生する可能性もあります。

表、索引、ロールバック・セグメントを含めて、アプリケーションが最も頻繁にアクセスするデータ・ブロックの数を見積もってください。この見積もりが、おおよその、キャッシュが保持すべきバッファの最小値となります。一般に、バッファ数の実際の最小値は 1000 ~ 2000 です。

関連項目：バッファ・キャッシュの調整の詳細は、『Oracle8 Server チューニング』を参照してください。

PROCESSES

このパラメータは、Oracle へ同時に接続できるオペレーティング・システム・プロセスの最大数を決定します。このパラメータには、バックグラウンド・プロセスのための値を 5、ユーザー・プロセスごとの値を 1 として、それらを加算した値を指定しなければなりません。たとえば、50 の同時実行ユーザーを計画している場合、このパラメータを最低でも 55 に設定してください。

ROLLBACK_SEGMENTS

このパラメータは、Oracle インスタンスがデータベースの起動時に獲得するロールバック・セグメントのリストです。このパラメータの値として、ロールバック・セグメントを記述してください。

注意： インストールした後、必ず SYSTEM ロールバック・セグメントの他に SYSTEM 表領域の中に 1 つ以上のロールバック・セグメントを作ってからスキーマ・オブジェクトを作ってください。

関連項目：必要なロールバック・セグメント数の詳細は、『Oracle8 Server チューニング』を参照してください。

ライセンスに関するパラメータ

Oracle は、サイトが Oracle のライセンス契約に従っていることを保証する手助けをします。サイトが同時ユーザーのライセンスを受けている場合、インスタンスに同時に接続するセッション数を追跡し、制限できます。サイトが端末ユーザーによってライセンスされている場合、データベース内に作れる端末ユーザー数を制限できます。この機能を使うには、サイトが持っているライセンス契約の種類とセッションまたは端末ユーザーの最大数を知っている必要があります。サイトは、どちらかの種類のライセンス（セッション・ライセンスまたは端末ユーザー・ライセンス）を使い、両方を使うことはありません。

関連項目：ライセンス管理の詳細は、20-2 ページの「セッションとユーザーのライセンス」を参照してください。

LICENSE_MAX_SESSIONS_ と LICENSE_SESSIONS WARNING

指定したコンピュータ上のデータベースに接続できる同時実行セッションの数に対して、制限を設定できます。インスタンスの同時実行セッションの最大数を設定するには、次の例に示すようにインスタンスを起動するパラメータ・ファイルに LICENSE_MAX_SESSIONS パラメータを設定します。

```
LICENSE_MAX_SESSIONS = 80
```

セッションの最大数の設定に加えて、同時実行セッションの数に対して警告制限を設定できます。この制限に達しても、追加のユーザーは（最大制限まで）接続できますが、Oracle は接続中の各ユーザーに警告を送信します。インスタンスに対して警告制限を設定するには、パラメータ LICENSE_SESSIONS_WARNING を設定します。警告制限は、LICENSE_MAX_SESSIONS よりも小さな値に設定してください。

Parallel Server で実行しているインスタンスの場合、各インスタンスには、固有の同時使用制限と警告制限を設定できます。ただし、それらのインスタンスの制限の合計は、サイトのセッション・ライセンスを超えてはなりません。

関連項目：Parallel Server を使用するときこれらの制限を設定する方法の詳細は、『Oracle8 Parallel Server 概要および管理』を参照してください。

LICENSE_MAX_USERS

データベース内に作るユーザーの数に対して、制限を設定できます。この制限に達すると、それ以上のユーザーは作成できません。

注意： このメカニズムでは、データベースにアクセスする人がそれぞれ一意のユーザー名を持ち、だれもユーザー名を共有していないものと想定しています。したがって、端末ユーザー・ライセンスは、Oracle のライセンス契約に従っていることを確認する手助けができるように、複数のユーザーが同じ名前を使ってログインすることを許可しないでください。

データベース内に作れるユーザー数を制限するには、次の例に示すようにそのデータベースのパラメータ・ファイルに LICENSE_MAX_USERS パラメータを設定します。

```
LICENSE_MAX_USERS = 200
```

Parallel Server で稼働しているインスタンスの場合、同一のデータベースに接続しているインスタンスはすべて同じ端末のユーザー制限を持っています。

関連項目： Parallel Server を使用するときこれらの制限を設定する方法の詳細は、『Oracle8 Parallel Server 概要および管理』を参照してください。

データベースを作成した後の考慮点

データベースを作成した後は、インスタンスは稼働したままであり、データベースはオープンしているので、普通にデータベースを使えます。その後のデータベースの起動と停止には Enterprise Manager を使ってください。データベース・システムに 1 つ以上のデータベースが存在する場合、以降のデータベース起動時に、使うパラメータ・ファイルを指定してください。

このデータベースとともに稼働する他の Oracle 製品をインストールする場合、それらの製品のインストール指示を参照してください。製品によっては、追加のデータ・ディクショナリ表を必ず作ってください。その他の製品の詳細は、オペレーティング・システム固有の Oracle マニュアルを参照してください。通常、データベースのデータ・ディクショナリにこれらの表を作り、データをロードするためのコマンド・ファイルが提供されます。

Oracle Server の配布メディアには、各種の SQL ファイルが格納されています。この SQL ファイルを使うと、システムで試しに操作したり、SQL について学んだり、その他の表、ビュー、シノニムを作成したりできます。

新たに作成したデータベースには、2 人のユーザー、SYS と SYSTEM が存在しています。この 2 つのユーザー名のパスワードは、データベースを作成した直後すぐに変更してください。

関連項目：SYS と SYSTEM の詳細は 1-5 ページの「データベース管理者のユーザー名」を参照してください。

ユーザーのパスワードの変更方法の詳細は、20-14 ページの「ユーザーを変更する」を参照してください。

初期のチューニング・ガイドライン

インストールに続いてただちに、2、3 の重要なチューニング上の変更を Oracle に加えることができます。次の指示に従うと、Oracle を稼働時に調整する必要が少なくなります。ここでは、次のインストール項目に対する推奨事項を示します。

- ロールバック・セグメントを割り当てる
- DB_BLOCK_LRU_LATCHES の数値を選択する
- I/O を分散させる

ロールバック・セグメントを割り当てる

ロールバック・セグメントを適切に割り当てると、データベースのパフォーマンスが最適化されます。最適なパフォーマンスのために必要なロールバック・セグメントのサイズと数は、アプリケーションにより異なります。『Oracle8 Server チューニング』には、Oracle Server 上の同時実行トランザクション数に基づいて割り当てるロールバック・セグメントの数を選択するための一般的なガイドラインが説明されています。ほとんどのアプリケーションでは、このガイドラインが当てはまります。

ロールバック・セグメントを作るには、CREATE ROLLBACK SEGMENT コマンドを使ってください。

関連項目：CREATE ROLLBACK SEGMENT コマンドの詳細は、『Oracle8 Server SQL リファレンス』を参照してください。

ロールバック・セグメントのサイズを選択する

ロールバック・セグメントのサイズもパフォーマンスに影響を及ぼします。ロールバック・セグメントのサイズは、CREATE ROLLBACK SEGMENT 文の記憶領域パラメータによって決定されます。ロールバック・セグメントは、トランザクションのロールバック・エントリを保持できるように十分大きくなければなりません。

関連項目：ロールバック・セグメントのサイズの選択の詳細は、『Oracle8 Server チューニング』を参照してください。

DB_BLOCK_LRU_LATCHES の数値を選択する

LRU ラッチに対する競合は、多数の CPU を持つ対称型マルチプロセッサ (SMP) マシン上ではパフォーマンスを低下させる可能性があります。LRU ラッチは、バッファ・キャッシュ

内のバッファの置換を制御します。SMP システムの場合、Oracle は自動的に LRU ラッチの数をシステム上の CPU の数の半分に設定します。SMP 以外のシステムの場合は、LRU ラッチは 1 つで十分です。

システム上の LRU ラッチの数は、初期化パラメータ `DB_BLOCK_LRU_LATCHES` で指定できます。このパラメータは、必要な LRU ラッチの数の最大値を設定します。各 LRU ラッチは一連のバッファを制御し、Oracle はそれらのバッファ間で置換バッファの割当てを平均化します。

関連項目：LRU ラッチの詳細は、『Oracle8 Server チューニング』を参照してください。

I/O を分散させる

I/O を適切に分散すると、データベースのパフォーマンスをかなり改善できます。I/O は、Oracle のインストール時に分散できます。インストール時に I/O を分散させると、Oracle の稼働時に I/O を分散させる必要を少なくできます。

Oracle をインストールするときに I/O を分散させる方法は、次のようにいくつかあります。

- REDO ログ・ファイルの配置
- データ・ファイルの配置
- 表と索引の分離
- データの密度（データ・ブロックあたりの行）

関連項目：I/O を分散する方法の詳細は、『Oracle8 Server チューニング』を参照してください。

起動と停止

この章では、Oracle データベースの起動と停止の手順について説明します。トピックは、次のとおりです。

- 起動手順
- データベースの可用性の変更
- 停止手順
- パラメータ・ファイルの使用

関連項目：Enterprise Manager/GUI または Server Manager/LineMode を使って特定のタスクを実行する方法の詳細は、『Oracle Enterprise Manager 管理者ガイド』と『Oracle Server Manager ユーザーズ・ガイド』を参照してください。

起動手順

この部分のトピックは次のとおりです。

- インスタンス起動の準備
- インスタンス起動の方法

データベースまたはインスタンスを起動するには、(Oracle に管理者権限で接続した後) Enterprise Manager の「データベースの起動」ダイアログ・ボックスまたは STARTUP コマンドを使います。次のように様々な方法でインスタンスとデータベースを起動できます。

- インスタンスを起動し、データベースはマウントしない。
- インスタンスを起動し、データベースをマウントするが、クローズしたままにする。
- インスタンスを起動し、データベースのマウントを次のどちらかのモードで行い、オープンする。
 - 非制限モード (すべてのユーザーに対して使用可能)
 - 制限モード (DBA に対してだけ使用可能)

注意： マルチスレッド・サーバー・プロセスを介してデータベースに接続している場合は、データベースのインスタンスを起動できません。

さらに、インスタンスを強制的に起動したり、インスタンスの起動直後に完全メディア回復を開始したりできます。使っているオペレーティング・システムが Oracle Parallel Server をサポートしている場合には、インスタンスを起動してから、排他モードまたは共有モードでデータベースをマウントします。

インスタンス起動の準備

インスタンスを起動する前に必ず実行する作業がいくつかあります。

1. Server Manager を起動して、管理者権限で接続する。

データベースやインスタンスを起動するには、Server Manager を起動しなければなりません。また、必ず管理者権限で接続してください。
2. データベース名を指定する。

データベース・インスタンスの起動時に、インスタンスにマウントされるデータベースの名前を次のどちらかの方法で指定してください。

 - STARTUP コマンドを使って、データベース名を指定する。
 - インスタンスの起動に使うパラメータ・ファイルに DB_NAME を指定する。
3. パラメータ・ファイル名を指定する。

データベース・インスタンスの起動時に、パラメータ・ファイルを選択してインスタンスの設定を初期化してください。

- 「データベース起動」ダイアログ・ボックスを使って、「パラメータ・ファイル」テキスト・エントリ・フィールドにファイル名を入力する。
- PFILE オプションを付け、ファイル名を省略せずに指定して STARTUP コマンドを実行する。

関連項目：ファイル名の仕様は、オペレーティング・システムによって異なります。使っているオペレーティング・システム固有の Oracle のマニュアルを参照してください。ファイル名を入力しないと、デフォルトのファイル名が使われます。

インスタンス起動の方法

次に、インスタンスを起動するいくつかの方法を説明します。

注意： 制御ファイルまたはデータベース・ファイル、REDO ログ・ファイルが使えないと、インスタンスの起動で問題が発生することがあります。CONTROL_FILES パラメータで指定された 1 つまたは複数のファイルが存在しない場合、またはオープンできない場合は、警告メッセージが表示され、データベースはマウントされません。データベースをオープンするときに、1 つまたは複数のデータ・ファイルや REDO ログ・ファイルが使えない場合、またはオープンできない場合は、警告メッセージが表示されデータベースはオープンされません。

インスタンスを起動し、データベースをマウントしない方法

データベースをマウントしないでインスタンスを起動する場合もあります。通常は、データベースを作るときに限りこのような要求があります。このためには、以下のいずれかを実行してください。

- Oracle Enterprise Manager の Oracle Instance Manager で、「データベース」プロパティ・シートを表示させ、起動オプションで「マウントしない」を選択する。
- Server Manager で管理者ユーザーで接続し、STARTUP コマンドで、NOMOUNT オプションを指定する。

インスタンスを起動し、データベースをマウントする方法

特定のメンテナンス操作のために、インスタンスを起動し、データベースをマウントしますが、そのデータベースをオープンしないようにしたい場合もあります。たとえば、次のような操作の実行中は、データベースをマウントしても、オープンしてはなりません。

- データ・ファイルを改名する。
- REDO ログ・ファイルを追加、削除、改名する。

- REDO ログ・アーカイブ・オプションを使用可能、または使用禁止にする。
- 完全なデータベース回復を実行する。

インスタンスを起動し、データベースをマウントするが、データベースをクローズしたままにする。

- Oracle Enterprise Manager の Oracle Instance Manager で、「データベース」プロパティ・シートを表示させ、起動オプションで「マウント」を選択する。
- Server Manager で管理者ユーザーで接続し、STARTUP コマンドで、MOUNT オプションを指定する。

インスタンスを起動し、データベースをオープンする方法

通常のデータベース操作とは、インスタンスが起動されており、データベースがマウントされてオープンされていることを意味します。これにより、有効なユーザーはすべてそのデータベースに接続して、典型的なデータ・アクセス操作を実行できるようになります。

インスタンスを起動し、データベースをマウントし、オープンする。

- Oracle Enterprise Manager の Oracle Instance Manager で、「データベース」プロパティ・シートを表示させ、起動オプションで「マウントおよびオープン」を選択する。
- Server Manager で管理者ユーザーで接続し、STARTUP コマンドで、OPEN オプションを指定する。

起動時にデータベースへのアクセスを制限する方法

データベースを管理担当者だけが使えて、データベースの一般ユーザーには使えないようにするために、制限モードでインスタンスを起動して、データベースをマウントし、オープンします。これらの作業の 1 つだけを実行するときは、このデータベース起動モードを使ってください。

- 索引の再作成のように、構造上のメンテナンスを実行する。
- データベース・データのエクスポートまたはインポートを実行する。
- (SQL*Loader によって) データ・ロードを実行する。
- 一時的に一般ユーザーがデータを使えないようにする。

一般に、CREATE SESSION システム権限を持つすべてのユーザーは、オープンしているデータベースに接続できます。制限モードでデータベースをオープンすると、CREATE SESSION システム権限と RESTRICTED SESSION システム権限の両方を持つユーザーだけにデータベース・アクセスを許可します。したがって、RESTRICTED SESSION システム権限は、データベース管理者だけが持つようにしてください。

制限モードでインスタンスを起動し、データベースをマウントし、オープンする。

- Oracle Enterprise Manager の Oracle Instance Manager で、「データベース」プロパティ・シートを表示させ、起動オプションで「制限」を選択する。

- Server Manager で管理者ユーザーで接続し、STARTUP コマンドで、RESTRICT オプションを指定する。

その後、RESTRICTED SESSION システム権限を持たないユーザーがデータベースにアクセスできるようにもできます。

インスタンスを強制的に起動する方法

データベース・インスタンスを起動しようとして、問題の発生することがまれにあります。次の問題が発生している場合を除いては、データベースのインスタンスを強制的に起動しないでください。

- 現行のインスタンスをうまく停止できない場合
- インスタンス起動時に問題が発生した場合

このような問題が発生した場合、以下のどちらかを実行して、新しいインスタンスを起動（オプションとしてデータベースをマウントし、オープン）すると、通常は問題を解決できます。

- Oracle Enterprise Manager の Oracle Instance Manager で、「データベース」プロパティ・シートを表示させ、起動オプションで「強制」を選択する。
- Server Manager で管理者ユーザーで接続し、STARTUP コマンドで、FORCE オプションを指定する。

インスタンスを起動し、データベースをマウントし、完全メディア回復を開始する方法

メディア回復が必要な場合には、以下を実行し、インスタンスを起動し、データベースをインスタンスにマウントし、回復処理を自動的に開始できます。Server Manager で管理者ユーザーで接続し、STARTUP コマンドで、RECOVER オプションを指定する。

排他モードまたはパラレル・モードで起動する方法

Oracle Server で、複数インスタンスが同時に 1 つのデータベースにアクセスできる場合、データベースを排他的にマウントするかパラレルにマウントするかを必ず選択します。

インスタンスとデータベースの起動例

次の例では、INITSALE.ORA という名前のパラメータ・ファイルを使ってインスタンスを起動し、SALES という名前のデータベースを排他モードでマウント、オープンし、アクセスを管理担当者に制限しています。データベース管理者はすでに管理者権限で接続しています。

```
STARTUP OPEN sales PFILE=INITSALE.ORA EXCLUSIVE RESTRICT;
```

オペレーティング・システム起動時のデータベース自動起動の方法

システム起動に続いて、すぐに 1 つ以上の Oracle インスタンスとデータベースの自動起動を使用可能にするための手順が、多くのサイトで使われます。そのための手順は、オペレーティング・システムによって異なります。

リモート・インスタンスを起動する方法

ローカルの Oracle Server が分散データベースの一部を構成している場合は、リモート・インスタンスとデータベースを起動しなければならないことがあります。リモート・インスタンスの起動と停止の手順は、通信プロトコルとオペレーティング・システムに依存し、かなり異なります。

関連項目：権限を持たないユーザーにデータベースを使えるようにする方法の詳細は、3-7 ページの「オープン状態のデータベースへのアクセスを制限する方法」を参照してください。

制御ファイル、データベース・ファイル、REDO ログの回復の詳細は、『Oracle8 Server バックアップおよびリカバリ』を参照してください。

現行のインスタンスの異常終了の副作用の詳細は、3-12 ページの「インスタンスを中断する方法」を参照してください。

排他モードまたはパラレル・モードでの起動方法の詳細は、『Oracle8 Parallel Server 概要および管理』を参照してください。

STARTUP コマンドのオプションを組み合わせるときに適用される制限の詳細は、『Oracle8 Server SQL リファレンス』を参照してください。

自動的な起動手順のトピックの詳細は、オペレーティング・システム固有の Oracle のマニュアルを参照してください。

データベースの可用性の変更

マウントはされているが、クローズされているデータベースをオープンすることによって、データベースを部分的に使用可能にし、ユーザーがそのデータベースに接続して使えます。

この後の部分では、データベースの可用性の変更方法について説明します。

- インスタンスにデータベースをマウントする方法
- クローズされたデータベースをオープンする方法
- オープン状態のデータベースへのアクセスを制限する方法

インスタンスにデータベースをマウントする方法

特定の管理操作の必要があるとき、データベースは、起動され、インスタンスにマウントされていても、クローズされていなければなりません。これは、インスタンスを起動しデータベースをマウントすることによってできます。

データベースをマウントするときに、データベースをそのインスタンスだけに排他的にマウントするのか、同時に他のインスタンスにもマウントするのかを指示できます。

すでに起動されているインスタンスにデータベースをマウントするには、以下のいずれかを実行します。

- Oracle Enterprise Manager の Oracle Instance Manager で、「データベース」メニューから「マウント」を選択する。

Server Manager で MOUNT オプションを指定する。

- SQL コマンドで、ALTER DATABASE の MOUNT オプションを指定する。

データベースを排他モードでマウントするには、次の文を使います。

```
ALTER DATABASE MOUNT;
```

関連項目：データベースがマウントされ、クローズされている必要のある操作（および、インスタンスの起動とデータベースのマウントを一度に実行する手順）の詳細は、3-3 ページの「インスタンスを起動し、データベースをマウントする方法」を参照してください。

クローズされたデータベースをオープンする方法

データベースをオープンすることによって、マウント済みのクローズされているデータベースを一般的な用途のために使用可能にできます。マウント済みのデータベースをオープンするには、以下のいずれかを実行します。

- Oracle Enterprise Manager の Oracle Instance Manager で、「データベース」メニューから「オープン」を選択する。

Server Manager で OPEN オプションを指定する。

- SQL コマンドで、ALTER DATABASE の OPEN オプションを指定する。

マウントされたデータベースをオープンするには、次の文を使います。

```
ALTER DATABASE OPEN;
```

この文の実行後は、CREATE SESSION システム権限を与えられた正規の Oracle ユーザーであれば、データベースに接続できます。

オープン状態のデータベースへのアクセスを制限する方法

正常な条件下では、CREATE SESSION システム権限を持つすべてのユーザーがインスタンスに接続できます。ただし、インスタンスは制限モード、または非制限モードにできます。インスタンスが制限モードになっていると、CREATE SESSION システム権限と RESTRICTED SESSION システム権限を持っているユーザーだけがインスタンスに接続できます。通常、管理者だけが RESTRICTED SESSION システム権限を持っています。

次の作業の必要があるときに制限モードが有効です。

- 索引の再編成のように、構造上のメンテナンスを実行する。
- データベース・データのエクスポートまたはインポートを実行する。
- （SQL*Loader によって）データ・ロードを実行する。
- 管理者以外のユーザーがデータを一時的に使えないようにする。

インスタンスを制限モードにするには、以下のいずれかを実行します。Oracle Enterprise Manager の Oracle Instance Manager で、「データベース」プロパティ・シートを表示させ、起動オプションで「制限」を選択する。

SQL コマンド、ALTER SYSTEM の ENABLE RESTRICTED SESSION オプションを指定する。インスタンスを制限モードにした後、管理作業を実行する前に現行のユーザー・セッションをすべて停止（KILL）する場合があります。

インスタンスの制限モードを解除するには、以下のいずれかを実行します。Oracle Enterprise Manager の Oracle Instance Manager で、「データベース」プロパティ・シートを表示させ、起動オプションで「許可」を選択する。

SQL コマンド、ALTER SYSTEM の DISABLE RESTRICTED SESSION オプションを指定する。

データベース・インスタンスの起動、および制限モードでのデータベースのマウントとオープンの詳細は、3-4 ページの「起動時にデータベースへのアクセスを制限する方法」を参照してください。

停止手順

ここでは、次の様々な手順について詳しく説明します。

- 通常の条件下でデータベースを停止する方法
- データベースを即時停止する方法
- シャットダウン・トランザクション

注意： SHUTDOWN IMMEDIATE 文は既存のアイドル接続をすべて切断し、データベースを停止します。ただし、結果待ちになっているプロセス（挿入、選択、更新など）を送った場合、SHUTDOWN IMMEDIATE 文でプロセスは切断する前に終了します。

データベースの停止操作を行うには、以下のいずれかを実行します。Oracle Enterprise Manager の Oracle Instance Manager で、「データベース」プロパティ・シートを表示させ、停止オプションで「標準」を選択する。

Server Manager で管理者ユーザーで接続し、SHUTDOWN コマンドを実行する。停止が完了するまで、データベース停止を開始するセッションに制御が戻りません。停止処理の進行中に接続しようとするユーザーは、次のようなメッセージを受け取ります。

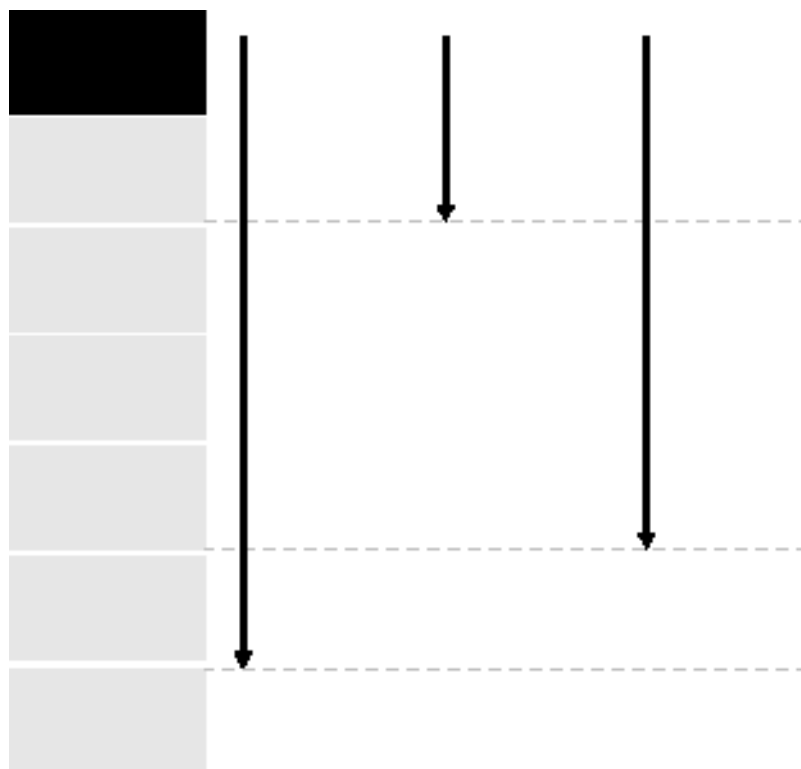
ORA-01090: 停止処理中なので接続はできません。

注意： マルチスレッド・サーバー・プロセスを介してデータベースに接続している場合は、データベースは停止できません。

データベースやインスタンスを停止するには、管理者権限で接続する必要があります。Oracle Enterprise Manager/GUI と Server Manager/LineMode(DBA コマンド) のどちらかを使う場合でも、この条件は適用されます。

図 3-1 には、ある銀行口座から別の銀行口座への資金の振替え時に各種の SHUTDOWN コマンドを入力するときのイベントの順序を示しています。

図 3-1 各種の SHUTDOWN 時のイベントの順序



通常の条件下でデータベースを停止する方法

通常のデータベース停止は、次のように処理を進めます。

- 文が発行された後、どのような新しい接続も許さない。
- データベースが停止される前に、Oracle は現在データベースに接続しているすべてのユーザーが切断されるのを待つ。
- データベースの次の起動では、どのようなインスタンス回復手順も要求しない。

通常の状況でデータベースを停止するには、以下のいずれかを実行します。

- Oracle Enterprise Manager の Oracle Instance Manager で、「データベース」プロパティ・シートを表示させ、停止オプションで「標準」を選択する。

- Server Manager で管理者ユーザーで接続し、SHUTDOWN コマンドで NORMAL オプションを指定する。

データベースを即時停止する方法

データベースの即時停止は、次のような状況のときにだけ使います。

- 電源供給が間もなく停止する。
- データベースまたはデータベース・アプリケーションの 1 つが不規則に作動している。

このデータベース停止は次のように処理を進めます。

- Oracle によって処理されている現行のクライアント SQL 文をただちに終了させる。
- コミットされていないトランザクションをロールバックする。コミットされていないトランザクションはすべてロールバックされる（コミットされていない大規模なトランザクションが存在する場合、この停止方法は、その名前に反してただちにされない可能性があります）。
- Oracle は現在データベースに接続しているユーザーが切断されるのを待たない。Oracle はアクティブ・トランザクションを暗黙のうちにロールバックしてから、接続しているユーザーをすべて切断します。

ただちにデータベースを停止するには、以下のいずれかを実行します。

- Oracle Enterprise Manager の Oracle Instance Manager で、「データベース」プロパティ・シートを表示させ、停止オプションで「即時」を選択する。
- Server Manager で管理者ユーザーで接続し、SHUTDOWN コマンドの IMMEDIATE オプションを指定する。

注意： SHUTDOWN IMMEDIATE 文は既存のアイドル接続をすべて切断し、データベースを停止します。ただし、結果待ちになっているプロセス（挿入、選択、更新など）を送った場合、SHUTDOWN IMMEDIATE 文でプロセスは切断する前に終了します。

シャットダウン・トランザクション

クライアントに対する割込みを最小限にしながら予定どおりのインスタンスの停止を行う場合、TRANSACTIONAL オプションを指定した SHUTDOWN コマンドを使えます。

```
SHUTDOWN TRANSACTIONAL;
```

この文を送った後、クライアントはこの特定のインスタンスで新しいトランザクションを起動できません。クライアントが新しいトランザクションを起動しようすると、切断されます。すべてのトランザクションがコミットまたは異常終了すると、インスタンスにまだ接続されているすべてのクライアントがすべて切断されます。このとき、インスタンスは SHUTDOWN IMMEDIATE 文を送る場合のように停止します。

トランザクションのシャットダウンはクライアントが作業を失わないようにすると同時に、すべてのユーザーがログオフする必要はありません。

インスタンスを中断する方法

データベースのインスタンスを中断することによって、データベースをただちに停止できます。ただし、このような停止はなるべく避けてください。このタイプの停止は、次のような状況だけで実行するようにしてください。

- データベースまたはデータベース・アプリケーションの 1 つが不規則に作動していて、かつ、他のタイプの停止がどれも作動しない。
- データベースを即時停止する必要がある（たとえば、電源供給停止が 1 分以内に起こることがわかっている）。
- データベース・インスタンスを起動するときに問題が発生した。

インスタンスの中断によって、データベースは次のように停止されます。

- Oracle によって処理されている現行のクライアント SQL 文をただちに終了させる。
- コミットされていないトランザクションをロールバックしない。
- Oracle は現在データベースに接続しているユーザーが切断されるのを待たない。Oracle は暗黙のうちに接続しているユーザーをすべて切断します。

通常の停止オプションと即時停止オプションのどちらも作動しない場合は、以下のいずれかを実行して、現行のデータベース・インスタンスをただちに中断させます。

- Oracle Enterprise Manager の Oracle Instance Manager で、「データベース」プロパティ・シートを表示させ、停止オプションで「異常終了」を選択する。
- Server Manager で管理者ユーザーで接続し、SHUTDOWN コマンドの ABORT オプションを指定する。

パラメータ・ファイルの使用

この後の部分では、パラメータ・ファイルの使用方法について説明します。

- サンプル・パラメータ・ファイル
- パラメータ・ファイルの数
- 分散環境におけるパラメータ・ファイルの位置

インスタンスを起動するためには、Oracle はインスタンス構成パラメータのリストが入っているテキスト・ファイルである、「パラメータ・ファイル」を読み込まなければなりません。必ずというわけではありませんが、通常、このファイルの名前は INIT.ORA または INITsid.ORA です。sid はオペレーティング・システムによって異なります。

パラメータ・ファイル内のパラメータ値は基本的なテキスト・エディタで編集できますが、編集方法はオペレーティング・システムによって異なります。

ファイル内の各国語サポート (NLS) パラメータに定義されている文字列リテラルは、データベースのキャラクタ・セットであるかのように扱われます。

関連項目：INITsid.ORA の詳細は、オペレーティング・システム固有の Oracle のマニュアルを参照してください。

サンプル・パラメータ・ファイル

サンプルのパラメータ・ファイル (INIT.ORA または INITsid.ORA) が、Oracle の配布セットに含まれています。このサンプル・ファイルのパラメータは、Oracle データベースの初期インストールのためには十分なものです。システムが稼働してから、Oracle をいくらか使ってみた後で、パラメータ値のいくつかを変更したいと思うかもしれません。

関連項目：パラメータ・ファイルを使用したデータベースのパフォーマンスの最適化の詳細は、『Oracle8 Server チューニング』を参照してください。

パラメータ・ファイルの数

Oracle データベースには、そのデータベースだけに対応するパラメータ・ファイルが 1 つ以上あります。このように、与えられたファイル内のデータベース特性パラメータ (DB_NAME、CONTROL_FILES など) は、常に特定のデータベースに関係があります。1 つのデータベースに複数の異なるパラメータ・ファイルを用意することもできます。たとえば、いろいろな状況でデータベースのパフォーマンスを最適化するために、1 つのデータベースに複数の異なるパラメータ・ファイルを用意できます。

分散環境におけるパラメータ・ファイルの位置

Oracle Enterprise Manager には、必ず、データベースのパラメータ・ファイルを読み込んでデータベースのインスタンスを起動する機能が必要です。したがって、データベースのパラメータ・ファイルは、必ず Oracle Enterprise Manager を稼働するコンピュータ (WindowsNT、または Windows95) 上に格納してください。

たとえば、分散処理構成でない場合、同一のコンピュータで Oracle と Oracle Enterprise Manager が実行されます。このコンピュータには、すでに、パラメータ・ファイルがディスク・ドライブのうちの 1 つに格納されています。

一方、分散処理構成では、ローカルのクライアント・ワークステーションで Oracle Enterprise Manager を稼働して、リモート・マシン上に格納されているデータベースを管理できます。このタイプの構成では、ローカルのクライアント・マシンには、対応するデータベースの各パラメータ・ファイルのコピーを格納しなければなりません。

関連項目：分散環境で管理用の Oracle を使用方法の詳細は、『Oracle8 Server 分散システム』を参照してください。

Enterprise Manager のセットアップとインプリメントの詳細は、オペレーティング・システム固有の Oracle のマニュアルを参照してください。

第 2 部

Oracle Server の構成

Oracle プロセスの管理

この章では、Oracle インスタンスのプロセスを管理する方法について説明します。トピックは次のとおりです。

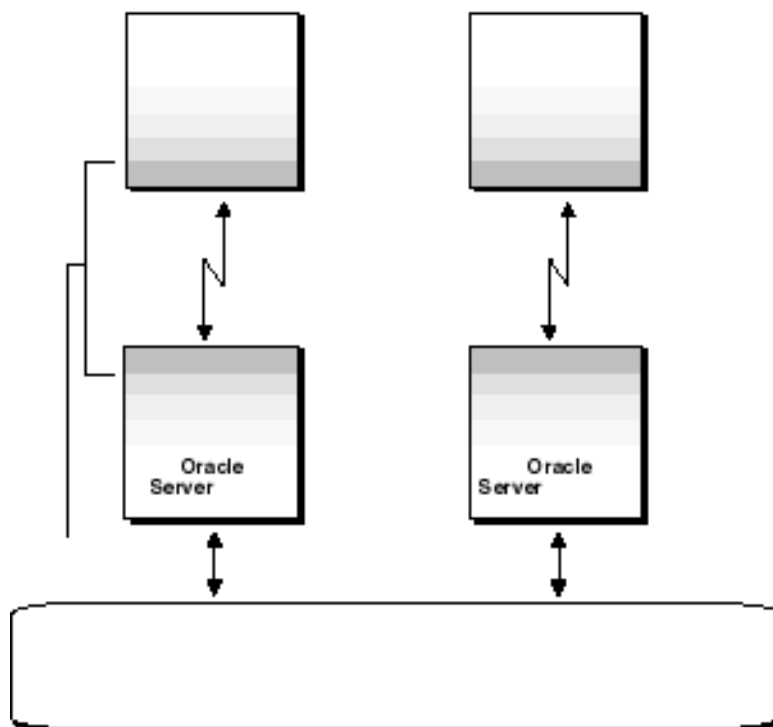
- 専用サーバー・プロセス用に Oracle を構成
- マルチスレッド・サーバー・プロセス用に Oracle を構成
- サーバー・プロセスの変更
- Oracle プロセスを追跡
- 並列問合せオプションのプロセスの管理
- 外部プロシージャのプロセスの管理
- セッションの停止

関連項目：Enterprise Manager/GUI または Server Manager/LineMode を使って特定のタスクを実行する方法の詳細は、『Oracle Enterprise Manager 管理者ガイド』および『Oracle Server Manager ユーザーズ・ガイド』を参照してください。

専用サーバー・プロセス用に Oracle を構成

ユーザー・プロセスが、あるマシン上でデータベース・アプリケーションを実行し、サーバー・プロセスが別のマシン上で対応付けられた Oracle Server を実行すると、別個の固有のプロセスが作られます。各ユーザーのために作られた別個のサーバー・プロセス専用サーバー・プロセスといいます（図 4-1 を参照）。Oracle は自動的にこの構成用にインストールされます。使っているオペレーティング・システムが、この構成で Oracle をサポートできる場合、マルチスレッド・サーバーもサポートする可能性があります。

図 4-1 Oracle 専用サーバー・プロセス



専用サーバー構成でインスタンスを起動するには、(パラメータ・ファイル内の) 次の初期化パラメータを NULL に設定するか、またはファイルにこれらのパラメータを指定しないでください。

- MTS_SERVICE
- MTS_DISPATCHERS
- MTS_SERVERS
- MTS_LISTENER_ADDRESS

専用サーバー・プロセスに接続する場合

可能であれば、ディスパッチャを介してインスタンスに接続してください。これにより、実行中のインスタンスに必要なプロセスの数を少なくできます。ただし、次の状況では、ユーザーとデータベース管理者は、専用サーバー・プロセスを使って明示的にインスタンスに接続しなければなりません。

- バッチ・ジョブを実行する (たとえば、ジョブがサーバー・プロセスに対して持つアイドル時間が少ないか、まったくない場合)
- Enterprise Manager を使ってデータベースを起動または停止するか、あるいはデータベース上でメディア回復を実行する。

専用サーバー接続を要求するには、ユーザーは Net8 TNS 接続文字列に SRVR=DEDICATED を指定しなければなりません。

関連項目: Net8 接続文字列の構文の詳細は、使っているオペレーティング・システム固有の Oracle のマニュアルおよび Net8 のマニュアルを参照してください。

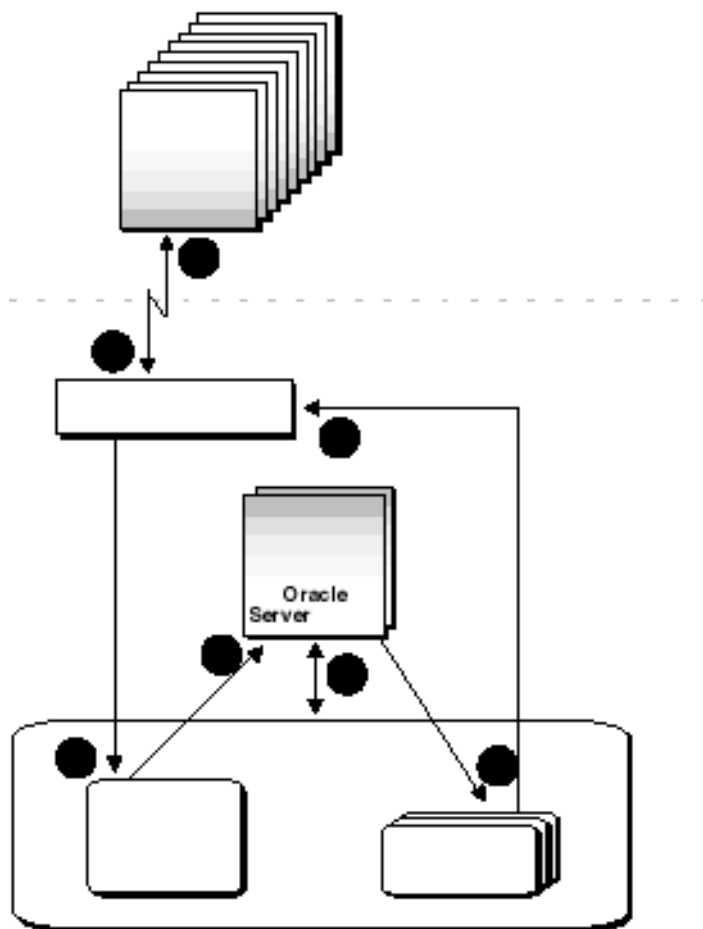
初期化パラメータとパラメータ・ファイルの詳細は、『Oracle8 Server リファレンス・マニュアル』を参照してください。

マルチスレッド・サーバー・プロセス用に Oracle を構成

専用サーバー・プロセスを持つオーダー・エントリ・システムを例にとって考えてみます。クレークがオーダーをデータベースに入力すると、カスタマはオーダーを入れます。大部分のトランザクションでは、クレークはカスタマと電話中で、クレークのユーザー・プロセス専用のサーバー・プロセスはアイドル状態のままです。サーバー・プロセスはトランザクション中はほとんど必要とされず、その他のクレークがオーダーを入力するとシステムは遅くなります。

マルチスレッド・サーバー構成では、接続ごとに専用サーバー・プロセスが必要ありません (図 4-2 を参照)。少数の共有サーバー・プロセスで多数の専用サーバー・プロセスと同じ量の処理を実行できます。また、各ユーザーに必要なメモリーの量は比較的少なくて済みます。必要なメモリーおよびプロセス管理が少ないので、より多くのユーザーをサポートできます。

図 4-2 Oracle のマルチスレッド・サーバー・プロセス



マルチスレッド・サーバー構成でシステムをセットアップするには、ネットワーク・リスナー・プロセスを起動して、次に示す初期化パラメータを設定します。

- SHARED_POOL_SIZE
- MTS_LISTENER_ADDRESS
- MTS_SERVICE
- MTS_DISPATCHERS
- MTS_MAX_DISPATCHERS
- MTS_SERVERS
- MTS_MAX_SERVERS

以上の初期化パラメータを設定した後でインスタンスを再起動すると、その時点でマルチスレッド・サーバー構成が使われます。マルチスレッド・サーバー・アーキテクチャでは、Net8 が必要です。マルチスレッド・サーバーを使用するユーザー・プロセスは、Oracle インスタンスと同じマシン上にある場合でも、必ず Net8 を介して接続してください。

関連項目：ネットワーク・リスナー・プロセスの起動と管理の詳細は、『Oracle8 Server 分散システム』と『Net8 管理者ガイド』を参照してください。

SHARED_POOL_SIZE: 共有サーバーの共有プールに追加領域を割り当てる

マルチスレッド・サーバーを通じて接続するとき、Oracle は、ユーザー・プロセスおよびディスパッチャ、サーバーの間の接続に関する情報を格納するために、共有プール内に追加の領域を割り当てなければなりません。マルチスレッド・サーバーを使って接続するユーザーごとに、パラメータ SHARED_POOL_SIZE の設定に 1K を追加してください。

関連項目：このパラメータの詳細は、『Oracle8 Server リファレンス・マニュアル』を参照してください。

チューニング方法の詳細は、『Oracle8 Server チューニング』を参照してください。

MTS_LISTENER_ADDRESS: リスナー・プロセスのアドレスを設定する

データベースのパラメータ・ファイルには、データベースが接続する各ポートに対して初期化パラメータ MTS_LISTENER_ADDRESS を設定します。このパラメータは、次の構文をサポートします。

```
MTS_LISTENER_ADDRESS = "(addr)"
```

この構文では、addr はリスナーが特定のプロトコルの接続要求を確認するアドレスです。パラメータ・ファイルには複数のアドレスを格納できます。

次の例は、リスナー・アドレスを指定するものです。

```
MTS_LISTENER_ADDRESS = "(ADDRESS=(PROTOCOL=tcp) (PORT=5000)¥  
(HOST=ZEUS))"
```

```
MTS_LISTENER_ADDRESS = "(ADDRESS=(PROTOCOL=decnet)¥  
(OBJECT=OUTA) (NODE=ZEUS))"
```

データベースのパラメータ・ファイルに指定する各アドレスは、対応するリスナーの構成ファイルにも指定しなければなりません。ネットワーク・プロトコルごとに異なるアドレスを指定します。

関連項目：ネットワーク・リスナー・プロセスのアドレス指定の詳細は、使っているオペレーティング・システム固有の Oracle マニュアルおよび Net8 のマニュアルを参照してください。

MTS_SERVICE: ディスパッチャに対してサービス名を指定する

リスナーに対応付けるサービスの名前をパラメータ MTS_SERVICE を使って指定してください。ユーザーは、接続文字列にこのサービス名を指定することによって、マルチスレッド・サーバーを要求します。サービス名は、一意でなければなりません。できるだけ、インスタンスの SID (システム識別子) を使ってください。

MTS_SERVICE パラメータを設定しない場合、デフォルトで DB_NAME パラメータが使われます。(DB_NAME も設定しないと、データベース起動時に ORA-00114 エラー「システム・パラメータ MTS_SERVICE の値が指定されていません」が戻されます。)

ディスパッチャのサービス名が TEST_DB の場合、このパラメータは次のように設定します。

```
MTS_SERVICE = "test_db"
```

このディスパッチャに接続するための接続文字列は、次のようになります。

```
SQLPLUS scott/tiger@¥  
(DESCRIPTION=(ADDRESS=(PROTOCOL=decnet) (NODE=hq)¥  
(OBJECT=mts7)) (CONNECT_DATA=(SID=test_db)))
```

関連項目：マルチスレッド・サーバー構成で使われる接続文字列の詳細は、使っているオペレーティング・システム固有の Oracle または Net8 のマニュアルを参照してください。

MTS_DISPATCHERS: ディスパッチャの最初の数を設定する

インスタンス起動時に起動されるディスパッチャ・プロセスの数は、パラメータ MTS_DISPATCHERS によって制御されます。インスタンスを起動する前に、各ネットワーク・プロトコルに対して、起動するディスパッチャの数を見積もってください。

MTS_DISPATCHERS パラメータを設定するときには、有効なプロトコルをすべて対象とすることが出来ます。

各インスタンスに対するディスパッチャ・プロセスの適切な数は、必要なデータベースのパフォーマンス、オペレーティング・システムによって異なるプロセスあたりの接続数に対するホスト・オペレーティング・システムの制限、およびネットワーク・プロトコルあたりに必要とされる接続数に依存します。

インスタンスは、データベース・システム上の同時実行ユーザーと同数の接続を提供できなければなりません。ディスパッチャが多ければ、ディスパッチャ・サービスを長い間待機する必要はないため、ユーザーに見えるデータベースのパフォーマンスは向上します。

インスタンスを起動した後、必要であればさらにディスパッチャ・プロセスを起動できますが、起動できるのは、データベースのパラメータ・ファイルで指定したプロトコルを使うディスパッチャだけです。たとえば、パラメータ・ファイルによってプロトコル A とプロトコル B に対してディスパッチャが起動される場合、パラメータ・ファイルを変更し、インスタンスを再起動しないで、後からプロトコル C のディスパッチャを起動できません。

関連項目：ディスパッチャ・プロセスの詳細は、4-9 ページの「ディスパッチャ・プロセスを追加、削除する」を参照してください。

ディスパッチャ・プロセスの最初の数を計算する

使っているオペレーティング・システムについて、プロセスあたりに可能な接続数が分かれば、ネットワーク・プロトコルごとに、インスタンス起動時に作るディスパッチャ・プロセスの最初の数を、次の公式を使って計算してください。

$$\begin{array}{lcl} \text{number} & & \text{maximum number of concurrent sessions} \\ \text{of} & = \text{CEIL} & (\text{-----} \\) & & \\ \text{dispatchers} & & \text{connections per dispatcher} \end{array}$$

注意：ここで、ディスパッチャごとの接続数の値はオペレーティング・システムによって違います。.

たとえば、通常、TCP/IP を介して 80 人のユーザーが同時に接続し、DECNet を介して 40 人のユーザーが接続するシステムがあるとします。この場合、MTS_DISPATCHERS パラメータは、次のように設定します。

```
MTS_DISPATCHERS = "(PROTOCOL=TCP) (DISPATCHERS=3)"
MTS_DISPATCHERS = "(PROTOCOL=DECNET) (DISPATCHERS=3)"
```

例

例 1 ディスパッチャが使う IP アドレスを設定するには、次のように入力します。

```
MTS_DISPATCHERS="(ADDRESS=(PARTIAL=TRUE) (PROTOCOL=TCP)¥
(HOST=144.25.16.201)) (DISPATCHERS=2)"
```

すると、2 つのディスパッチャが、HOST=144.25.16.201 でリスニングしますが、それらは、ディスパッチャからアクセス可能なカードでなければなりません。

例 2 ディスパッチャの正確な位置を知るためには、次のように PORT を追加します。

```
MTS_DISPATCHERS="(ADDRESS=(PARTIAL=TRUE)(PROTOCOL=TCP)¥  
  (HOST=144.25.16.201)(PORT=5000))(DISPATCHERS=1)"  
MTS_DISPATCHERS="(ADDRESS=(PARTIAL=TRUE)(PROTOCOL=TCP)¥  
  (HOST=144.25.16.201)(PORT=5001))(DISPATCHERS=1)"
```

注意: 複数の MTS_DISPATCHERS を INIT.ORA ファイルに指定できますが、それらは互いに隣接していなければなりません。

MTS_MAX_DISPATCHERS: ディスパッチャの最大数を設定する

パラメータ MTS_MAX_DISPATCHERS は、インスタンスの存続中に起動できるディスパッチャ・プロセスの最大数（すべてのネットワーク・プロトコルの合計）を設定します。

必要なだけディスパッチャ・プロセスを作成できますが、ディスパッチャを含むプロセスの総数が、実行プロセスの数に対するホスト・オペレーティング・システムの制限を超えることはできません。

ディスパッチャの最大数を見積もる

インスタンスが必要とするディスパッチャ・プロセスの最大数を見積もるには、次の公式を使ってください。

$$\text{MTS_MAX_DISPATCHERS} = \frac{\text{maximum number of concurrent sessions}}{\text{connections per dispatcher}}$$

MTS_SERVERS: 共有サーバー・プロセスの初期数を設定する

インスタンス起動時には、パラメータ MTS_SERVERS によって決定されている数の共有サーバー・プロセスが起動されます。データベース・システムごとの初期の共有サーバー・プロセスの適切な数は、通常の接続ユーザー数と各ユーザーが必要とする処理量によって決まります。ある一定期間にわたり、各ユーザーが比較的わずかな要求しかしないのであれば、対応付けられた各ユーザー・プロセスは、時間の大部分はアイドル状態です。その場合、1 つの共有サーバー・プロセスは、10 人から 20 人のユーザーを処理できます。各ユーザーが著しい量の処理を必要とする場合、ユーザー・プロセス対サーバー・プロセスのより高い比率が要求を処理するために必要とされます。

Oracle で共有サーバーを使う場合は、MTS_SERVERS を必ず 1 以上に設定してください。パラメータを指定しない場合、または 0 に設定する場合、Oracle は共有サーバーを起動しません。ただし、インスタンスの稼働中に、MTS_SERVERS を 0 より大きい値にあとで設定できます。

初期の共有サーバー・プロセスは、少なめに見積もるのがよいでしょう。追加の共有サーバーは、必要に応じて自動的に起動され、長時間にわたってアイドル状態のままである場合は、自動的に割当てが解除されます。しかし、アイドル状態であっても、初期サーバーは常に割り当てられたままになります。初期サーバーの数を大きく設定すると、不要なオーバーヘッドを招く可能性があります。通常のデータベース・アクティビティに対して理想的なシ

ステム・パフォーマンスを見つけるまで、初期共有サーバー・プロセスの数について実験し、共有サーバーを監視してください。

関連項目：共有サーバーの数の変更方法の詳細は、4-9 ページの「共有サーバー・プロセスの最小数を変更する」を参照してください。

MTS_MAX_SERVERS: 共有サーバー・プロセスの最大数を設定する

インスタンスの持続時間に対して起動する共有サーバー・プロセスの最大数は、初期化パラメータ MTS_MAX_SERVERS によってインスタンス起動時に決定されます。一般に、最高のアクティビティ時の共有サーバー・プロセスの適切な数を許可するように、このパラメータを設定します。このパラメータに対する理想的な設定を決定するために、この制限について実験し、共有サーバーを監視してください。

サーバー・プロセスの変更

ここでは、インスタンスの起動後にできる変更について説明します。トピックは次のとおりです。

- 共有サーバー・プロセスの最小数を変更する
- ディスパッチャ・プロセスを追加、削除する

共有サーバー・プロセスの最小数を変更する

インスタンスを起動した後は、SQL コマンド ALTER SYSTEM を使って共有サーバー・プロセスの最小数を変更できます。

Oracle は最終的に、指定した最小値より長い時間アイドル状態のディスパッチャとサーバーを終了させます。

MTS_SERVERS を 0 に設定した場合、Oracle は現行のサーバーすべてがアイドル状態になると、これらのサーバーをすべて終了させ、MTS_SERVERS の値を大きくしない限り新しいサーバーを起動しません。つまり、MTS_SERVERS を 0 に設定することによって、マルチスレッド・サーバーが一時的に使用禁止になります。

共有サーバー・プロセスの最小数を制御するには、ALTER SYSTEM 権限を持っていなければなりません。

次の文は、共有サーバー・プロセスの数を 2 に設定します。

```
ALTER SYSTEM SET MTS_SERVERS = 2
```

ディスパッチャ・プロセスを追加、削除する

インスタンスの中のディスパッチャ・プロセスの数を制御できます。ディスパッチャ・プロセスに対する負荷が一貫して高いことをビュー V\$QUEUE と V\$DISPATCHER が示している場合、待ち時間なしでユーザー要求をルーティングできるように追加のディスパッチャ・プロセスを起動してください。ディスパッチャの数が MTS_MAX_DISPATCHERS に等しくな

るまで、新しいディスパッチャを起動できます。逆に、ディスパッチャの負荷が一貫して低い場合はディスパッチャの数を少なくしてください。

ディスパッチャ・プロセスの数を変更するには、SQL コマンド ALTER SYSTEM を使ってください。特定のプロトコルに対するディスパッチャ数を変更しても、他のプロトコルのディスパッチャに影響は及びません。

MTS_LISTENER_ADDRESS パラメータと MTS_DISPATCHERS パラメータに指定したプロトコルに対して、ディスパッチャ・プロセスを新たに起動できます。したがって、ディスパッチャが存在するプロトコルに限りディスパッチャを追加できます。現在ディスパッチャが存在していないプロトコルに対してディスパッチャを起動するには、データベースを停止し、パラメータ・ファイルを変更してから、データベースを再起動してください。

特定のプロトコルに対するディスパッチャ数を少なくしても、ディスパッチャはただちには削除されません。Oracle は最終的に長時間アイドル状態のディスパッチャを終了させ、MTS_DISPATCHERS で指定した制限数にします。

ディスパッチャ・プロセスの数を制御するには、ALTER SYSTEM 権限を持っていないければなりません。

次の例は、ディスパッチャの数が 3 のときにディスパッチャ・プロセスを追加しています。

```
ALTER SYSTEM  
SET MTS_DISPATCHERS = '(PROTOCOL=TCP) (DISPATCHER=4)';
```

関連項目： マルチスレッド・サーバー・チューニングの詳細は、『Oracle8 Server チューニング』を参照してください。

Oracle プロセスを追跡

Oracle のインスタンスでは、多数のバックグラウンド・プロセスを実行できます。できれば、これらのプロセスを追跡するようにしてください。ここではこれらのプロセスの追跡方法を説明します。トピックは次のとおりです。

- Oracle インスタンスのプロセスを監視する
- トレース・ファイル、ALERT ファイル、バックグラウンド・プロセス
- チェックポイント・プロセスを起動する

関連項目： Oracle プロセスのチューニングの詳細は『Oracle8 Server チューニング』を参照してください。

Oracle インスタンスのプロセスを監視する

モニターを使って、データベース・アクティビティおよびリソースの使用を追跡できます。Enterprise Manager/GUI の監視機能を選択すると、Oracle のデータベースのプロセスに関する

る現行情報が表示されます。複数のモニターを同時に動作できます。表 4-1 に、Oracle のプロセスの追跡に役立つ Enterprise Manager モニターを示します。

表 4-1 Enterprise Manager モニター

モニター名	説明
プロセス	「プロセス・モニター」は、クライアント / サーバー・プロセス、ユーザー・プロセス、サーバー・プロセス、バックグラウンド・プロセスを含み、現行のデータベース・インスタンスを介して、その時点でデータベースにアクセスしているすべての Oracle プロセスに関する情報を要約します。
セッション	「セッション・モニター」は、接続されている各 Oracle セッションのセッション ID とその状態を示します。

ロックを監視する

表 4-2 に、インスタンス内で進行中のトランザクションについてのロック情報を監視する 2 つの方法を示します。

表 4-2 Oracle の監視機能

モニター名	説明
Enterprise Manager モニター	Enterprise Manager/GUI のモニター機能では、インスタンスのロック情報を表示する 2 つのモニター（ロック・モニターとラッチ・モニター）を提供します。
UTLLOCKT.SQL	UTLLOCKT.SQL スクリプトは、ツリー構造方式で簡単なロック待機グラフを表示します。スクリプトは、Enterprise Manager や SQL*Plus などの非定型の問合せツールを使って、ロックとそれに対応するブロッキング・ロックを、待機中のシステム内のセッションに出力します。このスクリプト・ファイルの位置は、オペレーティング・システムによって違います。オペレーティング・システム固有の Oracle のマニュアルを参照してください。（スクリプト CATBLOCK.SQL は UTLLOCKT.SQL が必要とするロック・ビューを作るため、UTLLOCKT.SQL を実行する前に実行しなければなりません）。

動的パフォーマンス表を監視する

次のビューは、動的パフォーマンス表に対して作られるものであり、Oracle インスタンス・プロセスの監視に役立ちます。

ビュー（モニター）名	説明
V\$CIRCUIT	ディスパッチャおよびサーバーを通したユーザー接続である、仮想回路についての情報が含まれています。
V\$QUEUE	マルチスレッド・メッセージ待ち行列についての情報が含まれています。
V\$DISPATCHER	ディスパッチャ・プロセスについての情報が含まれています。
V\$SHARED_SERVER	共有サーバー・プロセスについての情報が含まれています。
V\$SQLAREA	共有 SQL 領域に関する統計表示が含まれており、SQL 文字列ごとに 1 行になっています。また、メモリー内にあり、解析済みで、実行準備のできている SQL 文に関する統計表示も提供します。
V\$SESS_IO	各ユーザー・セッションの I/O 統計表示が含まれています。
V\$LATCH	非親ラッチの統計表示と、親ラッチのサマリー統計表示が含まれています。
V\$SYSSTAT	システム統計表示が含まれています。

次に示すのは、動的パフォーマンス表の 1 つ、V\$DISPATCHER の一般的な問合せです。出力には、システム内の各ディスパッチャに対する処理負荷が表示されます。

```
SELECT (busy/(busy + idle)) * 100 "% OF TIME BUSY"
FROM v$dispatcher;
```

オペレーティング・システム・バックグラウンド・プロセスから Oracle バックグラウンド・プロセスを識別する

1 台のコンピュータ上で多くの Oracle データベースを同時に実行する場合に、Oracle にはインスタンスのプロセスを命名するメカニズムがあります。バックグラウンド・プロセス名には、インスタンス識別子によって接頭辞が付けられ、各インスタンスに対するプロセスの集合を区別します。

たとえば、TEST というインスタンスには、次の名前のバックグラウンド・プロセスが存在します。

- ORA_TEST_DBWR
- ORA_TEST_LGWR
- ORA_TEST_SMON
- ORA_TEST_PMON
- ORA_TEST_RECO
- ORA_TEST_LCK0
- ORA_TEST_ARCH
- ORA_TEST_D000
- ORA_TEST_S000
- ORA_TEST_S001

関連項目：ビューと動的パフォーマンス表の詳細は、『Oracle8 Server リファレンス・マニュアル』を参照してください。

インスタンス識別子および Oracle のプロセス名の書式については、使用しているオペレーティング・システム固有の Oracle のマニュアルを参照してください。

トレース・ファイル、ALERT ファイル、バックグラウンド・プロセス

各サーバー・プロセスとバックグラウンド・プロセスは、対応付けられたトレース・ファイルに書き込むことができます。内部エラーがプロセスによって検出されると、エラーに関する情報がトレース・ファイルにダンプされます。トレース・ファイルに書き込まれる情報のいくつかは、データベース管理者のためのものであり、その他の情報は弊社技術サポートのためのものです。また、トレース・ファイルの情報は、アプリケーションとインスタンスの調整でも使えます。

ALERT ファイルは特殊なトレース・ファイルです。データベースの ALERT ファイルが、次のようなものを含む、メッセージとエラーの履歴ログです。

- 発生したすべての内部エラー (ORA-600)、ブロック障害エラー (ORA-1578)、デッドロック・エラー (ORA-60)
- 管理オペレーション、たとえば、CREATE/ALTER/DROP DATABASE/TABLESPACE/ROLLBACK SEGMENT の各 SQL 文と STARTUP、SHUTDOWN、ARCHIVE LOG、RECOVER の各 Enterprise Manager 文
- 共有サーバーとディスパッチャ・プロセスの機能に関するいくつかのメッセージとエラー
- スナップショットの自動リフレッシュ中に発生したエラー
- データベースとインスタンスの起動時のすべての初期化パラメータの値

Oracle は、オペレータ・コンソール上にこのような情報を表示するかわりに（ただし、多くのシステムがコンソール上に情報を表示する）これらの特別な操作のログを保持するために、ALERT ファイルを使います。操作が成功すると、タイムスタンプとともに「completed（完了）」というメッセージが、ALERT ファイルに書き込まれます。

トレース・ファイルを使う

ALERT ファイルとインスタンスのその他のトレース・ファイルを定期的にチェックして、バックグラウンド・プロセスがエラーを発見したかどうかを確認できます。たとえば、ログ・ライター・プロセス (LGWR) がグループのメンバーに書き込むことができないと、LGWR トレース・ファイルとデータベースの ALERT ファイルにエラー・メッセージが書き込まれます。このようなエラー・メッセージが見つかった場合、ただちに解決すべきメディアまたは I/O の問題が発生しています。

他の重要な統計に加えて初期化パラメータの値も ALERT ファイルに書き込まれます。たとえば、通常どおりに、または即時に（ただし、中断ではない）インスタンスを停止すると、Oracle7 は、インスタンスが起動してから、インスタンスに同時に接続されたセッションの最高数を ALERT ファイルに書き込みます。この数を使って、Oracle セッション・ライセンスをアップグレードする必要があるかどうかを確認できます。

トレース・ファイルの位置を指定する

バックグラウンド・プロセスに対するすべてのトレース・ファイルと ALERT ファイルは、初期化パラメータ BACKGROUND_DUMP_DEST によって指定された宛先に書き込まれます。サーバー・プロセスに対するすべてのトレース・ファイルは、初期化パラメータ USER_DUMP_DEST によって指定された宛先に書き込まれます。トレース・ファイルの名前はオペレーティング・システムによって異なりますが、通常ファイルを書き込んでいるプロセスの名前が含まれます (LGWR、RECO など)。

トレース・ファイルのサイズを制御する

すべてのトレース・ファイル (ALERT ファイルを除く) の最大サイズは、初期化パラメータ MAX_DUMP_FILE_SIZE を使って制御できます。この制限は、いくつかのオペレーティング・システム・ブロックとして設定されます。ALERT ファイルのサイズを制御するために、そのファイルが必要ではなくなったときに、手動で削除しなければなりません。そうしないと、Oracle はファイルを追加し続けます。ただし、最初に ALERT ファイルのアーカイブ・コピーを作っておくとよいでしょう。

Oracle がトレース・ファイルに書き込む時期を制御する

バックグラウンド・プロセスは適宜、トレース・ファイルに情報を書き込みます。ただし、初期化パラメータ SQL_TRACE が TRUE に設定されている場合だけ、トレース・ファイルはサーバー・プロセスのために書き込まれます（内部エラー中にも書き込まれます）。

SQL_TRACE の現行値にかかわらず、各セッションは、SET SQL_TRACE パラメータを指定して SQL コマンド ALTER SESSION を使うことによって、対応付けられたサーバー・プロセスにかわってトレース・ロギングを使用可能にしたり、使用禁止にしたりできます。

```
ALTER SESSION SET SQL_TRACE TRUE;
```

マルチスレッド・サーバーでは、ディスパッチャを使っている各セッションは、共有サーバー・プロセスへのルートが指定されています、またそのセッションがトレースを使用可能にしている場合（または、エラーが見つかった場合）に限り、トレース情報がサーバーのトレース・ファイルに書き込まれます。ディスパッチャを使って接続する特定のセッションに対するトレースを突き止めるには、いくつかの共有サーバーのトレース・ファイルを調べる必要があるかもしれません。サーバー・プロセスの SQL トレース機能は著しいシステム・オーバーヘッドを引き起こす可能性があるので、統計を収集するときだけこの機能を使用可能にしてください。

関連項目：ライセンスの更新の詳細は、20-2 ページの「セッションとユーザーのライセンス」を参照してください。

メッセージの詳細は『Oracle8 Server エラー・メッセージ』を参照してください。

トレース・ファイルの名前の詳細は、使っているオペレーティング・システム固有の Oracle のマニュアルを参照してください。

ALTER SESSION コマンドの詳細は、『Oracle8 Server SQL リファレンス』を参照してください。

チェックポイント・プロセスを起動する

チェックポイント・プロセス（CKPT）ができない場合には、ログ・ライター・プロセス（LGWR）が、最新のチェックポイントを反映するために、すべての制御ファイルとデータ・ファイルのヘッダーを更新しなければなりません。チェックポイントを完了するために必要な時間を節減するには（特にデータベースが多くのデータ・ファイルから構成されているとき）CKPT バックグラウンド・プロセスを使用可能にしてください。（デフォルト値は FALSE です。）

並列問合せオプションのプロセスの管理

ここでは、並列問合せオプションによって Oracle がどのように並列処理を実行するかを説明します。この構成では、Oracle で特定のタイプの SQL 文の処理作業を複数の問合せサーバー・プロセスの間で配分できます。トピックは次のとおりです。

- 問合せサーバーを管理する
- 問合せサーバー・プロセスの数の変化

関連項目：パラレル問合せオプションの詳細は、『Oracle8 Server チューニング』を参照してください。

問合せサーバーを管理する

インスタンスを起動すると、Oracle Server はどの問合せコーディネータでも使える問合せサーバー・プロセスをプールします。Oracle がインスタンスの起動時に作る問合せサーバー・プロセスの数は、初期化パラメータ `PARALLEL_MIN_SERVERS` で指定します。

問合せサーバー・プロセスは、その実行フェーズを通して 1 つの文に対応付けられた状態になっています。文が完全に処理されると、その文の問合せサーバー・プロセスが他の文を処理できるようになります。問合せコーディネータ・プロセスは、結果として生成されるデータを、その文を発行したユーザー・プロセスに戻します。

問合せサーバー・プロセスの数の変化

インスタンスによって同時に処理される SQL 文のボリュームの変化が非常に大きい場合は、Oracle Server がこのボリュームに対応するようにプール中の問合せサーバー・プロセスの数を自動的に変更します。

このボリュームが増えると、Oracle Server は自動的に問合せサーバー・プロセスを追加作成して、入力文を処理します。インスタンス用の問合せサーバー・プロセスの最大数は、初期化パラメータ `PARALLEL_MAX_SERVERS` で指定します。

後でこのボリュームが減ると、Oracle Server は初期化パラメータ `PARALLEL_SERVER_IDLE_TIME` で指定した時間アイドル状態になった場合、問合せサーバー・プロセスを終了させます。問合せサーバー・プロセスがアイドル状態となっていた時間に関係なく、Oracle Server がプールのサイズを `PARALLEL_MIN_SERVERS` の値より小さくすることはありません。

プール中のすべての問合せサーバーが使われ、最大数の問合せサーバーが起動されると、結果として、問合せコーディネータ・プロセスが文を処理します。

関連項目：インスタンスの問合せサーバーのプールの監視方法と、初期化パラメータの適切な値を決める方法の詳細は、『Oracle8 Server チューニング』を参照してください。

外部プロシージャのプロセスの管理

C 関数の共有ライブラリを Oracle データベースからコールしたい場合があるかもしれません。ここでは、それらの外部プロシージャをコールするための環境を設定する方法について説明します。トピックは次のとおりです。

注意： 必須ではありませんが、これらの作業はできるだけインストール時に行ってください。

データベース管理者が適切なライブラリについての実行権限をアプリケーションの開発者に付与し、そのアプリケーションの開発者が外部プロシージャを作成し、特定の外部プロシージャについての実行権限を他のユーザーに付与します。

外部プロシージャをコールするための環境を設定する手順

1. `tnsnames.ora` ファイルを編集して、リスナー・プロセス（およびその後 `EXTPROC` プロセス）に接続するための項目を追加する。
2. `listener.ora` ファイルを編集して、「外部プロシージャ・リスナー」のための項目を追加する。
3. 外部プロシージャを排他的に処理するための別個のリスナー・プロセスを開始する。
4. リスナーによって生成された `EXTPROC` プロセスはリスナーのオペレーティング・システム権限を継承するため、別個のリスナー・プロセスの権限は必ず限定されたものにしてください。そのプロセスに対して、データベース・ファイルまたは Oracle Server のアドレス・スペースへの読み込みまたは書き込みの許可を与えないようにしてください。

さらに、この別個のリスナー・プロセスの所有者は "oracle"（サーバーの実行可能ファイルおよびデータベース・ファイルのデフォルト所有者）であってはなりません。

5. まだインストールされていない場合、`extproc` 実行可能ファイルを `$ORACLE_HOME/bin` に入れる。

`tnsnames.ora` 内の項目のサンプル

`tnsnames.ora` に記述する、外部プロシージャ・リスナーのための項目のサンプルを次に示します。

```
extproc_connection_data = (DESCRIPTION =
                           (ADDRESS = (PROTOCOL=IPC)
                                (KEY=extproc_key)
                           )
                           (CONNECT_DATA = (SID = extproc_agent)
                           )
                           )
```

この例およびすべての外部プロシージャのコールアウトで、項目名 `extproc_connection_data` は変更できません。ここに示されているとおりに入力してください。指定するキー名（この

場合は **extproc_key**) は、listener.ora ファイルの KEY と一致していなければなりません。さらに、指定する SID 名 (この場合は **extproc_agent**) は、listener.ora ファイルの SID_NAME 項目と一致していなければなりません。

listener.ora 内の項目のサンプル

listener.ora に記述する、外部プロシージャのための項目のサンプルを次に示します。

```
EXTERNAL_PROCEDURE_LISTENER =

(ADDRESS_LIST =
  (ADDRESS = (PROTOCOL=ipc)
              (KEY=extproc_key)
            )
)
...
SID_LIST_EXTERNAL_PROCEDURE_LISTENER =

(SID_LIST =
  (SID_DESC = (SID_NAME=extproc_agent)
              (ORACLE_HOME=/oracle)
              (PROGRAM=extproc)
            )
)
```

この例で、プログラム名は **extproc** でなければならず、また変更できません。ここに示されているとおりに入力してください。SID_NAME は、tnsnames.ora ファイル内の名前と一致していなければなりません。ORACLE_HOME は、Oracle ソフトウェアがインストールされるディレクトリを設定しなければなりません。実行可能ファイル **extproc** は、\$ORACLE_HOME/bin になければなりません。

関連項目 : 外部プロシージャの詳細は、『PL/SQL ユーザーズ・ガイドおよびリファレンス』を参照してください。

セッションの停止

状況によっては、現行のユーザー・セッションを停止 (KILL) したい場合があります。たとえば、管理操作を実行する必要がある、管理に関係のないセッションをすべて終了しなければならない場合などです。

ここでは、セッションの停止について説明します。トピックは次のとおりです。

- 停止するセッションを識別する
- アクティブなセッションを停止する
- アクティブでないセッションを停止する

セッションが停止すると、そのセッションのトランザクションがロールバックされ、そのセッションが保持していたリリース（ロックやメモリー領域など）がただちに解除され、他のセッションで使えます。

現行のセッションを停止するには、Enterprise Manager の「セッション切断」メニュー項目または SQL コマンド ALTER SYSTEM...KILL SESSION を使います。

次の文は、SID が 7 でシリアル番号が 15 のセッションを停止します。

```
ALTER SYSTEM KILL SESSION '7,15';
```

停止するセッションを識別する

停止するセッションを識別するために、セッションの索引番号とシリアル番号を指定してください。セッションの索引（SID）とシリアル番号を識別するために、V\$SESSION 動的パフォーマンス表を問い合わせてください。

次の問合せは、ユーザー JWARD のすべてのセッションを識別します。

```
SELECT sid, serial#
FROM v$session
WHERE username = 'JWARD';
```

SID	SERIAL#	STATUS
7	15	ACTIVE
12	63	INACTIVE

セッションは、Oracle に対して SQL コールを実行しているときは ACTIVE です。セッションは、Oracle に対して SQL コールを実行していないときは INACTIVE です。

関連項目：セッションの状態値の詳細は、『Oracle8 Server チューニング』を参照してください。

アクティブなセッションを停止する

停止時にユーザー・セッションが Oracle へ SQL コールを実行している（ACTIVE である）場合、トランザクションはロールバックされ、ユーザーはただちに次のメッセージを受け取ります。

ORA-00028: セッションは強制終了されました。

ユーザーが、ORA-00028 のメッセージを受け取った後に、データベースに再接続する前に追加の文を実行すると、Oracle は次のメッセージを戻します。

ORA-01012: ログオンされていません。

アクティブ・セッションが中断できない（たとえば、ネットワーク I/O、トランザクションのロールバックを実行する）場合、その操作が完了するまで、そのセッションは停止できません。この場合、停止するまで、セッションはすべてのリソースを保持します。さらに、セッションを停止するために ALTER SYSTEM 文を発行するセッションは、そのセッション

が停止されるのを 60 秒まで待機します。中断できなかった操作が 1 分間続くと、ALTER SYSTEM 文の発行者は、セッションが停止したことを示す「マークを付けられた」ことを伝えるメッセージを受け取ります。そのようなマーク付きのセッションでは、V\$SESSION の状態 (STATUS) に KILLED が表示され、サーバー (SERVER) には「PSEUDO」以外の値が表示されます。

アクティブでないセッションを停止する

セッションが停止時に Oracle に対して SQL コールを実行していない (INACTIVE である) 場合、ORA-00028 メッセージはただちには戻されません。このメッセージは、ユーザーが後で停止されたセッションを使おうとするとときまで戻されません。

アクティブでないセッションを停止すると、V\$SESSION ビューの STATUS が KILLED になります。停止されたセッションの行は、ユーザーが再びそのセッションを使おうとすると V\$SESSION から削除され、ORA-00028 メッセージが表示されます。

次の例では、データベース管理者がアクティブでないセッションを停止する場合を示します。

```
SVRMGR> SELECT sid,serial#,status,server
2> FROM v$session
3> WHERE username = 'JWARD';

SID          SERIAL#  STATUS  SERVER
-----
          7      15  INACTIVE  DEDICATED
          12     63  INACTIVE  DEDICATED
2 rows selected.

SVRMGR> ALTER SYSTEM KILL SESSION '7,15';
Statement processed.

SVRMGR> SELECT sid, serial#, status, server
2> FROM v$session
3> WHERE username = 'JWARD';

SID          SERIAL#  STATUS  SERVER
-----
          7      15   KILLED  PSEUDO
          12     63  INACTIVE  DEDICATED
2 rows selected.<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2//EN">
```

オンライン REDO ログの管理

この章では、オンライン REDO ログを管理する方法について説明します。トピックは次のとおりです。

- オンライン REDO ログの計画
- オンライン REDO ログ・グループおよびメンバーの作成
- オンライン REDO ログ・メンバーを改名および再配置
- オンライン REDO ログ・グループの削除
- オンライン REDO ログ・メンバーの削除
- チェックポイントとログ・スイッチの制御
- REDO ログ・ファイル内のブロックの検証
- オンライン REDO ログ・ファイルの消去
- オンライン REDO ログ・ファイルに関する情報のリスト

関連項目 : Oracle Parallel Server を使っているときにインスタンスのオンライン REDO ログを管理する方法については、『Oracle8 Parallel Server 概要および管理』を参照してください。

REDO ログのアーカイブ方法の詳細は、第 23 章「REDO 情報のアーカイブ」を参照してください。

この章には、Oracle Enterprise Manager について述べている箇所がいくつかあります。Enterprise Manager/GUI または Server Manager/LineMode を使って特定のタスクを実行する方法の詳細は、『Oracle Enterprise Manager 管理者ガイド』および『Oracle Server Manager ユーザーズ・ガイド』を参照してください。

オンライン REDO ログの計画

Oracle データベースのインスタンスにはすべて、対応付けられた「オンライン REDO ログ」があります。そのオンライン REDO ログは、2 つ以上のオンライン・ログ・ファイル（コミットした変更をすべてデータベースに記録している）で構成されています。オンライン REDO ログは、インスタンス障害が発生した場合にデータベースを保護するのに役立ちます。トランザクションがコミットされると、一時的にシステム・グローバル領域の REDO ログ・バッファに格納されていた対応する REDO エントリが、バックグラウンド・プロセス LGWR によってオンライン REDO ログ・ファイルに書き込まれます。

オンライン REDO ログ・ファイルは、循環して使われます。たとえば 2 つのファイルがオンライン REDO ログを構成している場合、最初のファイルが満杯になり 2 番目のファイルも満杯になると、最初のファイルが再び使われ、それが満杯になると、2 番目のファイルが再び使われる、という具合です。ファイルが満杯になるたびに、「ログ順序番号」がそのファイルに割り当てられ、REDO エントリのセットを識別します。

ここでは、データベース・インスタンスのオンライン REDO ログを構成するときに考慮すべきガイドラインを説明します。トピックは次のとおりです。

- オンライン REDO ログを多重化する
- オンライン REDO ログ・メンバーを別々のディスクに配置する
- オンライン REDO ログ・メンバーのサイズを設定する
- 適切な数のオンライン REDO ログ・ファイルを選択する

オンライン REDO ログを多重化する

データベース・インスタンスのオンライン REDO ログは、多重化したオンライン REDO ログ・ファイルのグループで構成してください。さらに、単一ディスク障害が LGWR やデータベース・インスタンスの障害を引き起こさないように、同一グループ内のメンバーを別々のディスク上に格納しなければなりません。

1 箇所の障害のためにデータベースが損失するのを避けるため、Oracle では オンライン REDO ログファイルの複数のセットを保持できます。「多重化オンライン REDO ログ」は、物理的に別々のディスク上にあるオンライン REDO ログ・ファイルのコピーから構成され、グループのあるメンバーに対してなされた変更はすべてのメンバーに対して実行されます。オンライン REDO ログ・ファイルが入っているディスクに障害が発生しても、その他のコピーは影響を受けず、Oracle で使用可能です。システム操作は中断されず、失われたオンライン REDO ログ・ファイルは容易に回復できます。

警告： Oracle8 Server では、多重化されたグループがそれぞれ異なる数のメンバーを持つことができますが、これは、グループのメンバーの損傷を与えるような異常な状況（ディスク障害など）が発生した場合に、一時的に許されるだけです。どのグループにもメンバーが 1 つしかない場合は、そのメンバーが格納されているディスクに障害が起きると Oracle は停止します。

グループを多重化すると余分な記憶領域が必要となりますが、(ディスク障害によって多重化されていないオンライン REDO ログが破棄された場合に) 失われるデータのコストと比較すると、この領域のコストは通常それほど大きな問題ではありません。

オンライン REDO ログ・メンバーを別々のディスクに配置する

多重オンライン REDO ログを設定する場合、グループのメンバーを異なるディスク上に配置します。このようにすれば、1つのディスクで障害が発生しても、LGWR が使えなくなるのはグループの1つのメンバーだけで、それ以外のメンバーは引き続き LGWR にアクセスでき、インスタンスは機能し続けることができます。

REDO ログをアーカイブする場合には、バックグラウンド・プロセス LGWR と ARCH 間の競合をなくすためにオンライン REDO ログ・メンバーを複数のディスクに配置してください。たとえば、ミラー化したオンライン REDO ログ・メンバーのグループが2つある場合、各メンバーを異なるディスク上に配置し、アーカイブ先を5つ目のディスクに設定します。このようにすれば、LGWR (メンバーへの書き込み) と ARCH (メンバーの読み込み) 間の競合は起こりません。

データ・ブロックや REDO エントリの書き込み時の競合を減らすため、データ・ファイルとオンライン REDO ログ・ファイルも、異なるディスク上に配置する必要があります。

オンライン REDO ログ・メンバーのサイズを設定する

オンライン REDO ログ・ファイルのサイズを設定するときには、REDO ログをアーカイブするかどうかを考慮してください。オンライン REDO ログ・ファイルのサイズは、満杯になったグループを1つのオフライン記憶メディア (テープやディスクなど) にアーカイブできるとともに、そのメディア上の未使用領域が最小になるように設定してください。たとえば、テープあたり満杯のオンライン REDO ログ・グループを1つだけアーカイブし、テープの記憶容量の49% が未使用のまま残っているとします。この場合には、オンライン REDO ログ・ファイルのサイズを小さくして、テープに2つのオンライン REDO ログ・グループがアーカイブされるように構成するとよいでしょう。

オンライン REDO ログの多重グループでは、同一グループのメンバーはすべて同じサイズでなければなりません。異なるグループのメンバーは異なるサイズを持つことができます。ただし、グループ間でファイル・サイズを変えても利点はありません。ログ・スイッチ間にチェックポイントが発生するように設定していない場合には、グループのサイズを同一に設定して、チェックポイントが一定の間隔で発生することを保証してください。

関連項目：オンライン REDO ログ・ファイルのデフォルトのサイズはオペレーティング・システムによって異なります。詳細は、使っているオペレーティング・システム固有の Oracle のマニュアルを参照してください。

適切な数のオンライン REDO ログ・ファイルを選択する

データベース・インスタンスに対するオンライン REDO ログ・ファイルの適切な数を決定する最良の方法は、異なる構成をテストすることです。最適の構成では、グループ数は LGWR が REDO ログ情報を書き込むのを妨げない最小の値です。

データベース・インスタンスに必要なグループが 2 つだけの場合もあります。また、LGWR が使える再生グループが常に存在するようにするため、データベース・インスタンスにグループを追加しなければならない場合もあります。テスト中に、現在のオンライン REDO ログ構成に問題がないかどうかを確認するには、LGWR トレース・ファイルおよびデータベースの ALERT ファイルの内容を調べるのが最も簡単です。チェックポイントが完了されていなかったり、グループがアーカイブされていないために、LGWR が頻繁にグループを待つ必要があるというメッセージが示された場合は、グループを追加してください。

インスタンスのオンライン REDO ログ構成を設定または変更する前に、オンライン REDO ログ・ファイル数を制限するパラメータを検討してください。次の 3 つのパラメータは、データベースに追加できるオンライン REDO ログ・ファイルの数を制限します。

- CREATE DATABASE 文の MAXLOGFILES パラメータ。これは、データベースあたりのオンライン REDO ログ・ファイルの最大グループ数を決めるものです。グループの値は 1 ~ MAXLOGFILES です。この上限値を置き換える唯一の方法は、データベースまたは制御ファイルを再作成することです。したがって、データベースを作る前にこの上限値を検討することが重要です。CREATE DATABASE 文に MAXLOGFILES パラメータが指定されていない場合には、Oracle はオペレーティング・システムのデフォルト値を使います。
- LOG_FILES パラメータ (パラメータ・ファイルの中)。これを使うと、現行インスタンスの存続中、オンライン REDO ログ・ファイルのグループの最大数を一時的に小さくできます。ただし、LOG_FILES パラメータを使って MAXLOGFILES を書き換えて最大数を大きくすることはできません。データベースのパラメータ・ファイルに LOG_FILES パラメータが指定されていない場合、Oracle はオペレーティング・システムのデフォルト値を使います。
- CREATE DATABASE 文の MAXLOGMEMBERS パラメータ。これはグループに含まれるメンバーの最大値を決めるものです。MAXLOGFILES と同じように、この上限値を置き換える唯一の方法は、データベースまたは制御ファイルを再作成することです。したがって、データベースを作る前にこの上限値を検討することが重要です。CREATE DATABASE 文に MAXLOGMEMBERS パラメータが指定されていない場合には、Oracle はオペレーティング・システムのデフォルト値を使います。

関連項目： MAXLOGFILES パラメータと MAXLOGMEMBERS パラメータ、および LOG_FILES 初期化パラメータのデフォルト値と有効な値については、使用しているオペレーティング・システム固有の Oracle のマニュアルを参照してください。

オンライン REDO ログ・グループおよびメンバーの作成

オンライン REDO ログ・ファイルのグループとメンバーは、データベースを作るときと作った後の両方で作成できます。できれば、データベースのオンライン REDO ログを事前に計画し、データベースを作るときに必要なオンライン REDO ログ・ファイルのグループとメンバーをすべて作ってください。新たにオンライン REDO ログ・グループおよびメンバーを作るには、ALTER DATABASE システム権限を持っていなければなりません。

場合によっては、オンライン REDO ログ・ファイルのグループまたはメンバーを追加で作らなければならないことがあります。たとえば、オンライン REDO ログにグループを追加することによって、REDO ログ・グループの可用性の問題を解決できます。データベースは、MAXLOGFILES グループまで持つことができます。

オンライン REDO ログ・グループを作る

オンライン REDO ログ・ファイルのグループを新たに作るには、Enterprise Manager の「ログ・ファイル・グループの追加」プロパティ・シートを使うか、または SQL コマンド ALTER DATABASE に ADD LOGFILE パラメータを指定して使います。

次の文は、データベースに新しい REDO ログ・グループを追加します。

```
ALTER DATABASE
  ADD LOGFILE ('log1c', 'log2c') SIZE 500K;
```

注意: 新しいログ・メンバーのファイル名は、オペレーティング・システム・ファイルを作成すべき位置を含め省略しないで指定します。そうしないと、ファイルはオペレーティング・システムによって異なるデータベース・サーバーのデフォルトのディレクトリに作られます。なお、既存のオペレーティング・システム・ファイルを再利用する場合、ファイル・サイズを指定する必要はありません。.

ALTER DATABASE 文で ADD LOGFILE オプションとともに GROUP オプションを使うと、グループを識別する番号を指定できます。

```
ALTER DATABASE
  ADD LOGFILE GROUP 10 ('log1c', 'log2c') SIZE 500K;
```

グループ番号を使うことによって、REDO ログ・グループの管理がより簡単になります。ただし、グループ番号は 1 ~ MAXLOGFILES の数値でなければなりません。REDO ログ・ファイルのグループ番号をスキップする（つまり、グループに 10、20、30、... のように番号を付ける）と、データベースの制御ファイルで不要な領域が消費されてしまいますので、グループ番号はスキップしないでください。

オンライン REDO ログ・メンバーを作る

完全なオンライン REDO ログ・ファイルのグループを作る必要がない場合もあります。つまり、グループはすでに存在しているけれども、(ディスク障害などにより) 1 つ以上のグループのメンバーが削除されたために完全ではない場合などです。このような場合、既存のグループに新しいメンバーを追加できます。

既存のグループに対してオンライン REDO ログ・メンバーを作るには、Enterprise Manager の「ログ・ファイル・メンバー追加」プロパティ・シートを使うか、または SQL コマンド ALTER DATABASE に ADD LOG MEMBER パラメータを指定して使います。

次の文は、REDO ログ・グループ 2 に新しい REDO ログ・メンバーを追加します。

```
ALTER DATABASE
  ADD LOGFILE MEMBER 'log2b' TO GROUP 2;
```

なお、ファイル名は指定しなければなりませんが、サイズを指定する必要はありません。新しいメンバーのサイズは既存グループのメンバーのサイズによって決まります。

ALTER DATABASE コマンドを使うときには、次の例に示すように、TO パラメータにグループの他のメンバーをすべて指定することによって、目的のグループを選択的に識別できます。

```
ALTER DATABASE
  ADD LOGFILE MEMBER 'log2c' TO ('log2a', 'log2b');
```

注意： 新しいログ・メンバーのファイル名は、オペレーティング・システム・ファイルを作成すべき位置を含め省略しないで指定します。そうしないと、ファイルはデータベース・サーバーのデフォルトのディレクトリに作られます。

オンライン REDO ログ・メンバーを改名および再配置

オンライン REDO ログ・メンバーを改名して、その位置を変更できます。たとえば、オンライン REDO ログ・ファイルが現在保存されているディスクを移動しようとしたり、複数のデータ・ファイルやオンライン REDO ログ・ファイルが同一のディスク上に格納されていて、競合を少なくするためにそれらを分離すべき場合に、この手順が必要となります。

オンライン REDO ログ・メンバーを改名するには、ALTER DATABASE システム権限を持っていなければなりません。さらに、ファイルを希望の位置にコピーするためのオペレーティング・システム権限とデータベースをオープンし、バックアップするための権限が必要となることもあります。

オンライン REDO ログ・メンバーを改名する前に、新たなオンライン REDO ログ・ファイルがすでに存在していることを確認してください。

警告： 次を示す手順では、データベースの制御ファイル内の内部ファイル・ポインタが修正されるだけで、オペレーティング・システム・ファイルを物理的に改名したり、作ったりするわけではありません。オペレーティング・システムを使って、既存のオンライン REDO ログ・ファイルを新しい位置にコピーしてください。

オンライン REDO ログ・メンバーを改名する

1. データベースをバックアップする。

オンライン REDO ログ・メンバーの改名や再配置のように、データベースの構造を変更する場合は、その操作の実行時に発生する可能性のある問題に備えて、事前にデータベース（制御ファイルを含む）をバックアップします。

2. オンライン REDO ログ・ファイルを新しい位置にコピーする。

オンライン REDO ログ・メンバーのようなオペレーティング・システム・ファイルは、適切なオペレーティング・システム・コマンドを使ってコピーしなければなりません。ファイルのコピーに関する説明は、使っているオペレーティング・システムのマニュアルを参照してください。

提案： オペレーティング・システムのコマンドを実行すると、既存の Enterprise Manager を終了せずにファイルをコピーできます。Enterprise Manager の HOST コマンドを使ってください。

3. オンライン REDO ログ・メンバーを改名する。

「オンライン REDO ログ・メンバー改名」ダイアログ・ボックス、または RENAME FILE 句を指定した ALTER DATABASE コマンドを使って、データベースで使われるオンライン REDO ログ・ファイルを改名します。

4. 通常の操作を実行するためにデータベースをオープンする。

オンライン REDO ログの変更は、次にデータベースがオープンされたときに有効となります。データベースをオープンするには、現行インスタンスを停止しなければならない場合（データベースが現行インスタンスによって事前にオープンされていた場合）と、現行インスタンスを使ってデータベースをオープンするだけでよい場合があります。

5. 制御ファイルをバックアップする。

今後発生する可能性のある問題に備えて、一連のオンライン REDO ログ・ファイルを改名、再配置した後、ただちにデータベースの制御ファイルをバックアップしてください。

次に示すのは、オンライン REDO ログ・メンバーを改名する例です。ただし、次のことを前提とします。

- インスタンスに対して、データベースは現在マウントされ、クローズされている。
- オンライン REDO ログはミラー化されている。あるグループはメンバー LOG1A と LOG1B から構成され、第 2 のグループはメンバー LOG2A と LOG2B から構成される。LOG1A と LOG2A のファイルはディスク A に格納され、LOG1B と LOG2B はディスク B に格納されている。
- ディスク A に配置されたオンライン REDO ログ・ファイルをディスク C に再配置しなければならない。新しいファイル名は新しい配置に LOG1C と LOG2C として反映する。

ディスク A 上のファイル LOG1A と LOG2A をディスク C 上の新しいファイル LOG1C と LOG2C にコピーします。

```
ALTER DATABASE
  RENAME FILE 'log1a', 'log2a'
  TO 'log1c', 'log2c';
```

オンライン REDO ログ・グループの削除

場合によっては、オンライン REDO ログ・メンバーを含むグループ全体を削除する必要もあります。たとえば、インスタンスのオンライン REDO ログのグループ数を少なくする必要がある場合などです。

オンライン REDO ログ・グループを削除するには、ALTER DATABASE システム権限を持っていなければなりません。

オンライン REDO ログ・グループを削除する前に、次の制限と注意点について検討してください。

- グループ内のメンバーの数にかかわらず、インスタンスには、少なくともオンライン REDO ログ・ファイルのグループが 2 つ必要。(1 つのグループは 1 つ以上のメンバーから構成されます)。
- オンライン REDO ログ・グループは、アクティブ・グループでない場合にだけ削除できる。アクティブ・グループを削除する必要がある場合は、まず、ログ・スイッチを発生させる必要があります (5-12 ページの「ログ・スイッチを強制する」を参照)。
- 削除する前に、オンライン REDO ログ・グループがアーカイブされていることを確認する (アーカイブが使用可能になっている場合)。アーカイブされているかどうかを確認するには、Enterprise Manager の ARCHIVE LOG コマンドに LIST パラメータを指定して実行します。

オンライン REDO ログ・グループを削除するには、Enterprise Manager の「ログ・ファイル・グループの削除」メニュー項目を使うか、または SQL コマンド ALTER DATABASE に DROP LOGFILE 句を指定して使います。

次の文は、REDO ログ・グループ 3 を削除します。

```
ALTER DATABASE DROP LOGFILE GROUP 3;
```

データベースからオンライン REDO ログ・グループを削除しても、オペレーティング・システム・ファイルはディスクから削除されません。そのグループのメンバーがデータベース構造から削除され、データベースに関連する制御ファイルが更新されます。オンライン REDO ログ・グループを削除した後で、この処理が正常に終了したことを確認し、オペレーティング・システム・コマンドを使って、削除したオンライン REDO ログ・ファイルを実際に削除してください。

オンライン REDO ログ・メンバーの削除

場合によっては、1 つ以上の特定のオンライン REDO ログ・メンバーを削除する必要があります。たとえば、ディスク障害が発生した場合、アクセスできないファイルに書き込みがなされないように、障害のあったディスク上のオンライン REDO ログ・ファイルをすべて削除する必要があります。これ以外にも、適切ではない位置にファイルを格納した場合など、特定のオンライン REDO ログ・ファイルが不要になることがあります。

オンライン REDO ログ・メンバーを削除するには、ALTER DATABASE システム権限を持つていなければなりません。

各オンライン REDO ログ・メンバーを削除する前に、次の制限と注意点について検討してください。

- オンライン REDO ログ・ファイルを削除することにより、多重化したオンライン REDO ログが一時的に非対称になっても問題はない。たとえば、ミラー化したオンライン REDO ログ・ファイルのグループを使っている場合、他のグループにメンバーがそれぞれ 2 つ残っていても、あるグループのメンバーを 1 つ削除できます。ただし、すべてのグループに少なくともメンバーが 2 つ存在するように、この状態をただちに訂正し、オンライン REDO ログに対して単一点障害の発生する可能性をなくしてください。
- グループ内のメンバー数にかかわらず、インスタンスには常に少なくとも 2 つの有効なオンライン REDO ログ・ファイルのグループが必要。(1 つのグループは 1 つ以上のメンバーから構成されます)。削除するメンバーがグループの最後の有効なメンバーである場合、他のメンバーが有効になるまで、そのメンバーを削除できません。REDO ログ・ファイルの状態を確認するには、V\$LOGFILE ビューを使います。REDO ログ・ファイルは、Oracle がアクセスできないと INVALID になります。Oracle8 がそのログ・ファイルを完全でない、または正しくないと判断すると STALE になります。この使われなくなったログ・ファイルは、次にそのグループがアクティブ・グループになったとき、もう 1 度有効になります。
- オンライン REDO ログ・メンバーは、アクティブ・グループの一部でない場合にだけ削除できる。アクティブ・グループのメンバーを削除する場合は、まずログ・スイッチを発生させます。
- メンバーを削除する前に、そのオンライン REDO ログ・メンバーが属するグループがアーカイブされていることを確認する(アーカイブが使用可能になっている場合)。アーカイブされているかどうかを確認するには、Enterprise Manager の ARCHIVE LOG コマンドに LIST パラメータを指定して実行します。

特定のアクティブでないオンライン REDO ログ・メンバーを削除するには、Enterprise Manager の「ログ・ファイル・メンバーの削除」メニュー項目を使うか、または SQL コマンド ALTER DATABASE に DROP LOGFILE MEMBER 句を指定して使います。

次の文では、REDO ログ LOG3C が削除されます。

```
ALTER DATABASE DROP LOGFILE MEMBER 'log3c';
```

データベースからオンライン REDO ログ・メンバーを削除するときに、そのオペレーティング・システム・ファイルがディスクから削除されるわけではありません。つまり、データベース構造からメンバーを削除するために、対応するデータベースの制御ファイルが更新されます。オンライン REDO ログ・ファイルを削除した後で、処理が正常終了したことを確認し、適切なオペレーティング・システム・コマンドを使って、削除したオンライン REDO ログ・ファイルを実際に削除してください。

関連項目：アクティブ・グループのメンバー削除の詳細は、5-12 ページの「ログ・スイッチを強制する」を参照してください。

チェックポイントとログ・スイッチの制御

チェックポイントは、データベース・ライター・プロセス (DBWR) が SGA 内の修正されたデータベース・バッファを適切なデータ・ファイルに書き込むイベントです。ログ・スイッチは、LGWR があるオンライン REDO ログ・グループに書き込むのをやめて、別のログ・グループへの書き込みを開始するイベントです。この 2 つのイベントはよく連結されます。特に何も指定しないと、インスタンスはログ・スイッチごとにチェックポイントを実行します。特に指定しない場合、現行のオンライン REDO ログ・ファイル・グループが満杯になると、ログ・スイッチが自動的に発生します。

ただし、ログ・スイッチによるチェックポイントよりも多くのチェックポイントを発生させたり、ログ・スイッチを発生させずに、それよりも前にチェックポイントを発生させたりもできます。また、ログ・スイッチとチェックポイントを予定より早く発生させたり、チェックポイントを発生させずにログ・スイッチを発生させることもできます。

ここでは、チェックポイントとログ・スイッチについて次のトピックを説明しています。

- データベースのチェックポイント間隔を設定する
- ログ・スイッチを強制する
- ログ・スイッチなしで高速データベース・チェックポイントを強制する

データベースのチェックポイント間隔を設定する

大きなオンライン REDO ログ・ファイルを使っている場合、ログ・スイッチによって自動的に発生するデータベース・チェックポイントの間に、事前に設定した間隔で自動的に発生するチェックポイントを追加できます。データベース・チェックポイントの設定が多いほど、インスタンス障害からの回復に必要な時間は短くなります。ただし、チェックポイントを完了するために余分な I/O が必要となるため、Oracle Server のパフォーマンスが低下する可能性があります。

一般に、データベースが一貫して起動時のインスタンス回復を必要としない限り、ログ・スイッチによってだけチェックポイントが発生するように間隔を設定してください。小さなオンライン REDO ログ・ファイルを使うとチェックポイントは頻繁に（ログ・スイッチごとに）発生します。

自動的に発生するデータベース・チェックポイントの頻度は、パラメータ LOG_CHECKPOINT_INTERVAL および LOG_CHECKPOINT_TIMEOUT に設定した値によって制御できます。

LOG_CHECKPOINT_INTERVAL を設定する

データベース・チェックポイントをログ・スイッチによってだけ発生させる（デフォルト）には、使っているオンライン REDO ログ・ファイルのサイズより大きな値を LOG_CHECKPOINT_INTERVAL パラメータに設定します。また、2 つのログ・スイッチ間でチェックポイントを強制的に追加発生させるには、LOG_CHECKPOINT_INTERVAL パラメータの値を使っているオンライン REDO ログ・ファイルのサイズより小さく設定します。

LOG_CHECKPOINT_INTERVAL の値は、オペレーティング・システムのブロック数です。（Oracle のデータ・ブロックではありません）。したがって、使っているオペレーティング・システムのブロック・サイズ（バイト単位）を把握していなければなりません。このサイズから、オンライン REDO ログ・ファイルあたりのオペレーティング・システムのブロック数を算出します。

たとえば、次のような条件を想定します。

- データベース・インスタンスのオンライン REDO ログ・ファイルはすべて 512KB。
- オペレーティング・システムのブロック・サイズは 512 バイト。
- チェックポイントは、オンライン REDO ログ・ファイルが半分使われた時点で発生する必要がある。

この情報から、次のように、REDO ログ・ファイルあたりのブロック数を算出できます。

$$\frac{512K/\text{redo log file}}{512 \text{ bytes/OS block}} = \text{approximately } 1000 \text{ blocks/redo log file}$$

オンライン REDO ログ・ファイルあたりのおおよそのブロック数を把握したので、インスタンスのパラメータ・ファイルに LOG_CHECKPOINT_INTERVAL を次のように設定できます。

```
LOG_CHECKPOINT_INTERVAL=500
```

LOG_CHECKPOINT_TIMEOUT を設定する

データベース・チェックポイントをログ・スイッチによってだけ発生させる（デフォルト）には、LOG_CHECKPOINT_TIMEOUT パラメータの値をゼロに設定します。また、2 つのログ・スイッチ間でチェックポイントを強制的に追加発生させるには、LOG_CHECKPOINT_TIMEOUT パラメータの値をオンライン REDO ログ・ファイルが満杯

になる平均時間よりも短い間隔（秒単位）に設定します。オンライン REDO ログ・ファイルが満杯になるまでの平均時間を判断するには、ログ・スイッチの時間を示すメッセージについて LGWR トレース・ファイルを調べてください。

関連項目：オペレーティング・システムのブロック・サイズを決める方法の詳細は、オペレーティング・システム固有の Oracle マニュアルを参照してください。

チェックポイントに関する Oracle のチューニングの詳細は、『Oracle8 Server チューニング』を参照してください。

Oracle Parallel Server を使用しているときの LOG_CHECKPOINT_TIMEOUT パラメータの詳細は、『Oracle8 Parallel Server 概要および管理』を参照してください。

ログ・スイッチを強制する

ログ・スイッチを強制的に発生させて、現在アクティブなグループをアクティブでない状態にし、オンライン REDO ログのメンテナンス操作に使えます。たとえば、現在アクティブなグループを削除したい場合、アクティブでない状態になるまでそのグループを削除できないことがあります。また、現在アクティブなグループのメンバーが完全に満杯になる前に、特定の時点でそのグループをアーカイブする必要がある場合にも、ログ・スイッチの強制的な発生が必要です。このオプションは、満杯になるまで長い時間を必要とする、大きなオンライン REDO ログ・ファイルを使って構成する場合に有効です。

ログ・スイッチを強制するには、ALTER SYSTEM システム権限を持っていないけません。ログ・スイッチを強制するには、Enterprise Manager の「ログ・ファイル・スイッチ」メニュー項目を使うか、または SQL コマンド ALTER SYSTEM に SWITCH LOGFILE オプションを指定して使います。

次の文は、ログ・スイッチを強制します。

```
ALTER SYSTEM SWITCH LOGFILE;
```

ログ・スイッチなしで高速データベース・チェックポイントを強制する

場合によっては、高速データベース・チェックポイントを強制できます。高速チェックポイントはログ・スイッチとは関連のないチェックポイントであり、LGWR は現行のオンライン REDO ログ・ファイルへの書き込みを続けます。高速チェックポイントによって、DBWR は 1 回の I/O でより多くの変更済みデータベース・バッファをディスクに書き込むことができるようになります。したがって、高速チェックポイントは、処理の完了に必要な I/O が少なくてすみ、それによって時間が短くてすみます。

データベース・チェックポイントを強制するには、ALTER SYSTEM システム権限を持っていないけません。高速データベース・チェックボックスを強制するには、Enterprise Manager の「チェックボックスの強制」メニュー項目を使うか、SQL コマンド ALTER SYSTEM に CHECKPOINT オプションを指定して使います。

次の文は、チェックポイントを強制します。

```
ALTER SYSTEM CHECKPOINT;
```

GLOBAL オプションを指定すると、データベースのすべてのインスタンスに対してチェックポイントが強制され、指定しないと、接続されているインスタンスだけにチェックポイントが強制されます。Oracle Parallel Server を使っている場合に限り、ローカル・インスタンスにだけチェックポイントを強制すると便利です。パラレル・サーバー以外の構成では、グローバル・チェックポイントとローカル・チェックポイントは同じです。

関連項目：Oracle Parallel Server のチェックポイント強制の詳細は、『Oracle8 Parallel Server 概要および管理』を参照してください。

REDO ログ・ファイル内のブロックの検証

Oracle は、チェックサムを使って REDO ログ・ファイル内のブロックを検証するように構成できます。REDO ログ・ブロックがチェックできるようにするには、初期化パラメータ LOG_BLOCK_CHECKSUM を TRUE に設定します。LOG_BLOCK_CHECKSUM のデフォルト値は、FALSE です。

REDO ログ・ブロックをチェックできるようにすると、Oracle は現行のログに書き込まれた各 REDO ログ・ブロックのチェックサムを算出します。チェックサムは、ブロックのヘッダーに書き込まれます。

Oracle はチェックサムを使って、REDO ログ・ブロック内の破損データを検出します。Oracle は、REDO ログ・ブロックをアーカイブ・ログ・ファイルに書き込むとき、およびブロックが回復処理中にアーカイブ済みログから読み込まれるときに、そのブロックを検証しようとしています。

Oracle は、REDO ログ・ブロックをアーカイブしようとしているときにブロック内で破損データを検出すると、グループの中の別のメンバーからそのブロックを読み込もうとします。REDO ログ・グループ内のすべてのメンバーでブロックが破損していると、アーカイブ処理ができません。

関連項目：REDO ログ・ファイルのアーカイブ方法の詳細は、第 23 章「REDO 情報のアーカイブ」を参照してください。

オンライン REDO ログ・ファイルの消去

REDO ログ・ブロックのチェックができる場合、Oracle は各ブロックをアーカイブする前に検証します。特定の REDO ログ・ブロックがグループのすべてのメンバーで破損している場合は、アーカイブが停止します。最終的にはすべての REDO ログが埋め込まれ、データベース・アクティビティが停止してから、アーカイブが再開できるようになります。

この状況では、SQL コマンド ALTER DATABASE... CLEAR LOGFILE を使うと、破損した REDO ログを消去してアーカイブを防ぐことができます。消去された REDO ログはアーカイブされていなくても、使えます。

次の文は、REDO ログ・グループ 3 のログ・ファイルを消去します。

```
ALTER DATABASE CLEAR UNARCHIVED LOGFILE GROUP 3;
```

制限

REDO ログ・ファイルは、アーカイブされていてもアーカイブされていなくても消去できます。ただし、アーカイブされていない場合は、キーワード UNARCHIVED を挿入する必要があります。

バックアップの回復に必要なログ・ファイルを消去すると、そのバックアップからの回復処理ができなくなります。Oracle は、回復処理を実行できないバックアップについて記述する警告ログにメッセージを書き込みます。

注意： アーカイブされていない REDO ログ・ファイルを消去する場合は、データベースのバックアップをもう 1 つ作る必要があります。

オフラインの表領域をオンラインにするために使う、アーカイブされていない REDO ログを消去するには、ALTER DATABASE コマンドに UNRECOVERABLE DATAFILE 句を指定して使ってください。

オフラインの表領域をオンラインにするために必要な REDO ログを消去すると、その表領域を二度とオンラインにはできなくなります。表領域を削除するか、不完全回復を実行する必要があります。

関連項目： ALTER DATABASE コマンドの詳細は、『Oracle8 Server SQL リファレンス』を参照してください。

オンライン REDO ログ・ファイルに関する情報のリスト

データベースのオンライン REDO ログ・ファイルに関する情報を参照するには、ビュー V\$LOG、および V\$LOGFILE、V\$THREAD を使ってください。V\$THREAD ビューは Parallel Server 管理者にとって特に有効です。

次の問合せでは、Parallel Server を使わないデータベースのオンライン REDO ログ・ファイルに関する情報が戻されます。

```
SELECT group#, bytes, members
FROM sys.v$log;
```

GROUP#	BYTES	MEMBERS
-----	-----	-----
1	81920	2
2	81920	2

グループのすべてのメンバーの名前を表示するには、次のような問合せを使ってください。

```
SELECT *
FROM sys.v$logfile
WHERE group# = 2;
```

GROUP#	BYTES	MEMBERS
-----	-----	-----

2		LOG2A
2	STALE	LOG2B
2		LOG2C

メンバーの STATUS が空白の場合、そのファイルは使用中です。

制御ファイルの管理

この章では、データベースの制御ファイルを作り、メンテナンスする方法について説明します。トピックは次のとおりです。

- 制御ファイルのガイドライン
- 制御ファイルの作成
- 制御ファイルの作成後の問題解決
- 制御ファイルの削除

関連項目：この章には、Oracle Enterprise Manager について述べている箇所がいくつかあります。Enterprise Manager/GUI または Server Manager/LineMode を使って特定のタスクを実行する方法の詳細は、『Oracle Enterprise Manager 管理者ガイド』および『Oracle Server Manager ユーザーズ・ガイド』を参照してください。

制御ファイルのガイドライン

ここでは、データベースの制御ファイルを管理するためのガイドラインについて説明します。トピックは次のとおりです。

- 制御ファイルを命名する
- 制御ファイルを異なるディスク上に多重化する
- 制御ファイルを正しく配置する
- 制御ファイルのサイズを管理する

制御ファイルを命名する

データベースのパラメータ・ファイル内の `CONTROL_FILES` パラメータには、1 つ以上の制御ファイルの名前を指定します。`CONTROL_FILES` には、1 つ以上の制御ファイルの名前をコンマで区切って指定します。インスタンス起動時に、指定されたすべてのファイルが認識され、オープンされます。データベースの稼働中、インスタンスは指定された制御ファイルをすべてメンテナンスします。

Oracle Server は、データベースの稼働中に `CONTROL_FILES` パラメータに指定されたすべてのファイルに書き込みます。

制御ファイルを異なるディスク上に多重化する

どの Oracle のデータベースでも、制御ファイルは複数にし、それぞれを別々のディスクに格納してください。ディスク障害によって制御ファイルが破損すると、対応するインスタンスは停止させなければなりません。ディスク・ドライブを修復した後、破損した制御ファイルは完全な制御ファイルのコピーを使って復元し、インスタンスを再起動できます。つまり、メディア回復は必要ありません。

多重化された制御ファイルの動作

次に示す項目は、多重化された制御ファイルの動作について説明したものです。

- 2 つ以上のファイル名をデータベースのパラメータ・ファイルの初期化パラメータ `CONTROL_FILES` に指定する。
- `CONTROL_FILES` パラメータ・リストの最初のファイルだけが、データベースの稼働中に Oracle Server によって読み込まれる。
- データベースの稼働中に制御ファイルのどれかが使用不可能になった場合、インスタンスは作動できなくなり、異常終了する。

複数の制御ファイルを持つことの唯一の欠点は、制御ファイルを更新するすべての操作（データ・ファイルの追加やチェックポイントの実行など）に要する時間が少し長くなることです。ただし、この違いは通常それほど重要ではなく（特に複数の同時書き込みが可能な

オペレーティング・システムの場合)、単一制御ファイルの方がよいという裏付けにはなりません。

注意： Oracle では、各種ディスク上に使っているデータベースの 2 つ以上の制御ファイルを作っておくことを強くお勧めしています。

制御ファイルを正しく配置する

制御ファイルのコピーはそれぞれ異なるディスク上に格納する必要があります。さらに、オンライン REDO ログを多重化している場合は、オンライン REDO ログ・グループのメンバーが格納されているすべてのディスク上に制御ファイルのコピーを格納しなければなりません。このような配置をすることによって、単一のディスク障害によって制御ファイルとオンライン REDO ログ・グループがすべて失われる危険を少なくします。

制御ファイルのサイズを管理する

制御ファイルのサイズは、主に、対応するデータベースを作った CREATE DATABASE 文のパラメータ MAXDATAFILES、MAXLOGFILES、MAXLOGMEMBERS、MAXLOGHISTORY、MAXINSTANCES によって決まります。これらのパラメータの値を大きくすると、対応するデータベースの制御ファイルのサイズも大きくなります。

関連項目： 制御ファイルの最大サイズは、オペレーティング・システムによって異なります。詳細は、オペレーティング・システム固有の Oracle マニュアルを参照してください。

制御ファイルの作成

各 Oracle データベースにも、制御ファイルがあります。制御ファイルはデータベースの物理構造を記録するもので、次のような内容が含まれています。

- データベース名
- 対応付けられたデータベースとオンライン REDO ログ・ファイルの名前と位置
- データベース作成のタイムスタンプ
- 現行のログ順序番号
- チェックポイント情報

Oracle データベースの制御ファイルはデータベースとともに作られます。デフォルトでは、制御ファイルのコピーが、データベースの作成時に少なくとも 1 つ作られなければなりません。一部のオペレーティング・システムでは、Oracle が複数のコピーを作るものもあります。データベース作成時に、2 つ以上の制御ファイルのコピーを作ることをお勧めします。その後も、制御ファイルをなくしたり、制御ファイル内のある設定を変更したりする場合には、制御ファイルを作る必要があります。

ここでは、制御ファイルを作る方法について説明します。トピックは次のとおりです。

- 初期制御ファイルを作成する
- 制御ファイルの追加コピーの作成と制御ファイルの改名 / 再配置
- 新しい制御ファイル
- 新しい制御ファイルを作成する

初期制御ファイルを作成する

Oracle データベースの初期制御ファイルは、データベース作成時に使われるパラメータ・ファイル内の `CONTROL_FILES` パラメータに、1 つ以上の制御ファイル名を指定することによって作られます。`CONTROL_FILES` パラメータのファイル名は省略せずに指定してください。ファイル名の仕様は、オペレーティング・システムによって異なります。

指定した名前のファイルがデータベース作成の時点ですでに存在する場合、`CREATE DATABASE` コマンドに `CONTROLFILE REUSE` パラメータを指定しなければなりません。そうしないとエラーが発生します。なお、古い制御ファイルと新しい制御ファイルのサイズが異なる場合には、`REUSE` オプションは使えません。制御ファイル内に指定されたファイル数によって制御ファイルの大きさが異なるように、Oracle の新バージョンではリリース間で制御ファイルのサイズが異なります。つまり、`MAXLOGFILES`、`MAXLOGMEMBERS`、`MAXLOGHISTORY`、`MAXDATAFILES`、`MAXINSTANCES` などの構成パラメータが制御ファイルのサイズに影響します。

データベースを作る前に `CONTROL_FILES` にファイル名を指定しなかった場合、Oracle はデフォルトのファイル名を使います。デフォルトのファイル名もオペレーティング・システムによって異なります。

`CONTROL_FILES` パラメータの値を後で変更して、制御ファイルを追加したり、既存の制御ファイルの名前や位置を変更できます。

関連項目：制御ファイルの指定の詳細は、オペレーティング・システム固有の Oracle のマニュアルを参照してください。

制御ファイルの追加コピーの作成と制御ファイルの改名 / 再配置

既存のファイルを新しい位置にコピーし、そのファイル名を制御ファイルのリストに追加することによって、新しい制御ファイルを追加できます。

同じように、既存の制御ファイルを新しい位置にコピーし、制御ファイルのリストに指定されているそのファイルの名前を変更することによって、制御ファイルを改名できます。

どちらの場合も、作業中に制御ファイルが変更されないように、制御ファイルをコピーする前にインスタンスを停止してください。

現行制御ファイルの追加コピーを多重化、または移動する手順

1. データベースを停止する。
2. Enterprise Manager を終了する。

3. オペレーティング・システムのコマンドを使って、既存の制御ファイルを異なる位置にコピーする。
4. データベースのパラメータ・ファイルの CONTROL_FILES パラメータを編集して、新たな制御ファイルの名前を追加するか、または既存の制御ファイル名を変更する。
5. Enterprise Manager を再起動する。
6. データベースを再起動する。

新しい制御ファイル

CREATE CONTROLFILE コマンドを使って、データベースの新しい制御ファイルを作成できます。これは、次のような状況で実行します。

- データベースの制御ファイルがすべて破損し、制御ファイルのバックアップがない。
- データベースの名前、MAXLOGFILES および MAXLOGMEMBERS、MAXLOGHISTORY、MAXDATAFILES、MAXINSTANCES など、CREATE DATABASE 文で最初に指定したデータベース設定の 1 つを変更する。

たとえば、分散環境においてデータベースの名前が別のデータベースの名前と競合する場合、その名前の変更が必要になることがあります。または、上記パラメータの 1 つで、最初の設定が小さすぎる場合にもその値を変更する必要があります。

次の文は、PROD データベース（以前は別のデータベース名を使っていたデータベース）用に新しい制御ファイルを作ります。

```
CREATE CONTROLFILE
SET DATABASE prod
LOGFILE GROUP 1 ('logfile1A', 'logfile1B') SIZE 50K,
GROUP 2 ('logfile2A', 'logfile2B') SIZE 50K
NORESETLOGS
DATAFILE 'datafile1' SIZE 3M, 'datafile2' SIZE 5M
MAXLOGFILES 50
MAXLOGMEMBERS 3
MAXDATAFILES 200
MAXINSTANCES 6
ARCHIVELOG;
```

警告：CREATE CONTROLFILE コマンドは、指定したデータ・ファイルとオンライン REDO ログ・ファイルに損傷を与える可能性があります。ファイル名を指定しないと、そのファイル内のデータが失われたり、データベースそのものにアクセスできなくなることもあります。このコマンドを使うときには十分注意し、必ずこの後の部分の手順に従ってください。

関連項目：CREATE CONTROLFILE コマンドの詳細は、『Oracle8 Server SQL リファレンス』を参照してください。

新しい制御ファイルを作成する

ここでは、新しい制御ファイルを作るための手順について示します。

新しい制御ファイルを作る手順

1. データベースのデータ・ファイルとオンライン REDO ログ・ファイルすべてのリストを作る。

データベース・バックアップの推奨事項に従っていれば、現在のデータベース構造を反映するデータ・ファイルとオンライン REDO ログ・ファイルのリストがすでにあるはずです。

このようなリストが手元になく、制御ファイルが破損してデータベースがオープンできない場合、データベースを構成するデータ・ファイルとオンライン REDO ログ・ファイルのすべてをご自分の判断で配置してください。新しい制御ファイルが作られると、ステップ 5 で指定したファイル以外は回復できません。さらに、SYSTEM 表領域を構成するファイルを 1 つでも指定しないと、データベースを回復できないことがあります。

2. データベースを停止する。

データベースがオープンしている場合、できる限り通常の手順でデータベースを停止してください。最後の手段としてだけ、IMMEDIATE オプションまたは ABORT オプションを使ってください。

3. データベースのデータ・ファイルとオンライン REDO ログ・ファイルすべてのバックアップを作る。
4. 新しいインスタンスを起動する。ただし、データベースのマウントとオープンは行いません。
5. CREATE CONTROLFILE コマンドを使って、データベースの新しい制御ファイルを作る。

制御ファイルに加えて、オンライン REDO ログ・グループも失ってしまった場合には、新しい制御ファイルを作る際に、RESETLOGS オプションを選択してください。この場合には、失った REDO ファイル (ステップ 8) から回復する必要があります。データベースを改名している場合は、必ず RESETLOGS オプションも指定してください。それ以外の場合には、NORESETLOGS オプションを選択してください。

6. 新しい制御ファイルのバックアップをオフラインの記憶デバイスに格納する。
7. データベースのパラメータ・ファイルを編集する。

データベースのパラメータ・ファイルを編集して、ステップ 5 とステップ 6 で作った制御ファイルをすべて (バックアップの制御ファイルは含まない) CONTROL_FILES パラメータに指定してください。

8. 必要に応じて、データベースを回復させる。

回復の一部として制御ファイルを作っている場合には、データベースを回復してください。新しい制御ファイルを NORESETLOGS オプションを使って作った場合（ステップ 5）、完全な、クローズ・データベース回復でデータベースを回復できます。

RESETLOGS オプションを使って新しい制御ファイルを作った場合は、USING BACKUP CONTROL FILE を指定する必要があります。オンラインまたはアーカイブ済みの REDO ログ、またはデータ・ファイルが失われた場合は、これらのファイルの回復手順に従ってください。

9. データベースをオープンする。

次の方法の 1 つを使って、データベースをオープンしてください。

- 回復を実行しなかった場合、通常の手順でデータベースをオープンする。
- ステップ 8 でクローズされた完全なデータベース回復を行った場合、Enterprise Manager の「データベースの起動」ダイアログ・ボックスの「起動してオープン」ラジオ・ボタンを使用します。
- 制御ファイルを作るときに RESETLOGS を指定した場合、RESETLOGS を指定した ALTER DATABASE コマンドを使う。

これでデータベースはオープンされ、使用可能になっています。

関連項目：データベース・ファイルのリストの詳細は、『Oracle8 Server バックアップおよびリカバリ』を参照してください。

データベースのすべてのデータファイルおよびオンライン REDO ログ・ファイルのバックアップの詳細は、『Oracle8 Server バックアップおよびリカバリ』を参照してください。

オンラインおよびアーカイブ済み REDO ログ・ファイルの回復の詳細は、『Oracle8 Server バックアップおよびリカバリ』を参照してください。

クローズされたデータベース回復の詳細は、『Oracle8 Server バックアップおよびリカバリ』を参照してください。

制御ファイルの作成後の問題解決

CREATE CONTROLFILE 文を実行した後で、一般的なエラーが発生する場合があります。ここでは、制御ファイルの使用に関連する最も一般的なエラーについて説明します。トピックは次のとおりです。

- 欠落したまたは余分なファイルをチェックする
- CREATE CONTROLFILE でのエラーを処理する

欠落したまたは余分なファイルをチェックする

新しい制御ファイルを作り、これを使ってデータベースをオープンした後、ALERT ログをチェックして、Oracle がデータ・ディクショナリと制御ファイルの間で不整合（たとえば、

データ・ディクショナリにはデータ・ファイルがあるが、制御ファイルには示されていないなど)を検出したかどうかを調べてください。

データ・ディクショナリの中にはデータ・ファイルが存在するが、新しい制御ファイルに示されていない場合、Oracle は MISSINGnnnn (ここで nnnn は 10 進数のファイル番号) で制御ファイル内にブレースホルダ・エントリを作ります。MISSINGnnnn には、制御ファイル内でオフラインでメディア回復を必要とするエントリとしてフラグが立てられます。

次の 2 つの場合に限っては、MISSINGnnnn に対応する実際のデータ・ファイルを指すように MISSINGnnnn を改名することによって、実際のデータ・ファイルにアクセスできるようになります。

ケース 1: NORESETLOGS オプションを指定した CREATE CONTROLFILE コマンドを使って新しい制御ファイルが作られている場合。これによって、RESETLOGS オプションを使わずにデータベースをオープンできます。これができるのは、すべてのオンライン REDO ログが使える場合だけです。

ケース 2: CREATE CONTROLFILE コマンドで RESETLOGS オプションを使う必要があった場合。RESETLOGS オプションを使ってデータベースを強制的にオープンしますが、MISSINGnnnn に対応する実際のデータ・ファイルは読み専用または通常のアフラインでした。

一方、RESETLOGS オプションを使ってデータベースをオープンする必要があったときに、MISSINGnnnn が読み専用または通常のアフラインでないデータ・ファイルに対応している場合は、改名操作を使ってもデータ・ファイルへのアクセスを可能にはできません (データ・ファイルに RESETLOGS の結果として抑止されているメディア回復が必要になるため)。この場合、そのデータ・ファイルが入っている表領域を削除しなければなりません。

反対に、制御ファイルに指定されているデータ・ファイルが、データ・ディクショナリに存在しない場合、Oracle は新しい制御ファイルからそのファイルへの参照を削除します。どちらの場合も、Oracle は ALERT ファイルにどんな状態が検出されたかを通知するメッセージを書き込みます。

CREATE CONTROLFILE でのエラーを処理する

新しい制御ファイルを作った後、データベースをマウントしてオープンしようとしたときに、Oracle がエラー (通常、ORA-01173、ORA-01176、ORA-01177、ORA-01215、ORA-01216 のどれか) を送信した場合、最も可能性の高い原因は、CREATE CONTROLFILE 文に指定し忘れたファイルがあるか、またはリストに示されていないファイルを指定したということです。この場合、ステップ 3 でバックアップしたファイルを復元し、正しいファイル名を使ってステップ 4 から手順を実行し直してください。

制御ファイルの削除

制御ファイルはデータベースから削除できます。たとえば、制御ファイルの位置が不適切な場合には、その制御ファイルを削除できます。ただし、データベースには常に少なくとも 2 つの制御ファイルが存在しなければなりません。

1. データベースを停止する。
2. Enterprise Manager を終了する。
3. データベースのパラメータ・ファイルの CONTROL_FILES パラメータを編集して、古い制御ファイルの名前を削除する。
4. Enterprise Manager を再起動する。
5. データベースを再起動する。

警告： この操作では、不要な制御ファイルがディスクから物理的に削除されるわけではありません。データベースから制御ファイルを削除した後、オペレーティング・システムのコマンドを使って不要なファイルを削除してください。

ジョブ・キューの管理

この章では、ジョブ・キューを使って PL/SQL コードを定期的に行うスケジュール方法について説明します。この章のトピックは次のとおりです。

- SNP バックグラウンド・プロセス
- ジョブ・キューの管理
- ジョブ・キューについての情報の表示

関連項目：この章には、Oracle Enterprise Manager について述べている箇所がいくつかあります。Enterprise Manager/GUI または Server Manager/LineMode を使って特定のタスクを実行する方法の詳細は、『Oracle Enterprise Manager 管理者ガイド』および『Oracle Enterprise Manager ユーザーズ・ガイド』を参照してください。

SNP バックグラウンド・プロセス

ここでは、SNP バックグラウンド・プロセスと、ジョブ・キューの管理におけるその役割について説明します。トピックは次のとおりです。

- 複数の SNP プロセス
- SNP プロセスを起動する

ジョブ・キューを使うと、ルーチンが定期的に行われるようにスケジュールできます。このルーチンとは PL/SQL コードのことです。ジョブをスケジュールするには、ジョブをジョブ・キューに入れて、ジョブを実行する頻度を指定します。待ち行列に入れられたジョブは、変更または削除、使用禁止にできます。

パフォーマンスを最大にし、多くのユーザーから使えるようにするために、マルチプロセス Oracle システムでは「バックグラウンド・プロセス」と呼ばれるいくつかの付加的なプロセスが使われます。バックグラウンド・プロセスは、各ユーザー・プロセスで実行される複数の Oracle プログラムが処理するファンクションを整理統合します。バックグラウンド・プロセスは非同期的に I/O を実行して他の Oracle プロセスを監視し、並列性を高め、パフォーマンスと信頼性を向上させます。

SNP バックグラウンド・プロセスでは、ジョブ・キューを実行します。SNP プロセスでは、待ち行列に入っている実行予定のジョブを定期的に行動し、実行します。待ち行列に入っているジョブをバックグラウンドで実行するためには、1 つ以上の SNP プロセスが稼働していなければなりません。

SNP バックグラウンド・プロセスは、Oracle の他のバックグラウンド・プロセスとは異なり、SNP プロセスで障害が発生しても、インスタンスには影響を与えません。SNP プロセスで障害が発生すると、Oracle がこのプロセスを再起動させます。

システムが制限モードで開始された場合、SNP バックグラウンド・プロセスは、ジョブを実行しません。ただし、ALTER SYSTEM コマンドを使うと、次のように制限モードをオン・オフできます。

```
ALTER SYSTEM ENABLE RESTRICTED SESSION;  
ALTER SYSTEM DISABLE RESTRICTED SESSION;
```

制限セッションを使用可能にすると、SNP バックグラウンド・プロセスはジョブを実行しますが、使用禁止にすると実行します。

関連項目： SNP バックグラウンド・プロセスの詳細は、『Oracle8 Server 概要』を参照してください。

複数の SNP プロセス

1 つのインスタンスの SNP プロセスの数は、最大 36 個です (SNP0 ~ SNP9、および SNPA ~ SNPZ の名前)。1 つのインスタンスに複数の SNP プロセスがあると、待ち行列に入っているジョブを実行するタスクをこの複数のプロセス間で共有するため、パフォーマンスが向上します。ただし、各ジョブはどの時点でも 1 つのプロセスによって実行されることに注意してください。1 つのジョブを複数の SNP プロセスで同時には共有できません。

SNP プロセスを起動する

ジョブ・キューの初期化パラメータを使うと、SNP バックグラウンド・プロセスの動作を制御できます。インスタンスの初期化パラメータ・ファイルで次のパラメータを設定すると、設定したパラメータは、次にインスタンスを起動したときに有効になります。

表 7-1 に、ジョブ・キューの初期化パラメータを示します。

表 7-1 ジョブ・キュー初期化パラメータ

パラメータ名	説明
JOB_QUEUE_PROCESSES	デフォルト : 0 値の範囲 : 0...36 複数インスタンス : 異なる値を設定可能 インスタンスごとの SNP バックグラウンド・プロセスの数を設定する。
JOB_QUEUE_INTERVAL	デフォルト : 60(秒) 値の範囲 : 1...3600(秒) 複数インスタンス : 異なる値を設定可能 インスタンスの SNP バックグラウンド・プロセスの起動する間隔を設定する。

ジョブ・キューの管理

ここでは、ジョブ・キューの管理について説明します。トピックは次のとおりです。

- DBMS_JOB パッケージ
- ジョブをジョブ・キューに送る
- ジョブの実行方法
- ジョブ・キューからジョブを削除する
- ジョブを変更する
- 中断されたジョブ
- ジョブを強制的に実行する
- ジョブを終了する

DBMS_JOB パッケージ

ジョブ・キュー内のジョブのスケジューリングおよび管理は、DBMS_JOB パッケージのプロシージャを使って実行します。ジョブ・キューの使用に関係付けられたデータベース権限はありません。ジョブ・キュー・プロシージャを実行できるユーザーであればどのユーザーでもジョブ・キューを使えます。

表 7-2 に、DBMS_JOB パッケージに含まれているジョブ・キュー・プロシーダを示します。

表 7-2 DBMS_JOB パッケージのプロシーダ

プロシーダ	説明	次のものに関する説明
SUBMIT	ジョブをジョブ・キューに送る。	7-5 ページ
REMOVE	ジョブ・キューから指定したジョブを削除する。	7-11 ページ
CHANGE	指定したジョブを変更する。ジョブの定義、ジョブの実行時刻、ジョブの実行間隔が変更できる。	7-11 ページ
WHAT	指定したジョブのジョブの定義を変更する。	7-11 ページ
NEXT_DATE	指定したジョブの次の実行時刻を変更する。	7-12 ページ
INTERVAL	指定したジョブの実行間隔を変更する。	7-12 ページ
BROKEN	ジョブの実行を禁止にする。ジョブに中断状態を示すマークを付けると、Oracle はそのジョブを実行しない。	7-12 ページ
RUN	指定したジョブを強制的に実行させる。	7-13 ページ

ジョブをジョブ・キューに送る

新しいジョブをジョブ・キューに送るには、DBMS_JOB パッケージの SUBMIT プロシーダを使います。

```
DBMS_JOB.SUBMIT( job OUT BINARY_INTEGER,
                 what          IN   ARCHAR2,
                 next_date     IN   DATE DEFAULT SYSDATE,

                 interval      IN   VARCHAR2 DEFAULT 'null',
                 no_parse      IN   BOOLEAN DEFAULT FALSE)
```

SUBMIT プロシージャは、送られたジョブ番号を戻します。表 7-3 に、プロシージャのパラメータを示します。

表 7-3 DBMS_JOB.SUBMIT のパラメータ

パラメータ	説明
job	作ったジョブに割り当てる識別子。ジョブを変更または削除するときは必ず、ジョブ番号を使う。 ジョブ番号の詳細は、7-8 ページの「ジョブ番号」を参照。
what	実行する PL/SQL コード。 ジョブ定義方法の詳細は、7-8 ページの「ジョブ定義」を参照。
next_date	ジョブを次に実行する日付。デフォルト値は SYSDATE。
interval	次にジョブを実行する時刻を計算する日付ファンクション。デフォルト値は NULL。INTERVAL は将来のある時点または NULL に評価されなければならない。 実行間隔の指定方法の詳細は、7-8 ページの「ジョブの実行間隔」を参照。
no_parse	フラグ。デフォルト値は FALSE。 NO_PARSE を FALSE (デフォルト) に設定すると、Oracle はそのジョブに対応付けられているプロシージャを解析する。 NO_PARSE を TRUE に設定すると、Oracle はそのジョブの最初の実行時にそのジョブに対応付けられたプロシージャを解析する。たとえば、ジョブに対応付けられている表を作る前に、そのジョブを送る場合は、NO_PARSE を TRUE に設定する。

例として、新しいジョブをジョブ・キューに送るとします。このジョブは、DBMS_DDL.ANALYZE_OBJECT プロシージャをコールして、DQUON.ACCOUNTS 表のオプティマイザの統計表示を生成します。統計表示は、ACCOUNTS 表全体の行の半分をサンプルにしています。このジョブは、24 時間ごとに実行されます。

```
SVRMGR> VARIABLE jobno number;
SVRMGR> begin
2>         DBMS_JOB.SUBMIT(:jobno,
3>         'dbms_ddl.analyze_object(''TABLE'',
4>         'DQUON'', 'ACCOUNTS'',
5>         'ESTIMATE'', NULL, 50);'
6>         SYSDATE, 'SYSDATE + 1');
7>         commit;
8> end;
9> /
Statement processed.
SVRMGR> print jobno
JOBNO
-----
14144
```

ジョブ環境

ジョブがジョブ・キューに送られるか、ジョブの定義が変更されると、Oracle は次の環境特性を記録します。

- 現ユーザー
- ジョブを送るユーザーまたはジョブを変更するユーザー
- 現行のスキーマ
- MAC 権限（該当する場合）

次の NLS パラメータも記録されます。

- NLS_LANGUAGE
- NLS_TERRITORY
- NLS_CURRENCY
- NLS_ISO_CURRENCY
- NLS_NUMERIC_CHARACTERS
- NLS_DATE_FORMAT
- NLS_DATE_LANGUAGE
- NLS_SORT

これらの環境特性はジョブが実行されるたびに再格納されます。NLS_LANGUAGE パラメータと NLS_TERRITORY パラメータは、NLS パラメータが指定されなかった場合のデフォルトです。

DBMS_SQL パッケージと ALTER SESSION コマンドを使って、ジョブの環境を変更できます。

ジョブとインポート/エクスポート

ジョブは、インポートしたりエクスポートしたりできます。したがって、1つのデータベースで定義したジョブを別のデータベースに転送できます。ジョブのインポートでもエクスポートでも、ジョブ番号、環境、定義は変更されることはありません。

注意： インポートするジョブのジョブ番号がデータベースの既存のジョブの番号と同じ場合、そのジョブはインポートできません。そのジョブをデータベースの新しいジョブとして送ってください。

ジョブの所有者

ジョブをジョブ・キューに送ったユーザーは、ジョブの所有者として識別されます。ジョブの変更や強制実行、キューからのジョブの削除ができるのは、ジョブの所有者だけです。

ジョブ番号

キューに入っているジョブは、ジョブ番号によって識別されます。ジョブを送ると、SYS.JOBSEQ 順序を使ってジョブ番号が自動的に生成されます。

一度ジョブにジョブ番号が割り当てられると、その番号は変わりません。ジョブをインポートしたりエクスポートしたりしても、同じジョブ番号のままです。

ジョブ定義

SUBMIT プロシージャの WHAT パラメータに指定された PL/SQL コードがジョブ定義です。

標準では、ジョブ定義は 1 つのプロシージャを 1 度コールします。プロシージャ・コールには、任意の数のパラメータを指定できます。

注意： ジョブ定義では、文字列の前後を一重引用符で囲んでください。
ジョブ定義の末尾には必ずセミコロンを入れてください。

Oracle がジョブ定義の中で認識する特殊なパラメータ値がいくつかあります。表 7-4 に、このようなパラメータを示します。

表 7-4 ジョブ定義の特殊なパラメータの値

パラメータ	モード	説明
job	IN	現行ジョブの番号
next_date	IN/OUT	ジョブが次に実行される日付。デフォルト値は SYSDATE。
broken	IN/OUT	ジョブが中断されているか中断されていないかの状態。IN の値は FALSE。

次に、有効なジョブ定義の例を示します。

```
'myproc(''10-JAN-82'', next_date, broken);'  
'scott.emppackage.give_raise(''JFEE'', 3000.00);'  
'dms_job.remove(job);'
```

ジョブの実行間隔

ジョブの実行の直前に、INTERVAL 日付ファンクションが評価されます。ジョブが正常に実行されると、INTERVAL から計算された日付が新しい NEXT_DATE になります。

INTERVAL 日付ファンクションが NULL に評価され、ジョブが正常に実行されると、そのジョブは待ち行列から削除されます。

設定した間隔でジョブを定期的に行うには、INTERVAL パラメータで 'SYSDATE + 7' のような日付式を使ってください。たとえば、月曜日に実行間隔を 'SYSDATE + 7' と設定したときに、なんらかの理由で（たとえば、ネットワーク障害など）ジョブが火曜日まで実行されない場合は、'SYSDATE + 7' が月曜日ではなく毎週火曜日に実行されます。

ジョブを必ず特定の時間に自動的に実行するには、前回の実行時期（たとえば、毎週月曜日）に関係なく、INTERVAL パラメータと NEXT_DATE パラメータで 'NEXT_DAY(TRUNC(SYSDATE),'MONDAY')' のような日付式を指定する必要があります。

表 7-5 に、ジョブの実行間隔を求めるための一般的な日付式を示します。

表 7-5 一般的なジョブの実行間隔

日付式	評価
'SYSDATE + 7'	前回の実行からちょうど 7 日目
'SYSDATE + 1/48'	30 分ごと
'NEXT_DAY(TRUNC(SYSDATE), 'MONDAY') + 15/24'	毎週月曜日午後 3 時
'NEXT_DAY(ADD_MONTHS (TRUNC(SYSDATE, 'Q'), 3), 'THURSDAY')'	四半期ごとの第一木曜日

注意： NEXT_DATE または INTERVAL を指定するときは、日付リテラルと日付文字列を必ず一重引用符で囲んでください。また、INTERVAL の値も一重引用符で囲む必要があります。

データベース・リンクとジョブ

送られたジョブがデータベース・リンクを使う場合は、そのリンクにユーザー名とパスワードが入っていない必要はありません。名前のないデータベース・リンクは正常に機能しません。

関連項目： ALTER SESSION コマンドの詳細は、『Oracle8 Server SQL リファレンス』を参照してください。

DBMS_SQL パッケージの詳細は、『Oracle8 アプリケーション開発者ガイド』を参照してください。

ジョブの実行方法

各ジョブは、SNP バックグラウンド・プロセスで実行されます。ジョブを実行するため、SNP バックグラウンド・プロセスによりジョブを実行するセッションが作成されます。

SNP がジョブを実行するとき、ジョブは、送られたときと同じ環境で、所有者のデフォルトの権限で実行されます。

DBMS_JOB.RUN プロシージャを使ってジョブを強制的に実行させる場合、ジョブはユーザー・プロセスで実行されます。ユーザー・プロセスでジョブが実行される場合、ジョブは

直接付与されている権限だけで実行されます。ロールを介してユーザーに付与された権限は、使われません。

ジョブ・キューのロック

Oracle は、ジョブ・キューのロックを使って、1つのジョブが1度に1つのセッションで実行されるようにします。ジョブの実行中に、そのセッションはジョブに対するジョブ・キュー (JQ) ロックを獲得します。

JQ ロックについての情報を解釈する Enterprise Manager の「ロック・モニター」またはデータ・ディクショナリのロッキング・ビューによって、現在セッションが保持しているロックについての情報を調べることができます。

次の問合せは、JQ ロックを保持しているすべてのセッションのセッション識別子、ロック・タイプ、ロック識別子をリストします。

```
SVRMGR> SELECT sid, type, id1, id2
2> FROM v$llock
3> WHERE type = 'JQ';
```

SID	TY	ID1	ID2
12	JQ	0	14144

1 row selected.

上記の問合せでは、ロックを保持しているセッションの識別子は 12 個あります。JQ ロックでは、ID1 ロック識別子は常に 0 です。ID2 ロック識別子は、セッションで実行中のジョブのジョブ番号です。

ジョブ実行のエラー

ジョブの実行に失敗すると、トレース・ファイルとアラート・ログにその失敗に関する情報が記録されます。Oracle はメッセージ番号 ORA-12012 と失敗したジョブのジョブ番号を書き込みます。

キューに入っているジョブの正常な実行を妨げる原因として、次のものがあります。

- ジョブを実行するための SNP バックグラウンド・プロセスがない
- ネットワークまたはインスタンスの障害
- ジョブ実行時の例外

ジョブの失敗と実行時 Oracle がジョブの実行中にそのジョブがエラーを戻した場合、Oracle はそのジョブを再実行しようとします。1 分後に最初の実行が試みられ、2 分後に 2 度目、4 分後に 3 度目というようにそれぞれの試行の間隔は 2 倍になっていきます。再試行の間隔が実行間隔を超えると、Oracle は標準の実行間隔でジョブの再試行を続けます。ただし、ジョブの失敗の回数が 16 回になると、そのジョブには自動的に中断のマークが付けられ、それ以上そのジョブは実行されなくなります。

したがって、ジョブの実行失敗の回数が 16 回になる前に、そのジョブの実行を妨げる問題を訂正できれば、最終的に Oracle はそのジョブをもう 1 度実行します。

関連項目： ロッキング・ビューの詳細は、『Oracle8 Server リファレンス・マニュアル』を参照してください。

ロックの詳細は、『Oracle8 Server 概要』を参照してください。

ジョブ・キューからジョブを削除する

ジョブ・キューからジョブを削除するには、DBMS_JOB パッケージの REMOVE プロシージャを使います。

```
DBMS_JOB.REMOVE( job IN BINARY_INTEGER)
```

次の文は、ジョブ・キューからジョブ番号 14144 のジョブを削除します。

```
DBMS_JOB.REMOVE(14144);
```

制限

現在実行中のジョブをジョブ・キューから削除できます。ただし、そのジョブは中断されずに、現行の実行は完了します。

削除できるのは、自分が所有しているジョブだけです。自分が所有していないジョブを削除しようすると、そのジョブはジョブ・キューに入られていないジョブであることを示すメッセージが表示されます。

ジョブを変更する

ジョブ・キューに送られているジョブを変更するには、DBMS_JOB パッケージの CHANGE、WHAT、NEXT_DATE、INTERVAL のどれかのプロシージャを使います。

次の例では、14144 で識別されるジョブを 3 日ごとに実行するように変更します。

```
DBMS_JOB.CHANGE(14144, null, null, 'SYSDATE + 3');
```

制限

変更できるジョブは、自分の所有しているジョブだけです。自分の所有していないジョブを変更しようすると、そのジョブはジョブ・キューにないことを示すメッセージが表示されます。

CHANGE の構文

DBMS_JOB.CHANGE プロシージャをコールすると、ユーザーが定義可能なパラメータで、ジョブに対応付けられているものであればどのパラメータでも変更できます。表 7-3 に、プロシージャのパラメータを示しています。

```
DBMS_JOB.CHANGE( job          IN BINARY_INTEGER,
                  what          IN VARCHAR2,
                  next_date     IN DATE,
                  interval      IN VARCHAR2)
```

CHANGE プロシージャをコールするときに、WHAT、NEXT_DATE、INTERVAL のどれかに NULL を指定すると、現在の値は変更されません。

注意： CHANGE プロシージャで WHAT パラメータを使ってジョブの定義を変更すると、Oracle8 は現行の環境を記録します。この環境が、そのジョブの新しい環境になります。

WHAT の構文

DBMS_JOB.WHAT プロシージャをコールして、ジョブの定義を変更できます。表 7-3 に、プロシージャのパラメータを示しています。

```
DBMS_JOB.WHAT( job          IN BINARY_INTEGER,
               what          IN VARCHAR2)
```

注意： WHAT プロシージャを実行すると、Oracle は現行の環境を記録します。この環境が、そのジョブの新しい環境となります。

NEXT_DATE の構文

DBMS_JOB.NEXT_DATE プロシージャをコールすることによって、Oracle が次にジョブを実行する日付を変更できます。表 7-3 に、プロシージャのパラメータを示しています。

```
DBMS_JOB.NEXT_DATE( job      IN BINARY_INTEGER,
                   next_date  IN DATE)
```

INTERVAL の構文

DBMS_JOB.INTERVAL プロシージャをコールして、ジョブの実行間隔を変更できます。表 7-3 に、プロシージャのパラメータを示しています。

```
DBMS_JOB.INTERVAL( job      IN BINARY_INTEGER,
                  interval   IN VARCHAR2)
```

中断されたジョブ

ジョブには、中断または非中断状態を示すラベルが付けられます。中断されたジョブは実行されません。ただし、DBMS_JOB.RUN プロシージャをコールすると、中断されたジョブを強制的に実行できます。

ジョブをキューに送ると、そのジョブは中断されていないとみなされます。

ジョブは、次の 2 つの場合に中断されます。

- Oracle がジョブの実行を 16 回試みても、ジョブが正常に実行されなかった場合
- DBMS_JOB.BROKEN プロシージャを使って、ジョブに中断されたことを示すマークを付けた場合

ジョブに中断または非中断のマークを付けるには、DBMS_JOB パッケージの BROKEN プロシージャを使います。表 7-4 に、プロシージャのパラメータを示しています。

```
DBMS_JOB.BROKEN( job           IN BINARY_INTEGER,
                  broken        IN BOOLEAN,
                  next_date     IN DATE DEFAULT SYSDATE)
```

次の例では、ジョブ 14144 に非中断のマークを付け、次回の実行の日付を次の月曜日に設定します。

```
DBMS_JOB.BROKEN(14144, FALSE, NEXT_DAY(SYSDATE, 'MONDAY'));
```

中断のマークを付けられたジョブは、非中断のマークを付けるか、または DBMS_JOB.RUN プロシージャをコールしてジョブを強制的に実行させない限り、Oracle により実行されません。

制限

中断のマークを付けることができるジョブは、自分が所有しているジョブだけです。自分が所有していないジョブにマークを付けようとすると、そのジョブはジョブ・キューにないことを示すメッセージが表示されます。

中断されたジョブを実行する

問題が発生してジョブの実行が 16 回失敗すると、Oracle はそのジョブに中断のマークを付けます。発生した問題を訂正した後は、次のどちらかの方法でこのジョブを実行できます。

- DBMS_JOB.RUN をコールして、ジョブを強制実行する。
- DBMS_JOB.BROKEN をコールして、ジョブに非中断のマークを付け、Oracle がジョブを実行するのを待つ。

DBMS_JOB.RUN プロシージャをコールしてジョブを強制実行すると、そのジョブはただちに実行されます。ジョブが正常に実行されると、Oracle はそのジョブに非中断のマークを付け、そのジョブが実行に失敗した回数のカウントをリセットします。

ジョブの中断フラグを (RUN または BROKEN をコールして) リセットした後は、そのジョブに対してスケジュールされた実行間隔に従ってジョブの実行が再開されます。

ジョブを強制的に実行する

ジョブを手動で実行することがしばしばあります。たとえば、中断されたジョブを修正した場合、そのジョブを強制実行することによってただちにテストできます。

ジョブをただちに強制実行するには、DBMS_JOB パッケージの RUN プロシージャを使います。このプロシージャを使うと Oracle はジョブに中断のマークが付いていても、そのジョブを実行しようとします。

```
DBMS_JOB.RUN( job IN BINARY_INTEGER)
```

DBMS_JOB.RUN を使ってジョブを実行するとき、Oracle は次の実行の日付をもう一度計算します。たとえば、'SYSDATE' の NEXT_DATE 値と 'SYSDATE + 7' の INTERVAL 値を使ってジョブを月曜日に作ると、そのジョブは月曜日から 7 日ごとに実行されます。ただし、水曜日に DBMS_JOB. で実行すると、次の実行の日付が次の水曜日になります。

注意： ジョブを強制的に実行すると、そのジョブは現行のセッションで実行されます。ジョブを実行すると、セッションのパッケージが再度初期化されます。

制限

実行できるジョブは、自分が所有しているジョブだけです。自分が所有していないジョブを実行しようすると、そのジョブがジョブ・キューにないことを示すメッセージが表示されます。

次の文で、ジョブ 14144 は現行のセッションで実行し、次の実行の日付が改めて算出されます。

```
DBMS_JOB.RUN(14144);
```

RUN プロシージャには、暗黙のコミットが含まれています。RUN を使ってジョブを実行した後は、ロールバックはできません。

ジョブを終了する

ジョブに中断のマークを付け、そのジョブを実行しているセッションを識別して、そのセッションを切断すると、実行中のジョブを停止できます。Oracle がそのジョブを再実行しないように、そのジョブに中断のマークを付ける必要があります。

ジョブを実行しているセッションを（V\$SESSION で）識別した後で、Enterprise Manager の「セッション切断」メニュー項目または SQL コマンド ALTER SYSTEM を使ってそのセッションを切断できます。

関連項目：ジョブとセッションについての情報の表示例の詳細は、次項の「ジョブ・キューについての情報を表示する」を参照してください。

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

ジョブ・キューについての情報の表示

表 7-6 のデータ・ディクショナリ・ビューによって、ジョブ・キュー内のジョブについての情報を見ることができます。

表 7-6 ジョブ・キュー内の情報を示すビュー

ビュー	説明
DBA_JOBS	データベース内のすべてのジョブをリストする。
USER_JOBS	ユーザーが所有するすべてのジョブをリストする。
DBA_JOBS_RUNNING	データベース内の実行中のジョブをすべてリストする。このビューは V\$LOCK と JOB\$ を結合する。

たとえば、ジョブの状態および、失敗した実行についての情報を表示できます。次の問合せの例では、キューに送られている各ジョブのジョブ番号および次の実行時刻、失敗、中断の状態がリストされています。

```
SVRMGR> SELECT job, next_date, next_sec, failures, broken
2> FROM user_jobs;
```

JOB	NEXT_DATE	NEXT_SEC	FAILURES	B
-----	-----	-----	-----	-
9125	01-NOV-94	00:00:00	4	N
14144	24-OCT-94	16:35:35	0	N
41762	01-JAN-00	00:00:00	16	Y

3 rows selected.

また、現在実行中のジョブについての情報の表示もできます。次の問合せの例では、現在実行中のすべてのジョブについて、セッション識別子および、ジョブ番号、ジョブを送ったユーザー、開始時刻がリストされています。

```
SVRMGR> SELECT sid, r.job, log_user, r.this_date, r.this_sec
2> FROM dba_jobs_running r, dba_jobs j
3> WHERE r.job = j.job;
```

SID	JOB	LOG_USER	THIS_DATE	THIS_SEC
-----	-----	-----	-----	-----
12	14144	JFEE	24-OCT-94	17:21:24
25	8536	SCOTT	24-OCT-94	16:45:12

2 rows selected.

関連項目 : データ・ディクショナリ・ビューの詳細は、『Oracle8 Server リファレンス・マニュアル』を参照してください。

第 3 部

Oracle Server の構成

表領域の管理

この章では、表領域の管理について説明します。トピックは次のとおりです。

- 表領域を管理するためのガイドライン
- 表領域の作成
- 表領域割当ての管理
- 表領域の可用性の変更
- 表領域を読み込み専用にする方法
- 表領域の削除
- 表領域についての情報の表示

この章には、Oracle Enterprise Manager について述べている箇所がいくつかあります。Enterprise Manager/GUI または Server Manager/LineMode を使って特定のタスクを実行する方法の詳細は、『Oracle Enterprise Manager 管理者ガイド』および『Oracle Enterprise Manager ユーザーズ・ガイド』を参照してください。

表領域を管理するためのガイドライン

Oracle データベースの表領域を使って作業する前に、次に説明するガイドラインについて検討してください。

- 複数の表領域を使用する
- 表領域の記憶領域パラメータを使用する
- ユーザーに表領域割当て制限を割り当てる

複数の表領域を使用する

データベース操作を実行しているとき、複数の表領域を使うとシステムはより柔軟性に富んだものとなります。たとえば、データベースに複数の表領域があるときには、次のことができます。

- データ・ディクショナリのデータからユーザー・データを分離する。
- あるアプリケーションのデータを別のアプリケーションのデータから分離する。
- I/O の競合を低減するために、サーバーの別々のディスク・ドライブ上に異なる表領域を配置する。
- ロールバック・セグメントのデータをユーザー・データから分離する。
- 別の表領域をオンライン状態に保持しながら、個々の表領域をオフラインにする。
- 高い更新アクティビティ、読取り専用アクティビティ、一時セグメントなど、異なるタイプのデータベース利用のために異なる表領域を確保する。
- 個々の表領域をバックアップする。

オペレーティング・システムによっては、同時にオープンできるファイル数に制限があることがあります。つまり、これらの制限は、同時にオンラインにできる表領域の数に間接的な影響を与えます。使用しているオペレーティング・システムの制限を超えないように、表領域を効率よく計画してください。それとともに、できる限りファイル数が少なくなるように表領域を作成してください。表領域のサイズを大きくする必要がある場合は、小さいデータ・ファイルを多数作成するのではなく、1 つまたは 2 つの大きなデータ・ファイルを追加するか、または自動拡張オプションをオンに設定してデータ・ファイルを作成します。

これらの利点を考慮に入れてデータを見直し、データベース設計に必要な表領域の数を決定してください。

表領域の記憶領域パラメータを使用する

新しい表領域を作成するとき、次のルールに従って、その表領域に格納されるセグメントに使われるエクステントに対して、デフォルトの記憶領域パラメータを指定できます。オブジェクトを作成するときに指定された記憶領域パラメータによって、そのオブジェクトが含まれている表領域のデフォルトの記憶領域パラメータが上書きされます。ただし、オブジェ

クトを作成するときに記憶領域パラメータを指定しないと、そのオブジェクトのセグメントは表領域に対して自動的にデフォルトの記憶領域パラメータを使います。

表領域のデフォルトの記憶領域パラメータは、その表領域が収容する代表的なオブジェクトのサイズを計算して設定してください（サイズを見積もってください）。例外的なオブジェクトに対する記憶領域パラメータは、そのオブジェクトを作成するときに指定できます。

注意： 新しい表領域にデフォルトの記憶領域パラメータを指定しないと、Oracle のデフォルトの記憶領域パラメータが、その表領域のデフォルトの記憶領域パラメータとなります。

関連項目： オブジェクトのサイズの見積もりについては、第 9 章～第 16 章を参照してください。

ユーザーに表領域割当て制限を割り当てる

表、クラスタ、スナップショット、索引、その他のオブジェクトを作ろうとするユーザーには、そのオブジェクトを作成するための権限と、そのオブジェクトのセグメントを保有する予定の表領域内の割当て制限（領域の許容または制限）を与えます。セキュリティ管理者には、オブジェクトを作成するのに必要な権限をデータベース・ユーザーに付与し、また必要に応じて、表領域の割当て制限をデータベース・ユーザーに割り当てる責任があります。

関連項目： 表領域の割当て制限をデータベース・ユーザーに割り当てる方法については、20-12 ページの「表領域の割当て制限を割り当てる」を参照してください。

表領域の作成

表領域を作成するための手順は、オペレーティング・システムによって異なります。ほとんどのオペレーティング・システムでは、新しい表領域を作成するとき、またはデータ・ファイルを加えて表領域を変更するときには、データ・ファイルのサイズと完全なファイル名を指定します。それぞれの場合に、Oracle は、指定されたとおりにデータ・ファイルを自動的に割り当てて、フォーマットします。しかし、オペレーティング・システムによっては、インストールの前に、データ・ファイルを作っておかなければなりません。

どのデータベースでも最初の表領域は常に SYSTEM 表領域です。そのため、データベースを作成するときには、データベースの最初のデータ・ファイルが、SYSTEM 表領域に自動的に割り当てられます。

次のような場合には、新しい表領域を作成できます。

- 対応するデータベースに、より多くのディスク記憶領域を割り当て、データベースを大きくしたい。
- 特定のタイプのデータを、他のデータベース・データから分離して格納するために、論理記憶構造を作成する必要がある。

データベースの合計サイズを大きくするために、新しい表領域を追加するのではなく、既存の表領域にデータ・ファイルを追加できます。

注意： 現在のインスタンスに対して少なくとも2つのロールバック・セグメントが (SYSTEM ロールバック・セグメントを含めて) 獲得されるまで、どのようなデータも新しい表領域に挿入できません。

新しい表領域を作成するには、Enterprise Manager/GUI の「表領域作成」プロパティ・シート、または SQL コマンド CREATE TABLESPACE を使います。表領域を作成するには、CREATE TABLESPACE システム権限が必要です。

次の例では、(データベースのロールバック・セグメントを保持するために) 次のような特性を持つ表領域 RB_SEGS を作ります。

- 新しい表領域のデータには、1つのデータ・ファイル (サイズは 50MB) が含まれる。
- この表領域に作られるすべてのセグメントに対してデフォルト記憶領域パラメータが、明示的に設定される。
- 表領域が作られた後、オフラインになる。

次の文では、表領域 RB_SEGS が作られます。

```
CREATE TABLESPACE rb_segs
  DATAFILE 'datafilers_1' SIZE 50M
  DEFAULT STORAGE (
    INITIAL 50K
    NEXT 50K
    MINEXTENTS 2
    MAXEXTENTS 50
    PCTINCREASE 0)
  OFFLINE;
```

表領域の作成時に、ファイル名を完全に指定しないと、データ・ファイルはデータベース・サーバーのデフォルト・ディレクトリに作られます。

関連項目： 表領域の初期作成の詳細は、使っているオペレーティング・システム固有の Oracle のマニュアルを参照してください。

データ・ファイルの追加方法の詳細は、9-4 ページの「データ・ファイルの作成、表領域への追加」を参照してください。

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

一時表領域を作成する

複数のソート操作の並行性の改善、オーバーヘッドの減少、Oracle の領域管理操作の完全な回避を実現する場合には、一時表領域を作成できます。

一時表領域では、あるインスタンスと表領域のソート操作は、すべて1つのソート・セグメントを共有します。ソート・セグメントは、その表領域の中でソート操作を実行するすべてのインスタンスに存在します。一時表領域に永久オブジェクトを格納できません。V\$SORT_SEGMENTS 表で、一時表領域のソート・セグメント内の領域の割当てと割当て解除を表示できます。

表領域を作成するときにその表領域を一時表領域として指定するには、次の文を発行します。

```
CREATE TABLESPACE tablespace TEMPORARY;
```

既存の表領域で表領域を一時表領域として指定するには、次の文を発行します。

```
ALTER TABLESPACE tablespace TEMPORARY;
```

注意：一時表領域をオフラインにできます。一時表領域をオンラインに戻しても、一時的状態には影響しません。

関連項目：CREATE TABLESPACE コマンドおよび ALTER TABLESPACE コマンドの詳細は、『Oracle8 Server SQL リファレンス』を参照してください。

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

Oracle の領域管理の詳細は、『Oracle8 Server 概要』を参照してください。

表領域割当ての管理

ここでは、表領域の割当ての管理について説明します。トピックは次のとおりです。

- 表領域に対する記憶領域設定を変更する
- 空き領域を合わせる

表領域に対する記憶領域設定を変更する

表領域のデフォルトの記憶領域パラメータを変更して、将来その表領域に作られるオブジェクトのデフォルトの指定を変更できます。後で表領域に作られるオブジェクトのために記憶領域パラメータのデフォルトを変更するには、Enterprise Manager/GUI の「表領域変更」プロパティ・シート、または SQL コマンド ALTER TABLESPACE を使います。表領域のデータ・ファイルの追加または表領域のデフォルト記憶領域パラメータの変更には、ALTER TABLESPACE システム権限が必要です。

```
ALTER TABLESPACE users
  DEFAULT STORAGE (
    INITIAL 50K
    NEXT 50K
```

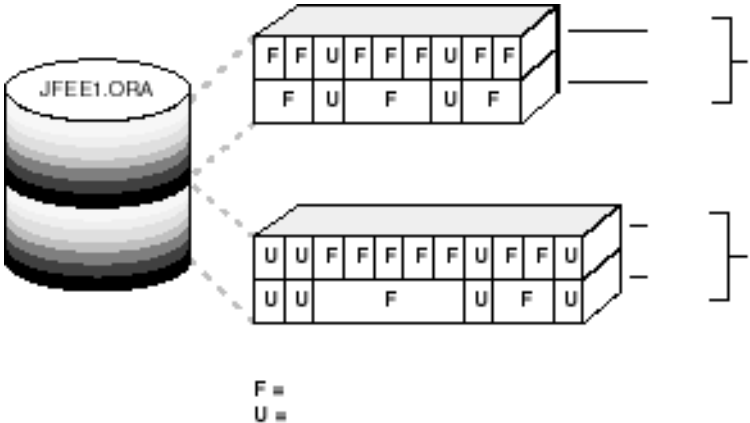
```
MINEXTENTS 2
MAXEXTENTS 20
PCTINCREASE 50);
```

表領域のデフォルト記憶領域パラメータの新しい値は、その表領域内のセグメントに対して割り当てられる、将来のエクステントにだけ影響を及ぼします。

空き領域を合わせる

表領域セグメントの領域は、特定の数の連続するデータ・ブロックから構成されるエクステントを使って管理されます。新しいエクステントを表領域セグメントに割り当てるときには、必要なエクステントに最も近いサイズの使用可能エクステントが使われます。したがって、大きな使用可能エクステントを分けたり、連続する小さな使用可能エクステントを合わせて1つの大きな空きエクステントにすることができます（図 8-1 を参照）。ただし、空き領域の割当ておよび割当て解除を連続して実行すると、表領域が分けられて、大きなエクステントの割当てが困難になります。デフォルトでは、SMON（システム監視）プロセスが表領域の使用可能エクステントをバックグラウンドで合わせます。必要に応じて、SMON が使用可能エクステントを合わせないようにもできます。

図 8-1 空き領域を合わせる



領域の断片化が激しい（ディスク上の連続した領域が連続していないように見える）場合には、1つの領域トランザクションで空き領域を合わせることができます。8回合わせるたびに領域トランザクションはコミットし、別のトランザクションが領域の割当てや割当て解除を実行できるようになります。表領域を合わせるには、ALTER TABLESPACE 権限を持っていないけません。次のコマンドを使うと、表領域ごとに、表領域内の使用可能なすべての空き領域を合わせて、連続する大きなエクステントにすることができます。

```
ALTER TABLESPACE tablespace COALESCE;
```

また、このコマンドを使って、SMON およびエクステント割当てを合わせる機能を補い、それによって、激しく断片化した表領域での領域割当てのパフォーマンスを向上させることもできます。このコマンドを発行しても、その表領域にアクセスしている他のユーザーのパフォーマンスには影響ありません。ALTER TABLESPACE コマンドのその他のオプションと同様に、COALESCE オプションも排他的であり、このオプションを指定するときは他のオプションは指定できません。

表領域についての情報を表示する

表領域について、合わせることのできるエクステントについての統計表示を表示するには、DBA_FREE_SPACE_COALESCED ビューを表示させます。特定の表領域で領域を合わせる必要があるかどうかを判断する場合に、このビューを問い合わせることができます。

関連項目：DBA_FREE_SPACE_COALESCED の内容の詳細は、『Oracle8 Server リファレンス・マニュアル』を参照してください。

表領域の可用性の変更

オフライン状態の表領域をオンラインにして、データベース・ユーザーがその表領域内のスキーマ・オブジェクトを使用できるようにすることができます。また、データベースをオープンしたままオンライン状態の表領域をオフラインにして、データベースのその部分だけを一時的に一般用途では使えないようにし、残りの部分をオープンのまま使用可能にしておくこともできます。この部分のトピックは次のとおりです。

- 表領域をオンラインにする
- 表領域をオフラインにする

表領域をオンラインにする

Oracle データベースがオープンされていれば、いつでもデータベース内の任意の表領域をオンラインにすることができます。ただし、データ・ディクショナリは常に Oracle で使える状態でなければならないので、SYSTEM 表領域は常にオンラインでなければなりません。通常、表領域は、データベース・ユーザーがその中のデータを使えるようにオンラインになっています。

データベースがオープンした状態でオフラインの表領域をオンラインにするには、Enterprise Manager/GUI の「オンラインにする」メニュー項目、または SQL コマンド ALTER TABLESPACE を使います。表領域をオンラインにするには、MANAGE TABLESPACE システム権限が必要です。

注意： オンラインにしようとする表領域が、「明らかに」(ALTER TABLESPACE OFFLINE コマンドの NORMAL オプションを使って) オフラインになっていない場合、最初にメディア回復をしない限りオンラインにできません。メディア回復をしないと、エラーが戻されて表領域はオフラインのままになります。

次の文は、USERS 表領域のオンラインの終了をマークします。

```
ALTER TABLESPACE users OFFLINE;
```

表領域をオフラインにする

表領域は、次のような理由の場合はオフラインにできます。

- データベースの一部だけを使えないようにし、残りの部分には正常にアクセスできるようにしたい。
- オフラインの表領域のバックアップを実行したい（ただし、表領域はオンラインで使用中の場合はバックアップ可能）。
- アプリケーションの更新時またはメンテナンス時にはアプリケーションとその表のグループを一時的に使えないようにしたい。

データベースがオープンした状態でオンラインの表領域をオフラインにするには、Enterprise Manager/GUI の「オフラインにする」メニュー項目、または SQL コマンド ALTER TABLESPACE を使います。表領域をオフラインにするには、MANAGE TABLESPACE システム権限が必要です。

表領域をオフラインにするときには、次の優先順位のどれかを指定できます。

通常のアフライン

表領域のどのデータ・ファイルにもエラー条件が存在していない場合は、この表領域を通常の方法でオフラインにできます。書込みエラーが発生していると、現時点では表領域のデータ・ファイルをオフラインにできません。通常のアフラインの優先順位では、Oracle は表領域のすべてのデータ・ファイルのチェックポイントを取って、データ・ファイルをオフラインにします。

一時オフライン

表領域の 1 つまたは複数のデータ・ファイルについてエラー条件が存在している場合でも、表領域を一時的にオフラインにできます。一時オフラインの優先順位では、Oracle はまだオフラインになっていないデータ・ファイルのチェックポイントを取って、これらのファイルをオフラインにします。

通常のアフライン

表領域のどのデータ・ファイルにもエラー条件が存在していない場合は、この表領域を通常の方法でオフラインにできます。書込みエラーが発生していると、現時点では表領域のデータ・ファイルをオフラインにできません。通常のアフラインの優先順位では、Oracle は表領域のすべてのデータ・ファイルのチェックポイントを取って、データ・ファイルをオフラインにします。

オフラインになっているファイルがないときに表領域を一時的にオフラインにする場合は、表領域をオンラインに戻す前にメディア回復をする必要はありません。ただし、表領域の1つまたは複数のファイルが書込みエラーのためにオフラインになっており、この表領域を一時的にオフラインにする場合は、表領域をオンラインに戻す前に回復をする必要があります。

即時オフライン

表領域を即時にオフラインにする場合は、Oracle がデータ・ファイルのチェックポイントを取る必要はありません。即時オフラインの優先順位では、表領域をオンラインに戻す前に、表領域のメディア回復が必要となります。データベースを NOARCHIVELOG モードで運用している場合、表領域を即時にオフラインにはできません。

警告： 表領域をオフラインにしなければならない場合、できれば通常のアフライン(デフォルト)を使用します。そうすれば、表領域をオフラインにしてからオンラインに戻すまでに、REDO ログ順序が(不完全メディア回復後に ALTER DATABASE OPEN RESETLOGS 文を使って)リセットされたとしても、表領域が通常どおりにオフラインであれば、表領域の回復を必要とすることなく、オンラインにできることが保証されます。

通常どおりにオフラインにできない場合に限り、表領域を一時的にオフラインにします。この場合、エラーが原因でオフラインにされたファイルだけを、表領域をオンラインにする前に回復する必要があります。通常アフラインと一時アフラインの両方でオフラインにできなかった後に限り、即時アフラインで表領域をオフラインにしてください。

次の例は、USERS 表領域を通常の方法でオフラインにします。

```
ALTER TABLESPACE users OFFLINE NORMAL;
```

関連項目：オンラインの表領域をオフラインにする前に、その表領域にアクティブなロールバック・セグメントが含まれていないことを確認してください。詳細は、18-12 ページの「ロールバック・セグメントをオフラインにする」を参照してください。

表領域を読み込み専用にする方法

ここでは、表領域を読み込み専用にする方法について説明します。トピックは次のとおりです。

- 前提条件
- 読み専用表領域を書込み可能にする
- WORM デバイスに読み専用表領域を作成する

表領域を読み込み専用にすると、それ以降は表領域のデータ・ファイルに対して書き込み操作ができなくなります。表領域を読み込み専用にしてから、その表領域をバックアップしてください。

SQL コマンド ALTER TABLESPACE を使って、表領域を読み込み専用に変更します。表領域を読み込み専用にするには、ALTER TABLESPACE システム権限が必要です。次の文は、FLIGHTS 表領域を読み込み専用にします。

```
ALTER TABLESPACE flights READ ONLY
```

表領域を読み込み専用にした後は、そのファイルを読み込み専用メディアにコピーできます。次に、SQL コマンド ALTER DATABASE RENAME を使って制御ファイル内のデータ・ファイルを改名し、新しい位置を指し示すようにしてください。

読み専用の表領域はオンラインでもオフラインでもありません。ONLINE または OFFLINE オプションを指定して ALTER TABLESPACE コマンドを発行しても、表領域の読み専用の状態は変更されません。このコマンドを実行すると、表領域内のすべてのデータ・ファイルがオンラインまたはオフラインになります。

前提条件

表領域を読み込み専用にするには、あらかじめ次の条件を満たしていなければなりません。これらの制限条件を満たすには、この機能を制限モードで実行して、RESTRICTED SESSION システム権限を持つユーザーだけがログインできるようにするのが最も簡単な方法です。

- 表領域がオンラインになっている。
- データベース内にアクティブ・トランザクションがない。
- 表領域にはアクティブなロールバック・セグメントが入っていない。

表領域に適用する必要がある取消し情報がないことを確認する必要がある。

したがって、SYSTEM 表領域には SYSTEM ロールバック・セグメントが含まれているため、SYSTEM 表領域は読み専用にできない。さらに、読み専用表領域のロール

バック・セグメントにはアクセスできないため、表領域を読み込み専用にする前にロールバック・セグメントを削除しておいた方がよい。

- バックアップの終了によって表領域内のすべてのデータ・ファイルのヘッダー・ファイルが更新されてしまうため、表領域がオンライン・バックアップに含まれていてはならない。
- 初期化パラメータ COMPATIBLE は 7.1.0 以上に設定してある。

読み込み専用表領域のデータにアクセスする際のパフォーマンスを向上させるためには、表領域を読み込み専用にする直前に、表領域の表のすべてのブロックにアクセスする問合せを発行しておくといよいでしょう。SELECT COUNT (*) などの単一の問合せを各表に対して実行しておく、それ以降、表領域のデータ・ブロックに最も効果的にアクセスできるようになります。つまり、これにより、最後にブロックを修正したトランザクションの状態を Oracle が確認する必要がなくなるからです。

警告： 読み込み専用表領域のデータ・ファイルの改名とサイズ変更はできません。

関連項目： 読み込み専用表領域の詳細は、『Oracle8 Server 概要』を参照してください。

読み込み専用表領域を書込み可能にする

表領域は、作成時には常に読み書き可能になっています。読み込み専用の表領域を読み書き可能な表領域に戻すには、SQL コマンド ALTER TABLESPACE を使います。読み込み専用の表領域を書込み可能に変更するには、ALTER TABLESPACE システム権限を持っていなければなりません。次のコマンドは、FLIGHTS 表領域を書込み可能にします。

```
ALTER TABLESPACE flights READ WRITE;
```

読み込み専用の表領域を書込み可能に変更すると、データ・ファイルの制御ファイルが更新されるので、読み込み専用バージョンのデータ・ファイルを回復の開始点として使用できます。

前提条件

このコマンドを発行するには、表領域内のすべてのデータ・ファイルがオンラインになっていなければなりません。データ・ファイルをオンラインにするには、ALTER DATABASE コマンドの DATAFILE ONLINE オプションを使います。V\$DATAFILE ビューはデータ・ファイルの現行の状態をリストします。

WORM デバイスに読み込み専用表領域を作成する

更新する必要のない読み込み専用ファイルがあるときは、WORM（追記型）デバイスに読み込み専用表領域を作成するとよいでしょう。

1. 別のデバイスに書き込み可能表領域を作成する。その表領域に属するオブジェクトを作成して、データを挿入する。

2. READ ONLY オプションを指定した ALTER TABLESPACE コマンドを発行して、表領域を読み込み専用に変更する。
3. 表領域のデータ・ファイルを WORM デバイスにコピーする。ファイルをコピーするには、オペレーティング・システムのコマンドを使います。
4. 表領域をオフラインにする。
5. データ・ファイルの名前が、WORM デバイスにコピーしたファイルと一致するように改名する。データ・ファイルを改名すると、制御ファイルにあるこれらのファイルの名前も変更されます。
6. 表領域をオンラインにする。

表領域の削除

表領域とその内容が必要ではなくなったと判断される場合には、その表領域と内容（表領域に含まれるセグメント）をデータベースから削除できます。Oracle のデータベースの表領域はすべて削除できます（ただし SYSTEM 表領域は除く）。表領域を削除するには、DROP TABLESPACE システム権限が必要です。

警告： 表領域が削除されると、表領域のデータを回復できません。そのため、削除しようとしている表領域に含まれているデータはすべて、将来的に必要なないことを確かめてください。また、表領域をデータベースから削除する前後では、ただちにデータベースを完全にバックアップする必要があります。表領域を間違えて削除した場合、または表領域が削除された後にデータベースが将来問題を起こした場合、データベースを回復できるように、バックアップを作成することを強くお勧めします。

表領域を削除すると、対応付けられたデータベースの制御ファイル中のファイル・ポインタだけが削除されます。削除された表領域を構成していたデータ・ファイルは削除されません。使われていたディスク領域を解放するには、オペレーティング・システムの適切なコマンドを使って、削除した表領域のデータ・ファイルを物理的に削除します。

削除しようとする表領域に、アクティブ・セグメントが存在することはできません。たとえば、表領域内の表（つまり、データ・セグメント）が現在使われている場合、またはアクティブ・ロールバック・セグメントが表領域に含まれている場合、その表領域を削除できません。簡単にするために、削除する前に表領域をオフラインにしてください。

表領域が削除された後、表領域のエントリは、データ・ディクショナリ（たとえば、DBA_TABLESPACE ビュー）に残っていますが、表領域の状態は INVALID（無効）に変更されます。

表領域を削除するには、Enterprise Manager/GUI の「表領域削除」メニュー項目、または SQL コマンド DROP TABLESPACE を使います。次の文は、USERS 表領域を、その中のセグメントも含めて削除します。

```
DROP TABLESPACE users INCLUDING CONTENTS;
```

表領域が空の場合（つまり、表、ビューなど何も含まない）その表領域を削除するために、「含まれているオブジェクトの削除」チェックボックスをチェックする必要はありません。しかし、表領域がなんらかのデータを含む場合には、表領域を削除するために、「含まれているオブジェクトの削除」チェックボックスをチェックする必要があります。さらに、他の表領域内の表の外部キーによって参照される主キー、または一意キーを持つ表が、表領域に含まれている場合、子表の FOREIGN KEY 制約の削除をカスケードする場合には、表領域を削除するために、「整合性制約により関連のあるオブジェクト削除」チェックボックスを選択してください。

子表の FOREIGN KEY 制約の削除をカスケードするには、CASCADE CONSTRAINTS オプションを使ってください。

関連項目：表領域をオフラインにする操作の詳細は 8-8 ページの「表領域をオフラインにする」を参照してください。

DROP TABLESPACE 文の詳細は、『Oracle8 Server SQL リファレンス』を参照してください。

表領域についての情報の表示

次のデータ・ディクショナリ・ビューは、データベースの表領域に関して有益な情報を提供します。

- USER_EXTENTS、DBA_EXTENTS
- USER_SEGMENTS、DBA_SEGMENTS
- USER_FREE_SPACE、DBA_FREE_SPACE
- DBA_USERS
- DBA_TS_QUOTAS
- USER_TABLESPACES、DBA_TABLESPACES
- DBA_DATA_FILES
- V\$DATAFILE

このマニュアルの他の章には記述されていないビューの使用方法を次の例で説明します。データベースには、2つの表領域 SYSTEM と USERS が含まれています。USERS は2つのファイル、FILE1(100MB) と FILE2(200MB) から構成されます。表領域は通常のアフライン状態になっています。

表領域とデフォルト記憶領域パラメータを記述する例

データベース内の表領域の名前とデフォルト記憶領域パラメータをすべて記述するには、DBA_TABLESPACES ビューに対して次の問合せを使ってください。

```
SELECT tablespace_name "TABLESPACE",
       initial_extent "INITIAL_EXT",
       next_extent "NEXT_EXT",
       min_extents "MIN_EXT",
       max_extents "MAX_EXT",
       pct_increase
FROM sys.dba_tablespaces;
```

TABLESPACE	INITIAL_EXT	NEXT_EXT	MIN_EXT	MAX_EXT	PCT_INCREASE
-----	-----	-----	-----	-----	-----
SYSTEM	10240000	10240000	1	99	50
USERS	10240000	10240000	1	99	50

データ・ファイルとデータベースの対応する表領域を記述する例

データ・ファイルの名前、およびサイズ、データベースの対応する表領域を記述するには、DBA_DATA_FILES ビューに対して次の問合せを入力してください。

```
SELECT file_name, bytes, tablespace_name
FROM sys.dba_data_files;
```

FILE_NAME	BYTES	TABLESPACE_NAME
-----	-----	-----
filename1	10240000	SYSTEM
filename2	10240000	USERS
filename3	20480000	USERS

各表領域の空き領域（エクステント）を記述する例

データベース内の各表領域の空きエクステント内の利用可能な領域の容量に関する情報を記述するには、次の問合せを入力してください。

```
SELECT tablespace_name, file_id,
       COUNT(*) "PIECES",
       MAX(blocks) "MAXIMUM",
       MIN(blocks) "MINIMUM",
       AVG(blocks) "AVERAGE",
       SUM(blocks) "TOTAL"
FROM sys.dba_free_space
WHERE tablespace_name = 'SYSTEM'
GROUP BY tablespace_name, file_id;
```

TABLESPACE	FILE_ID	PIECES	MAXIMUM	MINIMUM	AVERAGE	SUM
-----	-----	-----	-----	-----	-----	-----
SYSTEM	1	2	2928	115	1521.5	3043

SUM は各表領域の使用可能領域の総量、PIECES は表領域のデータ・ファイル内の断片化の総量、MAXIMUM は最大の連続領域を示します。新しいオブジェクトを作ろうとしているとき、またはセグメントが拡張しようとしていることが分かっていて、表領域に十分な領域があることを確かめたいときに、この問合せが役に立ちます。

データ・ファイルの管理

この章では、データ・ファイルの管理について説明します。トピックは次のとおりです。

- データ・ファイルを管理するためのガイドライン
- データ・ファイルの作成、表領域への追加
- データ・ファイルのサイズを変更
- データ・ファイルの可用性の変更
- データ・ファイルの改名、再配置
- データ・ファイル内のデータ・ブロックの検証
- データ・ファイルの情報の表示

関連項目：この章には、Oracle Enterprise Manager について述べている箇所がいくつかあります。Enterprise Manager/GUI または Server Manager/LineMode を使って特定のタスクを実行する方法の詳細は、『Oracle Enterprise Manager 管理者ガイド』および『Oracle Server Manager ユーザーズ・ガイド』を参照してください。

データ・ファイルは、メディア障害時にデータベース回復処理の一部として作られることもあります。詳細は、『Oracle8 Server バックアップおよびリカバリ』を参照してください。

データ・ファイルを管理するためのガイドライン

ここでは、データ・ファイルの管理について説明します。トピックは次のとおりです。

- データ・ファイルの数
- データ・ファイルのサイズ設定
- 適切なデータ・ファイルの配置
- REDO ログ・ファイルと分離したデータ・ファイルの格納

すべてのデータ・ファイルには、「絶対ファイル番号」と「相対ファイル番号」の2つのファイル番号が対応付けられています。

絶対ファイル番号は、データベース内のデータ・ファイルを固有に識別するものです。Oracle8 より前のバージョンでは、絶対ファイル番号が単に「ファイル番号」と呼ばれていました。

相対ファイル番号は、データ・ファイルを表領域内で固有に識別するものです。多くの場合、小規模および中規模のサイズのデータベースでは、相対ファイル番号と絶対ファイル番号とは同じです。しかし、データベース内のデータ・ファイル数が一定のしきい値（通常は1023）を超えると、相対ファイル番号と絶対ファイル番号が違ってきます。相対ファイル番号は、多くのデータ・ディクショナリ・ビューで使われています。

データ・ファイルの数

データベースの SYSTEM 表領域に対しては、データ・ファイルが最低1つ必要です。小規模システムでは、データ・ファイルを1つだけ持つこともあります。一般に、小さなファイルを数多く作るよりも、大きなファイルを数少なく作って、同時にオープンする必要のあるファイルを少なくすることを推奨します。

オペレーティング・システム固有の制限に応じて、表領域にデータ・ファイルを追加することもできます。

オペレーティング・システムの制限

各オペレーティング・システムには、プロセスあたりのオープン・ファイルの最大数に制限があります。他の制限にかかわらず、オープンしているファイル数がオペレーティング・システム制限に達すると、データ・ファイルを作成することができません。

Oracle システムの制限

Oracle では、インスタンスによってオープンされる Oracle データベースのデータ・ファイルの最大数が制限されます。この制限はポートによって異なります。

制御ファイルの上限

CREATE DATABASE 文または CREATE CONTROLFILE 文を発行するとき、MAXDATAFILES パラメータによって制御ファイルのデータ・ファイル部分の初期サイズが指定されます。あとでファイルを追加したとき、そのファイルの番号が MAXDATAFILES より大きく、かつ DB_FILES 以下になるなら、制御ファイルは、データ・ファイル部分にもっと多くのファイルが入るように自動的に拡張されます。

インスタンスまたは SGA の上限

Oracle インスタンスを起動すると、データベースのパラメータ・ファイルがデータ・ファイルの情報用に予約されている SGA 領域の容量を示します。つまり、データ・ファイルの最大数は DB_FILES パラメータによって制御されます。この制限が適用されるのは、インスタンスが存続する期間だけです。

注意：DB_FILES のデフォルト値はオペレーティング・システムによって違います。

Oracle Parallel Server では、すべてのインスタンスのデータ・ファイルの上限は同じ値に設定されていなければなりません。

DB_FILES の値を決めるときは、次のことを考慮してください。

- DB_FILES の値が低すぎると、DB_FILES 制限を超えてデータ・ファイルを追加する場合に、まずデータベースを停止させることが必要になります。
- DB_FILES の値が多すぎると、メモリーが不必要に消費されてしまいます。

理論上、Oracle データベースが処理できるデータ・ファイルの数には制限がありません。しかし、データ・ファイルの数を決めるときは、次のことを考慮してください。

- 小さいデータ・ファイルをたくさん処理するより、データ・ファイルの数を少なくしたほうがフォーマットが向上します。さらに、ファイルが大きいほうが回復可能単位をきめ細かく設定できます。
- ほとんどの場合、オペレーティング・システムでは、1つのプロセスで同時にオープンできるファイルの数に制限があります。Oracle の DBW0 プロセスは、すべてのオンライン・データ・ファイルをオープンできます。さらに、Oracle には、オープン・ファイル記述子をキャッシュとして処理し、オープン・ファイル記述子の数がオペレーティング・システムで定義されている制限に達したときに自動的にファイルをクローズする機能もあります。

Oracle のデータベース内のデータ・ファイルの数は、オペレーティング・システムで定義されている制限より大きくすることができます。その場合、パフォーマンスは落ちることがあります。できれば、オープン・ファイル記述子に関するオペレーティング・システムの制限

を調整して、それがデータベース内のオンライン・データ・ファイルの数より大きくなるようにしてください。

1 つの表領域の中のデータ・ファイルの最大数に関するオペレーティング・システム固有の制限は、ほとんどの場合、1023 ファイルです。

関連項目：オペレーティング・システムの制限の詳細は、オペレーティング・システム固有の Oracle のマニュアルを参照してください。

Parallel Server のオペレーティング・システムの制限の詳細は、『Oracle8 Parallel Server 概要および管理』を参照してください。

MAXDATAFILES の詳細は、『Oracle8 Server SQL リファレンス』を参照してください。

データ・ファイルのサイズ設定

最初の（オリジナルの SYSTEM 表領域内の）データ・ファイルの大きさは、初期のデータ・ディクショナリとロールバック・セグメントを収容するために、最低でも 7M 必要です。他の Oracle 製品をインストールする場合は、これらの製品には SYSTEM 表領域内の領域が（たとえば、オンライン・ヘルプ用の領域など）さらに必要となります。これらの製品の詳細は、インストールの指示を参照してください。

適切なデータ・ファイルの配置

表領域の位置は、その表領域を構成するデータ・ファイルの物理的な位置によって決定されます。コンピュータのハードウェア資源を適切に使ってください。

たとえば、データベースを格納するために利用できるディスク・ドライブがいくつかある場合には、1 つのディスク・ドライブ上の表領域に表データを格納し、別のディスク・ドライブ上の表領域に索引データを格納すると有効です。このようにユーザーが表情報を問い合わせることにより、両方のディスク・ドライブが同時に作動して、表データと索引データの両方を同時に検索し、データベース・システムのパフォーマンスを改善できます。

REDO ログ・ファイルと分離したデータ・ファイルの格納

データ・ファイルは、データベースの REDO ログ・ファイルが格納されているディスク・ドライブに格納してはなりません。データ・ファイルと REDO ログ・ファイルが同じディスク・ドライブに格納されていて、このディスク・ドライブで障害が発生すると、これらのファイルをデータベースの回復手順で使えなくなります。

REDO ログ・ファイルを多重化すると、全 REDO ログ・ファイルが失われる可能性が低くなるので、データ・ファイルを一部の REDO ログ・ファイルと同じドライブに格納できます。

データ・ファイルの作成、表領域への追加

表領域に割り当てられたディスク領域のすべての容量（その結果としてデータベース）を大きくするために、データ・ファイルを作って表領域に追加できます。

表領域を作るときにデータベース管理者がデータベース・オブジェクトの潜在的なサイズを見積もって、十分な数のファイルまたはデバイスを追加しておくことが理想です。そうすれば、データがすべてのデバイスに均等に分散します。

表領域にデータ・ファイルを追加するには、以下のいずれかを実行します。Oracle Enterprise Manager の Oracle Storage Manager で、「データベース」メニューで「作成」を選択し、「表領域」メニューで「データ・ファイルを追加」を選択する。

SQL コマンドで、ALTER TABLESPACE を実行します。データ・ファイルを表領域に追加するには、ALTER TABLESPACE システム権限が必要です。

次の文では、RB_SEGS 表領域の新しいデータ・ファイルが作られます。

```
ALTER TABLESPACE rb_segs  
  ADD DATAFILE 'filename1' SIZE 1M;
```

表領域に新しいデータ・ファイルを追加して、ファイル名を完全に指定しないと、データ・ファイルはデータベース・サーバーのデフォルト・ディレクトリに作られます。既存のファイルを再使用しない場合は、必ず他のファイルと重複しないファイル名を指定してください。

データ・ファイルのサイズを変更

ここでは、データ・ファイルのサイズを変更する様々な方法について説明します。トピックは次のとおりです。

- データ・ファイルの自動拡張機能を使用可能および使用禁止にする
- 手動でデータ・ファイルのサイズを変更する

データ・ファイルの自動拡張機能を使用可能および使用禁止にする

データベースに領域がさらに必要になった場合に自動的にサイズを大きくできるようにデータ・ファイルを作ったり、既存のデータ・ファイルを変更したりできます。ファイルのサイズは、指定されている最大値に達するまで、指定の増分値で大きくなります。

データ・ファイルを自動的に拡張するよう設定しておくことには、次のような利点があります。

- 表領域で領域が足りなくなった場合に即時中断をする必要性が減る。
- エクステンツの割当て失敗が原因でアプリケーションが停止することがなくなる。

データ・ファイルが自動的に拡張できるかどうかを知るには、DBA_DATA_FILES ビューに問合せ、AUTOEXTENSIBLE 列を調べます。

次の SQL コマンドを使って、データ・ファイルを作るときに自動ファイル拡張機能を指定できます。

- CREATE DATABASE

- CREATE TABLESPACE
- ALTER TABLESPACE

既存のデータ・ファイルの自動ファイル拡張機能を使用可能または使用禁止にしたり、データ・ファイルのサイズを手動で変更するには、SQL コマンド ALTER DATABASE を使います。

次の例は、USERS 表領域に追加されたデータ・ファイル FILENAME2 の自動拡張機能を使用可能にします。

```
ALTER TABLESPACE users
  ADD DATAFILE 'filename2' SIZE 10M
  AUTOEXTEND ON
  NEXT 512K
  MAXSIZE 250M
```

NEXT の値は、データ・ファイルの拡張時にこのファイルに追加される増分値の最小サイズです。MAXSIZE の値は、自動拡張可能なファイルの最大サイズです。

次の例は、データ・ファイル FILENAME2 の自動拡張機能を使用禁止にします。

```
ALTER DATABASE DATAFILE 'filename2'
  AUTOEXTEND OFF
```

関連項目：データ・ファイルの作成または変更のための SQL コマンドの詳細は、『Oracle8 Server SQL リファレンス』を参照してください。

手動でデータ・ファイルのサイズを変更する

手動でデータ・ファイルのサイズを増減させるには、ALTER DATABASE コマンドを使います。

データ・ファイルのサイズを変更できるため、データ・ファイルをさらに追加しなくてもデータベースに領域を追加できます。データベース内で許容されているデータ・ファイルの最大数に達することが懸念される場合に、このコマンドが有効です。

また、データ・ファイルのサイズを手動で小さくして、データベース内の未使用の領域の再生もできます。これは必要な領域の容量見積もりを誤った場合にこれを訂正するとき、有効です。

この例では、データ・ファイル FILENAME2 が 250MB まで拡張されていることを想定しています。ただし、その表領域には現在小さなオブジェクトが格納されているので、データ・ファイルのサイズを小さくできます。

次のコマンドは、データ・ファイル FILENAME2 のサイズを小さくします。

```
ALTER DATABASE DATAFILE 'filename2'
  RESIZE 100M
```

注意：常にファイルのサイズを指定した値まで小さくできるわけではありません。

関連項目：ダウングレードに対してリサイジング・ファイルが持つ意味の詳細は、『Oracle8 Server 移行ガイド』を参照してください。

ALTER DATABASE コマンドの詳細は、『Oracle8 Server リファレンス・マニュアル』を参照してください。

データ・ファイルの可用性の変更

ここでは、データ・ファイルの可用性を変更するさまざまな方法について説明します。トピックは次のとおりです。

- ARCHIVELOG モードでデータ・ファイルをオンラインにする
- NOARCHIVELOG モードでデータ・ファイルをオフラインにする

非常にまれな状況においては、特定のデータ・ファイルをオンライン（利用できる状態）にしたり、特定のファイルをオフライン（利用できない状態）にしたりする必要があるかもしれません。たとえば、データ・ファイルへの書込みに問題がある場合、Oracle はそのデータ・ファイルを自動的にオフラインにすることができます。このような場合メディア回復の前に、破損したデータ・ファイルを手動でオンラインにしたり、オフラインにしたりする必要があるかもしれません。

注意：表領域をオフラインにすることによって、表領域内のすべてのデータ・ファイル（SYSTEM 表領域内のファイル以外）を一時的に使えないようにすることができます。表領域をオンラインに戻すために、表領域内のファイルはそのままにしておかなければなりません。

オフライン・データ・ファイルにはアクセスできません。読み専用表領域のデータ・ファイルをオンラインにすると、そのデータ・ファイルは読み可能になります。このファイルに対応付けられた表領域が読み書き可能な状態に戻らない限り、このファイルには書き込めません。読み専用表領域のファイルを個別にオンラインまたはオフラインにするには、ALTER DATABASE コマンドの DATAFILE オプションを使います。

データ・ファイルをオンラインにしたり、オフラインにしたりするには、どちらのアーカイブ・モードであっても、ALTER DATABASE システム権限を持っていなければなりません。データベースが排他モードでオープンされているときに限り、これらの操作を実行できます。

ARCHIVELOG モードでデータ・ファイルをオンラインにする

個々のデータ・ファイルをオンラインにするには、SQL コマンド ALTER DATABASE を発行し、DATAFILE パラメータを入力します。

注意： ALTER DATABASE コマンドのこのオプションを使うには、データベースは ARCHIVELOG モードでなければなりません。一方、NOARCHIVELOG モードでデータ・ファイルをオフラインにするとファイルが失われる可能性があります。このモードでは、データ・ファイルの不慮の損害を防止できます。

次の文は、指定したデータ・ファイルをオンラインにします。

```
ALTER DATABASE DATAFILE 'filename' ONLINE;
```

関連項目：メディア回復時にデータ・ファイルをオンラインにする方法の詳細は、『Oracle8 Server バックアップおよびリカバリ』を参照してください。

NOARCHIVELOG モードでデータ・ファイルをオフラインにする

データベースが NOARCHIVELOG モードのときにデータ・ファイルをオフラインにするには、DATAFILE パラメータと OFFLINE DROP オプション付きの ALTER DATABASE コマンドを使います。これによって、ただちにデータ・ファイルをオフラインにし、削除できます。たとえば、データベースは NOARCHIVELOG モードで運用されていて、データ・ファイルが、一時セグメントからのデータだけを含み、バックアップされていない場合に便利です。

次の文は、指定したデータ・ファイルをオフラインにします。

```
ALTER DATABASE DATAFILE 'filename' OFFLINE DROP;
```

データ・ファイルの改名、再配置

ここでは、データ・ファイルの改名と再配置について説明します。トピックは次のとおりです。

- 単一の表領域のデータ・ファイルを改名、再配置する
- 複数の表領域のデータ・ファイルを改名、再配置する

データ・ファイルを改名して、それらの名前や位置を変更できます。Oracle には、次のような変更のためのオプションが用意されています。

- データベースの残りの部分をオープンしたまま、単一のオフライン表領域内のデータ・ファイル（たとえば、TBSPACE1 内の FILENAME1 と FILENAME2）を改名、再配置する。

- データベースをマウントしても、クローズしたままの状態、複数の表領域内のデータ・ファイル（たとえば、TBSP1 内の FILE1 と TBSP2 内の FILE2）を同時に改名、再配置する。

注意： SYSTEM 表領域のデータ・ファイルを改名または再配置する場合、SYSTEM 表領域はオフラインにできないため、上記の后者のオプションを使わなければなりません。

これらの手順によるデータ・ファイルの改名と再配置は、データベースの制御ファイルに記録される、データ・ファイルへのポインタだけを変更します。オペレーティング・システム・ファイルを物理的に改名したり、オペレーティング・システム・レベルでファイルをコピーしたりするわけではありません。そのために、データ・ファイルの改名と再配置には、複数のステップが必要になります。この手順を実行する前に、ステップと例題をよく理解してください。

単一表領域のデータ・ファイルを改名するには、ALTER TABLESPACE システム権限が必要です。

単一の表領域のデータ・ファイルを改名、再配置する

単一の表領域のデータ・ファイルを改名、再配置する手順

1. データ・ファイルを含む SYSTEM 表領域以外の表領域をオフラインにする。
2. オペレーティング・システムを使って、データ・ファイルを新しい位置または新しい名前にコピーする。
3. 指定された新しい完全なファイル名が、古いファイル名とは異なることを確認する。
4. SQL コマンドで、ALTER TABLESPACE の RENAME DATAFILE オプションを使って、データベース内のファイル名を変更する。

たとえば、次の文は、データ・ファイル FILENAME1 と FILENAME2 を FILENAME3 と FILENAME4 にそれぞれ改名します。

```
ALTER TABLESPACE users
  RENAME DATAFILE 'filename1', 'filename2'
  TO 'filename3', 'filename4';
```

新しいファイルがすでに存在している必要があります。このコマンドではファイルは作られません。また、古いデータ・ファイルと新しいデータ・ファイルを正しく識別するために、必ず完全なファイル名を指定してください。特に、古いファイル名は、データ・ディクショナリの DBA_DATA_FILE ビューに表示されるとおり、正確に指定してください。

複数の表領域のデータ・ファイルを改名、再配置する

1 つ以上の表領域のデータ・ファイルは、RENAME FILE オプションを指定した SQL コマンド ALTER DATABASE を使って、改名、再配置できます。一回の操作で、複数の表領域のデータ・ファイルを改名、再配置する場合、または SYSTEM 表領域のデータ・ファイルを改名、再配置する場合、このオプションを使う以外に方法はありません。データベースをオープンしておかなければならない場合、前ページで説明した手順を検討してください。

一回の操作で、複数の表領域のデータ・ファイルを改名する、または SYSTEM 表領域のデータ・ファイルを改名するには、ALTER DATABASE システム権限を持っていなければなりません。

1. データベースがマウントされ、クローズされていることを確認する。
2. オペレーティング・システムを使って、データ・ファイルを新しい位置と新しい名前にコピーする。
3. 新しくコピーしたデータ・ファイル名が、現在使っているデータ・ファイルと違う、指定された完全なファイル名になっていることを確認する。
4. SQL コマンド ALTER DATABASE を使って、データベースの制御ファイル内のファイル・ポインタを改名する。

たとえば、次の文は、データ・ファイル FILENAME1 と FILENAME2 を FILENAME3 と FILENAME4 にそれぞれ改名します。

```
ALTER DATABASE
  RENAME FILE 'filename1', 'filename2'
  TO 'filename3', 'filename4';
```

新しいファイルがすでに存在している必要があります。このコマンドではファイルは作られません。また、古いデータ・ファイルと新しいデータ・ファイルを正しく識別するために、必ず完全なファイル名を指定してください。特に、古いファイル名は、データ・ディクショナリの DBA_DATA_FILE ビューに表示されるとおり、正確に指定してください。

データ・ファイルを再配置する例

次のような条件を想定します。

- オープンしているデータベースには、USERS という表領域が存在し、コンピュータの同じディスク上に位置する 1 つ以上のデータ・ファイルによって構成されている。
- USERS 表領域のデータ・ファイルが別のディスク・ドライブに再配置される。
- 現在 Server Manager/Line Mode を使用しており、管理者権限でオープン・データベースに接続している。

データ・ファイルを再配置する方法

1. 対象のデータ・ファイル名を確認する。

次のようにデータ・ディクショナリ・ビュー DBA_DATA_FILES を問い合わせると、USERS 表領域のデータ・ファイル名とそれぞれのサイズ（バイト単位）が記述されます。

```
SELECT file_name, bytes FROM sys.dba_data_files
       WHERE tablespace_name = 'USERS';
FILE_NAME          BYTES
-----
FILENAME1          102400000
FILENAME2          102400000
```

FILENAME1 と FILENAME2 は完全に指定した 2 つのファイル名であり、それぞれのサイズは 1MB です。

2. データベースをバックアップする。

1 つ以上の表領域のデータ・ファイルを改名、再配置するなど、データベースに対する構造上の変更を実行する前に、この操作中なんらかの問題が発生するのに備えて、データベースを完全にバックアップしておくことが非常に重要です。

3. データ・ファイルが入っている表領域をオフラインにするか、またはデータベースを停止してから再起動し、マウントしてクローズしたままにする。どちらの場合でも、表領域のデータ・ファイルはクローズされます。
4. オペレーティング・システム・コマンドを使って、データ・ファイルを新しい位置にコピーする。たとえば、既存のファイル FILENAME1 と FILENAME2 を、それぞれ FILENAME3 と FILENAME4 にコピーします。

注意： オペレーティング・システム・コマンドを実行すると、HOST コマンドを使って既存の Server Manager/LineMode を終了せずにファイルをコピーできます。

5. Oracle 内のデータ・ファイルを改名する。

USERS 表領域を構成するファイルに対するデータ・ファイル・ポインタは、対応付けられているデータベースの制御ファイルに記録されていますが、これらのポインタをこの時点で FILENAME1 と FILENAME2 から FILENAME3 と FILENAME4 にそれぞれ変更してください。

表領域がオフラインであるがデータベースがオープンしている場合は、SQL コマンドで ALTER TABLESPACE...RENAME DATAFILE コマンドを使います。データベースがマウントされているがクローズしている場合は、ALTER DATABASE...RENAME FILE コマンドを使います。

6. 表領域をオンラインにする。または、データベースを停止してから、再起動する。

USERS 表領域がオフラインで、データベースがオープンしている場合、表領域をオンラインに戻します。データベースがマウントされたが、クローズしている場合は、データベースをオープンしてください。

7. データベースをバックアップする。データベースの構造を変更した後は、ただちにデータベースを完全バックアップしてください。

関連項目：DBA_DATA_FILES データ・ディクショナリ・ビューの詳細は、『Oracle8 Server リファレンス・マニュアル』を参照してください。

表領域をオンラインにする方法の詳細は、8-8 ページの「表領域をオンラインにする」を参照してください。

データ・ファイル内のデータ・ブロックの検証

チェックサムを使ってデータ・ブロックを検証するように Oracle を構成する場合は、初期化パラメータ DB_BLOCK_CHECKSUM を TRUE に設定します。このパラメータ値は、動的に変更でき、または init.ora パラメータ・ファイルに設定できます。DB_BLOCK_CHECKSUM のデフォルト値は FALSE です。

ブロックのチェックを使用可能にすると、Oracle がディスクに書き込まれる各ブロックのチェックサムを計算します。チェックサムは一時ブロックを含むすべてのデータ・ブロックについて計算されます。

DBW0 プロセスは各ブロックのチェックサムを計算して、ブロック・ヘッダーにこのチェックサムを格納します。チェックサムはダイレクト・ロードによっても計算されます。

Oracle は次にデータ・ブロックを読み込むときに、ブロックの破損を検出するためにチェックサムを使います。破損が検出されると、メッセージ ORA-01578 が戻され、破損に関する情報がトレース・ファイルに書き込まれます。

警告： DB_BLOCK_CHECKSUM を TRUE に設定すると、パフォーマンスのオーバーヘッドが発生することがあります。このパラメータを TRUE に設定するのは、データ破損の問題を診断するために Oracle の技術サポート担当員の指示があった場合だけにしてください。

データ・ファイルの情報の表示

次のデータ・ディクショナリ・ビューは、データベースのデータ・ファイルに関して有益な情報を提供します。

- USER_EXTENTS、DBA_EXTENTS
- USER_SEGMENTS、DBA_SEGMENTS
- USER_FREE_SPACE、DBA_FREE_SPACE
- DBA_USERS

- DBA_TS_QUOTAS
- USER_TABLESPACES、DBA_TABLESPACES
- DBA_DATA_FILES
- V\$DATAFILE

次の例では、このマニュアルの他の章でまだ説明されていないビューの使用方法を説明します。データベースには、2つの表領域 SYSTEM と USERS が含まれているとします。USERS は2つのファイル、FILE1(100MB) と FILE2(200MB) から構成されます。表領域は通常のオフライン状態になっています。ここで、データベースのデータ・ファイルに関する状態情報を表示するには、V\$DATAFILE を問い合わせます。

```
SELECT name,
       file#,
       status,
       checkpoint_change# "CHECKPOINT" FROM   $datafile;
```

NAME	FILE#	STATUS	CHECKPOINT
-----	-----	-----	-----
filename1	1	SYSTEM	3839
filename2	2	OFFLINE	3782
filename3	3	OFFLINE	3782

FILE# は各データ・ファイルのファイル番号を示します。データベースとともに作られる、SYSTEM 表領域内の最初のデータ・ファイルは常にファイル 1 になります。STATUS は、データ・ファイルに関する他の情報を示します。データ・ファイルが SYSTEM 表領域の一部である場合、このファイルの STATUS は SYSTEM になります（ただし、ファイルが回復を必要と場合を除きます）。SYSTEM 表領域以外の表領域内のデータ・ファイルがオンラインになっている場合、このファイルの STATUS は ONLINE になります。SYSTEM 表領域以外の表領域内のデータ・ファイルがオフラインになっている場合、このファイルの STATUS は OFFLINE または RECOVER になります。CHECKPOINT は、データ・ファイルの最も最近のチェックポイントに対して書き込まれた最後の SCN を示します。

スキーマ・オブジェクトを管理するための ガイドライン

この章では、スキーマ・オブジェクトを管理するためのガイドラインについて説明します。
トピックは次のとおりです。

- データ・ブロックの領域管理
- 記憶領域パラメータの設定
- 領域の割当ての解除
- データ型の使用領域の理解

第 11 章～第 16 章の説明に従って特定のスキーマ・オブジェクトを管理する前に、この章で説明する概念をよく理解しておく必要があります。

データ・ブロックの領域管理

ここでは、データ・ブロックの領域を管理する方法について説明します。トピックは、次のとおりです。

- PCTFREE パラメータ
- PCTUSED パラメータ
- 関連する PCTUSED と PCTFREE の値を選択する

PCTFREE パラメータと PCTUSED パラメータを使って、次のような変更ができます。

- データの書込みと検索のパフォーマンスを向上させる。
- データ・ブロック内の未使用領域を少なくする。
- データ・ブロック間の行連鎖を少なくする。

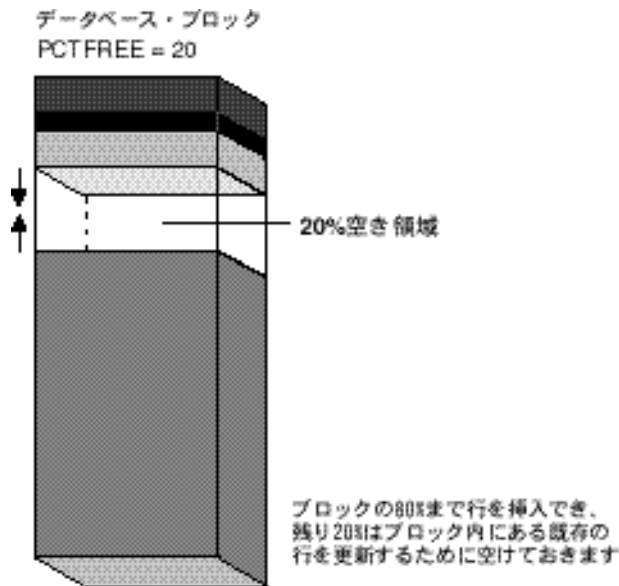
PCTFREE パラメータ

PCTFREE パラメータは、すでにブロックに含まれている行の更新用に確保しておくブロックの割合を設定します。たとえば、CREATE TABLE 文で次のようなパラメータを指定するとします。

```
PCTFREE 20
```

これは、この表のデータ・セグメントに使われる各データ・ブロックの 20 パーセントが空きのまま保持され、各ブロックにすでに存在する行を更新する際に使えることを示します。図 10-1 は、PCTFREE を示したものです。

図 10-1 PCTFREE



ブロックが PCTFREE に達するまで、新しい行の挿入とデータ・ブロック・ヘッダーの増加によって、データ・ブロックの空き領域が埋められていきます。

PCTFREE を指定する

PCTFREE のデフォルト値は 10 パーセントです。PCTFREE と PCTUSED の合計が 100 を超えない範囲であれば、0 ～ 99 の任意の整数（0 と 99 を含む）を指定できます。

PCTFREE の値を小さくすると、次のような影響が現れます。

- 既存の表の行を更新するために確保される領域が小さくなる。
- 挿入によってより完全にブロックが満杯になる。
- 表または索引の全データが格納されるブロックが少なくなる（ブロックあたりの行またはエントリは多くなる）ので、領域が節約されることもある。

ほとんど変更されないセグメントなどでは、PCTFREE の値は小さい方が適しています。

PCTFREE の値を大きくすると、次のような影響が現れます。

- 既存の表の行を更新するために確保される領域が大きくなる。
- 同じ量の挿入データに必要なブロックが多くなる（ブロックあたりに挿入する行は少なくなる）こともある。

- Oracle は行断片を頻繁に連鎖する必要がないので、更新パフォーマンスが改善されることもある。

頻繁に更新されるセグメントなどでは、PCTFREE の値が大きい方が適しています。

PCTFREE を設定する前に必ず、表や索引データの性質を把握しておいてください。更新により行が大きくなることがあります。新しい値は、それらが置き換わる値と同じサイズかもしれないし、違うサイズかもしれません。データ値が増加していく更新を何度もする場合は、PCTFREE を大きくする必要があります。行に対する更新が全体の行幅に影響を与えない場合は、PCTFREE は小さくてもかまいません。データベース管理者の目標は、高密度でまとめたデータと優れた更新パフォーマンスとの間で満足のいく妥協点を見いだすことです。

クラスタ化されていない表の PCTFREE クラスタ化されていない表の行のデータ・サイズが大きくなりそうな場合、更新のための領域をいくらか確保してください。そうしないと、更新された行がブロック間で連鎖してしまう可能性が高くなります。

クラスタ化表の PCTFREE クラスタ化されていない表についての説明は、クラスタ化された表にもあてはまります。ただし、PCTFREE に達すると、同じクラスタ・キーに含まれている任意の表の新しい行は、既存のクラスタ・キーに連鎖される新しいデータ・ブロックに格納されます。

索引の PCTFREE PCTFREE を指定できるのは、初めて索引を作るときだけです。

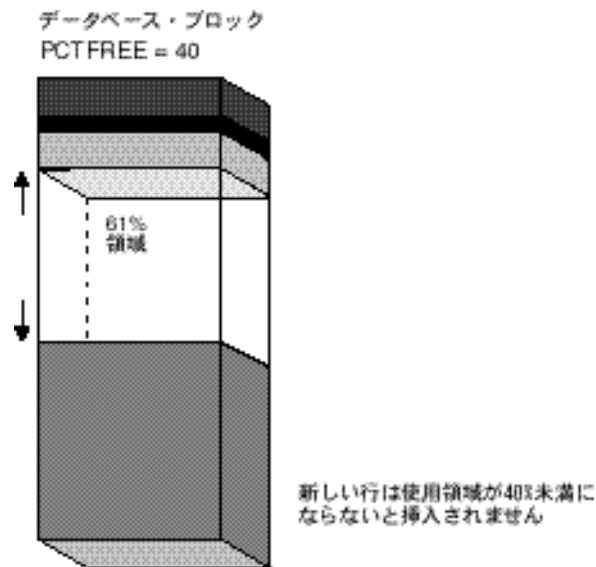
PCTUSED パラメータ

一度データ・ブロックが PCTFREE まで達すると、使われているブロックの割合が PCTUSED パラメータより小さくなるまで、そのブロックに新しい行は挿入されません。この値に達するまでは、データ・ブロックの空き領域はそのデータ・ブロックにすでに含まれている行の更新用にしか使われません。たとえば、CREATE TABLE 文で次のようなパラメータを指定するとします。

```
PCTUSED 40
```

この場合、この表のデータ・セグメントに使われるデータ・ブロックには、ブロック内で使われている領域が 39 パーセント以下になるまでは、新しい行は挿入されません（ブロックが使用する領域がすでに PCTFREE に達していると仮定する）。図 10-2 は、このことを示したものです。

図 10-2 PCTUSED



PCTUSED を指定する

PCTUSED のデフォルト値は 40 パーセントです。一度データ・ブロックの空き領域が PCTFREE に達すると、使われている領域の割合が PCTUSED を下回るまで、そのブロックに新しい行は挿入されません。そのパーセント値は、合計領域からオーバーヘッドが差し引かれた後、データに使えるブロック領域に対するものです。

PCTUSED と PCTFREE の合計が 100 を超えない範囲であれば、PCTUSED には 0 ~ 99 の間の任意の整数 (0 と 99 を含む) を指定できます。

PCTUSED の値を小さくすると、次のような影響が現れます。

- その使用の割合を下回ったときに、ブロックを空きリストに移動するために UPDATE 文と DELETE 文において発生する処理コストが少なくなる。
- データベース内の未使用領域が増大する。

PCTUSED の値を大きくすると、次のような影響が現れます。

- 領域効率が改善される。
- INSERT と UPDATE における処理コストが増大する。

関連する PCTUSED と PCTFREE の値を選択する

PCTFREE または PCTUSED のデフォルト値を使わない場合は、次のガイドラインを参考にしてください。

- PCTFREE と PCTUSED の和は 100 以下でなければならない。
- 和が 100 になる場合、Oracle は PCTFREE の空き領域を維持しようとし、処理コストは最大になる。
- ブロック・オーバーヘッドは、PCTUSED または PCTFREE の計算に含まれない。
- PCTUSED が 75 で PCTFREE が 20 の場合のように、PCTFREE と PCTUSED の和と 100 との差が小さいほど、パフォーマンス・コストで領域使用の効率がよくなる。

PCTFREE と PCTUSED の値を選択する例

次の例では、PCTFREE と PCTUSED の値を表に指定する方法とその理由を示します。

例 1	シナリオ：	一般的なアクティビティに、行のサイズを大きくする UPDATE 文が含まれている。
	設定値：	PCTFREE = 20 PCTUSED = 40
例 2	シナリオ：	多くのアクティビティに、処理の対象となる行のサイズを大きくしない INSERT 文や DELETE 文、および UPDATE 文が含まれている。
	設定値：	PCTFREE = 5 PCTUSED = 60
	説明：	ほとんどの UPDATE 文は、行サイズを大きくしないので、PCTFREE を 5 に設定します。DELETE 文によって解放される領域がすぐに使われるように PCTUSED を 60 に設定し、処理も最小にします。
例 3	シナリオ：	表が非常に大きいため、記憶領域が最大の問題。多くのアクティビティに、読み専用トランザクションが含まれている。
	設定値：	PCTFREE = 5 PCTUSED = 40
	説明：	この表は大きく、各ブロックを完全に満杯にするのが望ましいので、PCTFREE を 5 に設定します。

記憶領域パラメータの設定

ここでは、各種のデータ構造に対して設定できる記憶領域パラメータについて説明します。トピックは次のとおりです。

- 指定可能な記憶領域パラメータ
- INITTRANS と MAXTRANS を設定する
- 表領域内のセグメントのデフォルト記憶領域パラメータを設定する
- データ・セグメントの記憶領域パラメータを設定する
- 索引セグメントの記憶領域パラメータを設定する
- LOB セグメントの記憶領域パラメータを設定する
- 記憶領域パラメータの値を変更する
- 記憶領域パラメータの優先順位を理解する

記憶領域パラメータは、次のタイプの論理記憶構造に対して設定できます。

- 表領域（表領域内の任意のセグメントの大半のデフォルト）
- 表、クラスタ、スナップショット、スナップショット・ログ（データ・セグメント）
- 索引（索引セグメント）
- ロールバック・セグメント

指定可能な記憶領域パラメータ

すべてのデータベースには、記憶領域パラメータのデフォルト値があります。システムのデフォルトのかわりに、その表領域で作られるオブジェクトだけに適用されるものとなる表領域のデフォルトを指定できます。さらに、個々のオブジェクトに対して記憶設定値を指定できます。設定できる記憶領域パラメータを次に示します。

INITIAL

セグメントが作られるときに、割り当てられる第 1 エクステントのサイズ。

デフォルト: 5 データ・ブロック

最小値: 2 データ・ブロック（切上げ）

最大値: オペレーティング・システム固有

NEXT

セグメントに次に割り当てられる増分エクステントのサイズ。単位はバイトで指定します。第 2 エクステントは、NEXT のオリジナルの設定値になります。それ以降、NEXT は 1 つ前の NEXT のサイズに $(1 + \text{PCTINCREASE}/100)$ を掛けたサイズに設定されます。

デフォルト: 5 データ・ブロック
最小値: 1 データ・ブロック
最大値: オペレーティング・システム固有

MAXEXTENTS

最初のエクステントも含めて、セグメントに割り当てられる全体のエクステント数。

デフォルト: データ・ブロック・サイズとオペレーティング・システムによって異なる。
最小値: 1 (エクステント)
最大値: 無制限

MINEXTENTS

セグメントが作られるときに、割り当てられる全体のエクステント数。これによって、連続した領域が使えなくても、作成時に大きな領域を割り当てることができます。

デフォルト: 1 (エクステント)
最小値: 1 (エクステント)
最大値: 無制限

PCTINCREASE

セグメントに割り当てられた最後の増分エクステントに掛けることによって、各増分エクステントが大きくなる割合 (百分率)。PCTINCREASE が 0 である場合、各増分エクステントのサイズは一定となります。ただし、PCTINCREASE が 0 (ゼロ) より大きいと、NEXT は計算されるたびに PCTINCREASE だけ大きくなります。PCTINCREASE が負の値になることはありません。

新しい NEXT は、 $1 + \text{PCTINCREASE} / 100$ に、最後の増分エクステントのサイズ (古い NEXT) を掛けて、ブロック・サイズの次の倍数に切り上げられます。

デフォルト: 50 (%)
最小値: 0 (%)
最大値: オペレーティング・システム固有

INITRANS

データ・ブロック内の複数の行を同時にアクセスするために、最初のいくつかの DML トランザクション・エントリに対して事前に割り当てる領域を確保します。この領域は、対応するデータ・セグメントまたは索引セグメント内のすべてのデータ・ブロックのヘッダー内に確保されます。

デフォルト値は表については 1、クラスタと索引については 2 です。

MAXTRANS

複数のトランザクションが同時に同じデータ・ブロックの行にアクセスするために、ブロック内の各 DML トランザクションのエントリに対して領域が割り当てられます。INITRANS

によって確保された領域が使い果たされると、利用できる場合には、追加のトランザクション・エントリのための領域が、ブロック内の空き領域から割り当てられます。いったん割り当てられると、この領域は効果よくブロック・ヘッダーの永久部分となります。

MAXTRANS パラメータは、データ・ブロック内のデータを同時に使えるトランザクション・エントリ数を制限します。したがって、データ・ブロック内のトランザクション・エントリに割り当てることのできる空き領域を、MAXTRANS を使って制限できます。

デフォルト値はオペレーティング・システム固有のブロック・サイズの間数であり、255 を超えません。

関連項目：記憶領域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。

デフォルト値にはオペレーティング・システムによって違うものであるため、使っているオペレーティング・システム固有の Oracle のマニュアルを参照してください。

INITRANS と MAXTRANS を設定する

表またはクラスタ、索引に割り当てられたデータ・ブロックに対するトランザクション・エントリの設定は、次の基準に基づいて、オブジェクトごとに個別に設定してください。

- データベース・データに確保する領域と比較した、トランザクション・エントリに確保する領域。
- ある特定の時間に同じデータ・ブロックにアクセスする同時実行トランザクションの数。

たとえば、かなり大きな表でも、わずかなユーザーしか同時にアクセスしないのであれば、複数の同時実行のトランザクションが同じデータ・ブロックへのアクセスを必要とする確率は低いでしょう。したがって、データベースにおいて領域が特に貴重な場合には、INITRANS を小さく設定できます。

一方、日常的に、多くのユーザーが表に同時にアクセスする場合もあります。この場合、(オブジェクトが使われているときに必要となる、トランザクション・エントリ領域を割り当てるオーバーヘッドをなくすために) INITRANS を大きく設定し、ユーザーが必要なデータ・ブロックにアクセスするのを待たないように MAXTRANS を大きく設定することによって、事前に割り当てるトランザクション・エントリ領域を検討するとよいでしょう。

表領域内のセグメントのデフォルト記憶領域パラメータを設定する

データベースの表領域ごとにデフォルトの記憶領域パラメータを設定できます。表領域内のセグメントを作るとき、または作ってから後に変更するときに明示的に記憶領域パラメータを設定しないと、そのセグメントが存在する表領域の対応するデフォルト記憶領域パラメータに自動的に設定されます。

パーティション表を使用して、表レベルでデフォルト記憶領域パラメータを設定できます。表の新しいパーティションを作成すると、デフォルト記憶領域パラメータはパーティション表から継承されます。パーティション表がない場合、デフォルト記憶領域パラメータは表領域から継承されます。

表領域レベルで MINEXTENTS を指定すると、表領域に割り当てられたエクステントは最小のエクステント数の倍数に四捨五入されます。基本的には、エクステント数はブロック数の倍数です。

データ・セグメントの記憶領域パラメータを設定する

クラスタ化されてない表またはスナップショット、スナップショット・ログのデータ・セグメントに対する記憶領域パラメータは、表またはスナップショット、スナップショット・ログの CREATE 文や ALTER 文の STORAGE 句を使って設定できます。

対照的に、クラスタのデータ・セグメントに対する記憶領域パラメータは、CREATE CLUSTER コマンドまたは ALTER CLUSTER コマンドの STORAGE 句を使って設定します。クラスタ内に表やスナップショットを設定する個々の CREATE コマンドや ALTER コマンドは使いません。クラスタ化された表またはスナップショットの作成または変更時に指定された記憶領域パラメータは、無視されます。クラスタに設定された記憶領域パラメータは、表の記憶領域パラメータを上書きします。

索引セグメントの記憶領域パラメータを設定する

表の索引のために作られる索引セグメントの記憶領域パラメータは、CREATE INDEX コマンドまたは ALTER INDEX コマンドの STORAGE 句を使って設定できます。主キー制約または一意キー制約を施行するために使われる索引に対して作る索引セグメントの記憶領域パラメータは、CREATE TABLE コマンド、ALTER TABLE コマンドの ENABLE 句、あるいは ALTER INDEX コマンドの STORAGE 句に設定できます。

索引に PCTFREE を設定すると、索引を作るときにだけ効果があります。索引セグメントに PCTUSED を指定できません。

LOB セグメントの記憶領域パラメータを設定する

CREATE TABLE コマンドの NOCACHE、NOLOGGING、PCTVERSION LOB の各記憶領域パラメータを使用して LOB セグメントに記憶領域パラメータを設定できます。

記憶領域パラメータの値を変更する

現行の設定が不適切であると判断した場合、表領域に対するデフォルト記憶領域パラメータと、個々のセグメントに対する特定の記憶領域パラメータを変更できます。デフォルトの記憶領域パラメータはすべて表領域に再設定できます。ただし、変更は、その表領域に作られる新しいオブジェクトまたはセグメントに割り当てられる新しいエクステントにしか反映されません。

既存の表またはクラスタ、索引、ロールバック・セグメントの INITIAL 記憶領域パラメータと MINEXTENTS 記憶領域パラメータは変更できません。セグメントの NEXT だけを変更すると、次の増分エクステントは新しい NEXT のサイズとなり、後続するエクステントは、通常どおりに PCTINCREASE だけ大きくなります。

セグメントの NEXT と PCTINCREASE の両方を変更する場合、次のエクステントは NEXT の新しい値となり、それ以降は通常どおり PCTINCREASE を使って、NEXT が計算されます。

記憶領域パラメータの優先順位を理解する

ある時点で有効な記憶領域パラメータは、SQL 文のタイプによって決まります。次に、SQL 文を優先順位の高いものから順に記述します。

1. ALTER TABLE/CLUSTER/SNAPSHOT/SNAPSHOT LOG/INDEX/ROLLBACK SEGMENT 文
2. CREATE TABLE/CLUSTER/SNAPSHOT/SNAPSHOT LOG/CREATE INDEX/ROLLBACK SEGMENT 文
3. ALTER TABLESPACE 文
4. CREATE TABLESPACE 文
5. Oracle のデフォルトの文

オブジェクト・レベルで指定された記憶領域パラメータはどれも、表領域レベルで設定された対応するオプションに置き換わります。記憶領域パラメータがオブジェクト・レベルで明示的に設定されていないとき、表領域レベルにおけるデフォルトが適用されます。記憶領域パラメータが表領域レベルで明示的に設定されていないときは、Oracle システム・デフォルトが適用されます。記憶領域パラメータが変更されると、新しいオプションは、まだ割り当てられていないエクステントにだけ適用されます。

注意： 一時セグメントの記憶領域パラメータは、対応する表領域のために設定されたデフォルト記憶領域パラメータを常に使います。

記憶領域パラメータの例

次の文が実行されていると想定します。

```
CREATE TABLE test_storage
( . . . )
STORAGE (INITIAL 100K NEXT 100K
MINEXTENTS 2 MAXEXTENTS 5
PCTINCREASE 50);
```

また、初期化パラメータ DB_BLOCK_SIZE は 2KB に設定されています。次の表は、エクステントが TEST_STORAGE 表に割り当てられる様子を示しています。また、増分エクステン

トの値も示します。この値は USER_SEGMENTS データ・ディクショナリ・ビューまたは DBA_SEGMENTS データ・ディクショナリ・ビューの NEXT 列で参照できます。

表 10-1 エクステントの割当て

エクステン ト番号	エクステント・サイズ	NEXT の値
1	50 ブロックまたは 102400 バイト	50 ブロックまたは 102400 バイト
2	50 ブロックまたは 102400 バイト	75 ブロックまたは 153600 バイト
3	75 ブロックまたは 153600 バイト	113 ブロックまたは 231424 バイ ト
4	115 ブロックまたは 235520 バイト	170 ブロックまたは 348160 バイ ト
5	170 ブロックまたは 348160 バイト	255 ブロックまたは 522240 バイ ト

NEXT または PCTINCREASE 記憶領域パラメータが ALTER 文（たとえば、ALTER TABLE）で変更されると、その指定された値が、データ・ディクショナリに格納されている現行値に置き換わります。たとえば、3 番目のエクステントが表に割り当てられる前に、次の文によって TEST_STORAGE 表の NEXT 記憶領域パラメータを修正します。

```
ALTER TABLE test_storage STORAGE (NEXT 500K);
```

その結果、割り当てるときに 3 番目のエクステントは 500KB、4 番目のエクステントは (500KB*1.5) = 750KB となり、それ以降は同じように計算されます。

領域の割当ての解除

ここでは、未使用の領域の割当て解除について説明します。トピックは次のとおりです。

- 高水位標を表示する
- 領域割当て解除文を発行する

セグメントに領域を割り当てた後で、その領域が使われていないとわかることがよくあります。たとえば、PCTINCREASE を高い値に設定し、それによって、一部しか使われない大きなエクステントが作られることがあるかもしれません。また、ALTER TABLE ALLOCATE EXTENT 文を発行して、領域を明示的に過度に割り当ててしまうこともあるかもしれません。未使用の領域または過度に割り当てられた領域があることがわかった場合は、その領域を解放して、その未使用領域を他のセグメントから使えるようにできます。

高水位標を表示する

割当てを解除する前に、高水位標の位置とセグメント内の未使用領域の量についての情報を戻すプロシージャ (UNUSED_SPACE) が入っている DBMS_SPACE パッケージを使えます。

セグメント内で、高水位標は使われた領域の量を示します。(割当てを解除したい領域にデータがまったくない場合でも) 高水位標より下の領域は解放できません。ただし、セグメントが完全に空の場合は、TRUNCATE DROP STORAGE 文を使って領域を解放できます。

領域割当て解除文を発行する

次の文は、セグメント (表または索引、クラスタ) 内の未使用領域の割当てを解除します。KEEP 句は、オプションです。

```
ALTER TABLE table DEALLOCATE UNUSED KEEP integer;
ALTER INDEX index DEALLOCATE UNUSED KEEP integer;
ALTER CLUSTER cluster DEALLOCATE UNUSED KEEP integer;
```

未使用領域の量を KEEP に明示的に指定すると、残りの未使用領域の割当てが解除されても、この領域は保持されます。残りのエクステントの数が、MINEXTENTS よりも少なくなると、MINEXTENTS の値は新しい数を反映して変更されます。初期エクステントが小さくなった場合は、初期エクステントの新しいサイズを反映して INITIAL の値が変更されます。

KEEP 句を指定しないと、初期エクステントのサイズと MINEXTENTS が保たれる限り、すべての未使用領域 (高水位標より上のすべて) の割当てが解除されます。したがって、高水位標が MINEXTENTS の境界の内部にある場合でも、MINEXTENTS は保持され、初期エクステントのサイズは減少しません。

関連項目: 未使用領域の割当て解除に係する構文とオプションの詳細は、『Oracle8 Server SQL リファレンス』を参照してください。

DBA_FREE_SPACE ビューを見ることによって、割当て解除された領域を確認できます。このビューの詳細は、『Oracle8 Server リファレンス・マニュアル』を参照してください。

領域の割当てを解除する例

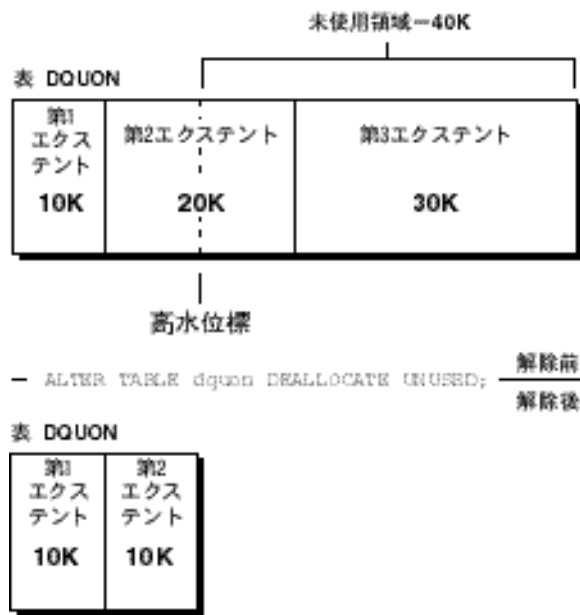
ここでは、領域の割当てを解除する様々なシナリオを示します。あらかじめ、『Oracle8 Server リファレンス・マニュアル』の ALTER...DEALLOCATE UNUSED 文について理解しておく必要があります。

例 1

表 DQUON は、3 つのエクステントから構成されています (図 10-3 を参照してください)。第 1 エクステントは 10KB、第 2 エクステントは 20KB、第 3 エクステントは 30KB です。高水位標は第 2 エクステントの中央にあり、40KB の未使用領域があります。次の文は、未使用領域の割当てをすべて解除し、表 DQUON には 2 つのエクステントが残ります。第 3 エクステントはなくなり、第 2 エクステントのサイズは 10KB になります。

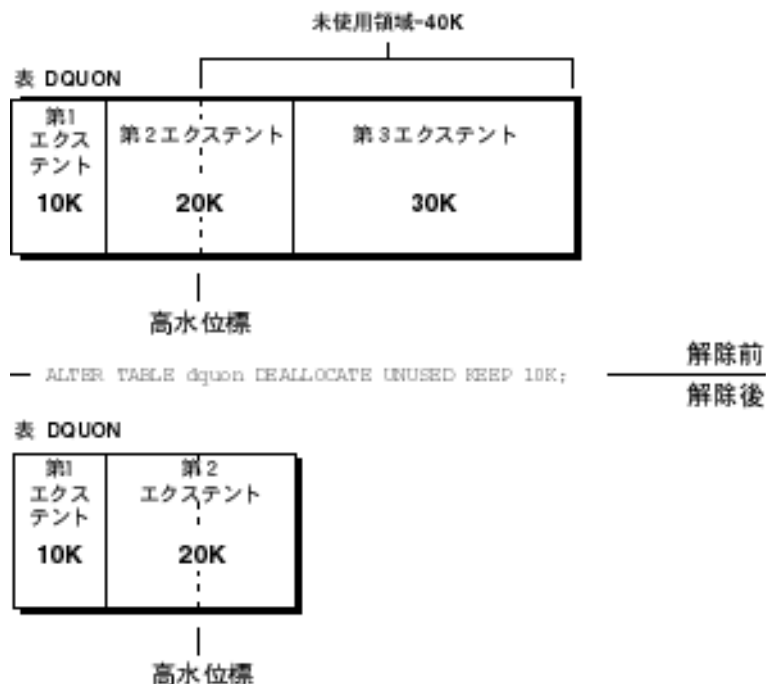
```
ALTER TABLE dquon DEALLOCATE UNUSED;
```

図 10-3 すべての未使用領域の割当てを解除する



DQUON のすべての未使用領域の割当てを解除して KEEP 10K を指定すると（図 10-4 を参照）、第 3 エクステントの割当てが解除され、第 2 エクステントはそのまま残ります。

図 10-4 未使用領域の割当てを解除し、KEEP 10K を指定する



DQUON のすべての未使用領域の割当てを解除して KEEP 20K を指定すると、第 3 エクステントは 10KB に縮小され、第 2 エクステントのサイズは変わりません。

```
ALTER TABLE dquon DEALLOCATE UNUSED KEEP 20K;
```

例 2

ALTER TABLE DQUON DEALLOCATE UNUSED 文を発行すると、第 3 エクステントの割当ては完全に解除され、第 2 エクステントには 10KB 残ります。次に割り当てられるエクステントのデフォルトのサイズは、最後に完全に割当て解除されたエクステントのサイズ、つまり、この例では 30K に設定されます。ただし、次のエクステントのサイズを明示的に設定できる場合は、ALTER ... STORAGE [NEXT] 文を使います。

例 3

MINEXTENTS に設定した数のエクステントを保持するのに、DEALLOCATE では、そのセグメントにもともと割り当てられていたエクステントの割当てを解除する一方で、もともとインスタンスに割り当てられたエクステント（高水位標より下に追加されたもの）を保持できます。

たとえば、表 DQUON の MINEXTENTS の値が 2 だと仮定します。例 1 と例 2 では結果は同じです。ただし、MINEXTENTS の値が 3 の場合は、ALTER TABLE DQUON DEALLOCATE UNUSED 文は有効ではなく、ALTER TABLE DQUON DEALLOCATE UNUSED KEEP 10KB 文では第 3 エクステントが削除され、MINEXTENTS の値は 2 に変更されます。

データ型の使用領域の理解

表またはその他のデータ構造を作る際には、必要となる領域の大きさを把握しておく必要があります。次に説明するように、データ型ごとに異なる領域要件があります。

文字 データ型

CHAR データ型と VARCHAR2 データ型では、Oracle を稼働させるハードウェアが使うキャラクタ・セットに応じて、英数字データを ASCII (アメリカ合衆国標準情報交換用符号) または EBCDIC (拡張 2 進化 10 進コード) の文字列で格納します。また、Oracle の各国語サポート (NLS) 機能によってサポートされるキャラクタ・セットを使って、データも格納できます。

CHAR データ型は、固定長文字列を格納します。CHAR 列を含む表を作るとき、CHAR 列の列長は 1 ~ 255 の間で指定できます (単位は文字ではなくバイト)。デフォルトは 1 バイトです。列の値がその列長より小さい場合、列の残りの領域にはブランクが埋められます。

VARCHAR2 データ型は、可変長文字列を格納します。VARCHAR2 列を含む表を作るとき、VARCHAR2 列の最大列長は 1 ~ 4000 の間で指定できます (単位は文字ではなくバイト)。各行では、列の中のそれぞれの値が可変長フィールドとして格納されます。列の残りの領域には、ブランクが追加されません。

NUMBER データ型

NUMBER データ型には、固定小数点数および浮動小数点数を格納します。つまり、 $1 \times 10^{-130} \sim 9.99...9 \times 10^{125}$ (最大有効桁数 38) の範囲の正数と、 $-1 \times 10^{-130} \sim -9.99...9 \times 10^{125}$ (最大有効桁数は 38) の範囲の負数、および 0 (ゼロ) を格納します。

NUMBER データ型の列を定義するときには、オプションとして精度 (桁数の合計) と位取り (小数点以下の桁数) も指定できます。精度を指定しないと、列は値をそのとおりに格納します。位取りと精度は、両方とも省略できます。その場合、精度は 38 となり、指定された位取りが維持されます。次のように、精度を指定しないで位取りを指定することもできます。

```
column_name NUMBER (*, scale)
```

DATE データ型	DATE データ型は、日付や時刻など、ある時点を表す値を格納します。日付データは、それぞれ 7 バイトの固定長フィールドに格納されます。
LONG データ型	LONG として定義される列は、2GB までの情報を持つ可変長文字データを格納します。LONG データは、テキスト・データであり、異なるキャラクタ・セットに移動するときには適切に変換されます。LONG データに索引を付けることはできません。
RAW と LONG RAW データ型	<p>RAW は VARCHAR2 文字データ型に似た可変長のデータ型です。ただし、RAW データと LONG RAW データの送信時には、Net8(ユーザー・セッションをインスタンスに接続する)およびインポート・ユーティリティとエクスポート・ユーティリティで文字が変換されない点が異なります。これとは対照的に、データベースのキャラクタ・セットとユーザー・セッションのキャラクタ・セットが違う場合、CHAR データおよび VARCHAR2 データ、LONG データに関し、この 2 つのキャラクタ・セットの間の変換が Net8 とエクスポート / インポートによって自動的になされます。</p> <p>RAW データに索引を付けることはできますが、LONG RAW データに索引を付けることはできません。</p>
ROWID と ROWID データ型	<p>Oracle データベースのクラスタ化されてない表のあらゆる行に、行の行断片 (複数の行断片の間で連鎖される行の場合には最初の行断片) の物理アドレスに対応する、一意の ROWID が割り当てられます。</p> <p>Oracle データベースの各表の内部には、ROWID という疑似列があります。この疑似列は、SELECT 文 (SQL*Plus を使っているなら DESCRIBE 文) を実行して、表の構造を記述したときは表示されませんが、SQL 問合せで予約語 ROWID を列名として使って検索できます。</p>

ROWID と ROWID データ型

Oracle データベースのクラスタ化されていない表のあらゆる行に、行の行断片（複数の行断片の間で連鎖される行の場合には最初の行断片）の物理アドレスに対応する、一意の ROWID が割り当てられます。

ROWID は選択された行ごとの物理アドレスを 2 進数を使って表します。ROWID の VARCHAR2 を 16 進数で表す場合は、block.slot.file の 3 つの部分に分かれます。block はファイルのうちその行を含むデータ・ブロックであり、データ・ファイルに対する相対値です。row はそのブロック内の行です。file はその行を含むデータ・ファイルです。通常、行に割り当てられた ROWID は変更されません。インポート・ユーティリティとエクスポート・ユーティリティを使って、行がインポートまたはエクスポートされたときには例外が発生します。行が表から削除されると（そしてそれを含むトランザクションがコミットされると）削除された行に対応付けられていた ROWID は、後続のトランザクションで挿入する行に割り当てることができます。

関連項目：NLS と各種キャラクタ・セットのサポートの詳細は、『Oracle8 Server リファレンス・マニュアル』を参照してください。

Oracle のデータ型のまとめ

表 10-2 に、Oracle の各データ型についての重要な情報をまとめます。

表 10-2 Oracle のデータ型についての情報のまとめ

データ型	説明	列の長さ (バイト)
CHAR (size)	長さが size バイトの固定長文字データ。	表の各行ごとに固定（後続空白あり）。最大サイズは行あたり 2000 バイト、デフォルト・サイズは行あたり 1 バイト。size を設定する前に、使うキャラクタ・セットの検討が必要。（1 バイトと 2 バイトのどちらのキャラクタ・セットを使うか）
NCHAR (size)	長さが size 文字またはバイトの固定長文字データ（各国文字セットの選択による）	最大サイズは、各文字の格納に必要なバイト数で判断される。上限の値は 2000 バイト。size のデフォルトおよび最小サイズは、1 文字または 1 バイト（キャラクタ・セットによる）

表 10-2 Oracle のデータ型についての情報のまとめ

データ型	説明	列の長さ (バイト)
VARCHAR2 (<i>size</i>) NVARCHAR	可変長文字データ。最大サイズを指定しなければならない。 最大長 <i>size</i> 文字またはバイトの可変長文字列 (各国文字セットの選択による)	行ごとに可変、行あたり最大 4000 バイト。 <i>size</i> を設定するの前に、使うキャラクタ・セットの検討が必要。 最大サイズは、各文字の格納に必要なバイト数で判断される。上限の値は 4000 バイト。NVARCHAR2 にはサイズを指定しなければならない。
NUMBER (<i>p</i> , <i>s</i>)	可変長数値データ。最大の精度 <i>p</i> と位取り <i>s</i> は 38。位取り走査範囲は -84 ~ 127。	行ごとに可変。列に必要な最大領域は、行あたり 21 バイト。
DATE	固定長の日付時間データ。紀元前 4712 年 1 月 1 日 ~ 西暦 4712 年 12 月 31 日の日付。デフォルトの書式は、DD-MON-YY。	表の各行に対して 7 バイトに固定。
LONG	可変長文字データ。	表の行ごとに可変、行あたり最大 2 ³¹ バイト (2GB)。
RAW (<i>size</i>)	可変長バイナリ・データ。最大サイズを指定しなければならない。	表の行ごとに可変、行あたり最大 2000 バイト。
LONG RAW	可変長バイナリ・データ。	表の行ごとに可変、行あたり最大 2GB。
ROWID	行のアドレスを表すバイナリ・データ。	表の行ごとに 6 バイトに固定。
MLSLABEL	ラベルの表現を格納する信頼性のある Oracle データ型。	Trusted Oracle のマニュアルを参照してください。

パーティション表と索引の管理

ここでは、パーティション表と索引の管理について説明します。トピックは次のとおりです。

- パーティション表と索引とは何か
- パーティションの作成
- パーティションのメンテナンス

パーティション表と索引とは何か

注意： パーティション表または索引を作る前に、またはパーティションに関するメンテナンス操作を実行する前に、『Oracle8 Server 概要』にあるパーティションについての情報を参照してください。

最近の企業には、数百 GB に上るデータ（数 TB のデータになることも多い）からなる 業務上重要なデータベースが多く稼働しています。これらの企業には、大規模データベース (VLDB) のサポートおよびメンテナンスが要求されており、それらの要求を満たすような方法が必要になります。

VLDB 要求を満たす 1 つの方法は、パーティション表と索引を作り、それを使うことです。パーティション表または索引は、同じ論理属性を持つ多数の部分（パーティション）に分けられています。たとえば、1 つの表の中のすべてのパーティションで同じ列および制約定義を共有し、1 つの索引のすべてのパーティションで同じ索引オプションを共有します。各パーティションは、それぞれ別個のセグメントに格納され、それぞれ違う物理属性にできます（PCTFREE および PCTUSED、INITRANS、MAXTRANS、TABLESPACE、STORAGE など）。

各オブジェクトのパーティションをそれぞれ別個の表領域に入れる必要はありませんが、それには利点があります。別個の領域にパーティションを格納すると、次のようになります。

- 複数のパーティションでのデータ破損の可能性を低くする。
- 各パーティションを個別にバックアップおよび回復できる。
- パーティションとディスク・ドライブの対応を制御できる（I/O ロードのバランスを調整するのに重要）。

関連項目： パーティションの概念および利点の詳細は、『Oracle8 Server 概要』を参照してください。

パーティションの作成

ここでは、表と索引のパーティションを作る方法を説明します。

パーティションを作ることは、表や索引を作るのと似ています。それには PARTITION 句を含む CREATE TABLE 文を使います。また、異なる表領域に各パーティションを入れる場合は、各パーティションの表領域名も指定します。

次の例は、パーティション 4 個を含む CREATE TABLE 文であり、1 つの四半期の販売ごとに 1 つのパーティションを使います。SALE_YEAR=1994 および SALE_MONTH=7、SALE_DAY=18 の行のパーティション化キーは (1994, 7, 18) であり、第 3 パーティション（表領域 TSC）になります。SALE_YEAR=1994 および SALE_MONTH=7、SALE_DAY=1 の行のパーティション化キーは (1994, 7, 1) であり、これも第 3 パーティションです。

```
CREATE TABLE sales
Splitting PartitionsSplitting Partitions    ( invoice_no NUMBER,
```

```
sale_year INT NOT NULL,  
sale_month INT NOT NULL,  
sale_day INT NOT NULL )  
PARTITION BY RANGE ( sale_year, sale_month, sale_day)  
( PARTITION sales_q1 VALUES LESS THAN ( 1994, 04, 01 )  
  TABLESPACE tsa,  
  PARTITION sales_q2 VALUES LESS THAN ( 1994, 07, 01 )  
  TABLESPACE tsb,  
  PARTITION sales_q3 VALUES LESS THAN ( 1994, 10, 01 )  
  TABLESPACE tsc,  
  PARTITION sales_q4 VALUES LESS THAN ( 1995, 01, 01 )  
  TABLESPACE tsd);
```

関連項目：CREATE TABLE 文と PARTITION 句の詳細は、『Oracle8 Server SQL リファレンス』を参照してください。

パーティション・キー、パーティション名、バウンド、等パーティション表と索引の詳細は、『Oracle8 Server 概要』を参照してください。

パーティションのメンテナンス

ここでは、パーティションの特定のメンテナンス操作について説明します。内容は次のとおりです。

- パーティションを移動する
- パーティションを追加する
- パーティションを削除する
- パーティションを切り捨てる
- パーティションを分割する
- パーティションをマージする
- 表のパーティションの交換
- 履歴表での時間枠の移動
- 索引のパーティションの再構築
- 複数ステップのメンテナンス操作中のアプリケーション静止

関連項目：DDL 文の SQL 構文の詳細は、『Oracle8 Server SQL リファレンス』を参照してください。

パーティション表と索引を記述しているカタログ・ビューおよびパーティション表または索引のパーティションの詳細は、『Oracle8 Server リファレンス・マニュアル』を参照してください。

インポート (Import) およびエクスポート (Export)、パーティションの詳細は、『Oracle8 Server ユーティリティ』を参照してください。

パーティション化の一般的な情報は、『Oracle8 Server 概要』を参照してください。

パーティションを移動する

ALTER TABLE 文の MOVE PARTITION 句を使うと、次のことができます。

- データをクラスタ化しなおして断片化を少なくする。
- パーティションを別の表領域に移動する。
- 作成時間属性を修正する。

一般に、ALTER TABLE/INDEX MODIFY PARTITION 文を使うと、1 ステップでパーティションの物理記憶領域属性を変更できます。しかし、MODIFY PARTITION では修正できない物理属性もあります (TABLESPACE など)。そのような場合は、MOVE PARTITION 句を使います。

表のパーティションの移動

MOVE PARTITION 句を使うと、パーティションを移動できます。たとえば、DBA が I/O のバランスを調整するため、最もアクティブなパーティションを自分のディスク上の表領域に移動させたいとします。その場合、DBA は次の文を発行します。

```
ALTER TABLE parts MOVE PARTITION depot2  
TABLESPACE ts094 NOLOGGING;
```

この文は、常にパーティションの旧セグメントを削除し、新しい表領域を指定しなくても新しいセグメントが作られます。

移動するパーティションにデータが含まれているとき、MOVE PARTITION は、それぞれのローカルな索引の中のそれに一致するパーティションとグローバルな索引のすべてのパーティションとを使用不可にします。MOVE PARTITION を発行した後で、それらの索引のパーティションを再構築する必要があります。

索引のパーティションの移動

MOVE PARTITION や DROP TABLE PARTITION などの操作では、グローバルな索引のすべてのパーティションが使用不可になります。ALTER INDEX REBUILD PARTITION 文を使って個々に各パーティションを再構築することによって、索引全体を再構築できます。そのような再構築は、並行実行できます。

また、単に索引を削除して作りなおすという方法もあります。

パーティションを追加する

ここでは、新しいパーティションをパーティション表に追加する方法と、パーティションをローカル索引に追加する方法を説明します。

表のパーティションの追加

ALTER TABLE ADD PARTITION 文を使うと、新しいパーティションを「高位」側の最後（既存の最後のパーティションの次の位置）に追加できます。パーティションを表の先頭または中央に追加する場合、あるいは最高位のパーティションのパーティション・バウンドが MAXVALUE である場合は、かわりに SPLIT PARTITION 文を使う必要があります。

最高位のパーティションのパーティション・バウンドが MAXVALUE 以外のときは、ALTER TABLE ADD PARTITION 文を使ってパーティションを追加できます。

たとえば、SALES という表があり、それには今月のデータと過去 12 か月分のデータが含まれているとします。1996 年 1 月 1 日に、DBA は 1 月用のパーティションを追加します。

```
ALTER TABLE sales
  ADD PARTITION jan96 VALUES LESS THAN ( '960201' )
  TABLESPACE tsx;
```

表にローカル索引が定義されていて、ALTER TABLE...ADD PARTITION 文を発行すると、一致するパーティションは各ローカル索引にも追加されます。新しい索引のパーティションには、Oracle によって名前とデフォルトの物理記憶領域属性が割り当てられるため、ADD 操作が完了したら、それらを改名または変更するとよいでしょう。

索引のパーティションの追加

パーティションは、明示的にはローカル索引に追加できません。新しいパーティションをローカル索引に追加できるのは、パーティションをその基礎である表に追加するときだけです。

最高位のパーティションのパーティション・バウンドが MAXVALUE であるため、グローバル索引にはパーティションを追加できません。最高位のパーティションを新しく追加する場合は、ALTER INDEX SPLIT PARTITION 文を使ってください。

パーティションを削除する

ここでは、ALTER TABLE DROP PARTITION 文を使って表および索引のパーティションとそれらのデータを削除する方法を説明します。

表のパーティションの削除

ALTER TABLE DROP PARTITION 文を使って、表のパーティションを削除できます。

この表にローカルな索引が定義されている場合は、ALTER TABLE DROP PARTITION により、各ローカル索引からも一致するパーティションが削除されます。

データおよびグローバルな索引を含む表のパーティションを削除する ただし、パーティションにデータとグローバルな索引が含まれる場合、表のパーティションを削除するには次のいずれかの方法を使います。

1. グローバルな索引は、ALTER TABLE DROP PARTITION 文中の正しい位置にあるようにしておく。この状況では、DROP PARTITION により、グローバルな索引のすべてのパーティションが使用不可になるため、後でそれらを再構築しなければなりません。

注意： ALTER TABLE DROP PARTITION 文はすべてのグローバルな索引パーティションを使用不可とマーク設定するだけでなく、すべてのパーティション化されていない索引を使用不可状態にします。パーティション化された索引全体は 1 つの文で再構成できないので、次の文の sal1 はパーティション化されていない索引です。

```
ALTER TABLE sales DROP PARTITION dec94;  
ALTER INDEX sales_area_ix REBUILD sal1;
```

削除するパーティションに、その表の全データのうち大部分が含まれるような大規模な表の場合には、この方法が最適です。

2. ALTER TABLE DROP PARTITION 文を発行する前に、DELETE コマンドを発行し、パーティションからすべての行を削除する。DELETE コマンドでグローバルな索引が更新され、さらにトリガーが起動されて、再実行 (REDO) および取消し (UNDO) ログが生成されます。

注意： すべてのパーティションの行を削除する前に、パーティションの NOLOGGING 属性を設定する (ALTER TABLE...MODIFY PARTITION...NOLOGGING) ことにより、ロギングの量を大幅に削減できます。

たとえば、DBA がパーティション・バウンド 10000 である最初のパーティションを削除するとします。DBA は次の文を発行します。

```
DELETE FROM sales WHERE TRANSID < 10000;  
ALTER TABLE sales DROP PARTITION dec94;
```

これは、小さい表の場合、または削除されるパーティションにその表の全データのうちのく一部分だけが含まれるような大規模な表の場合には最適な方法です。

データおよび参照整合性制約索引を含む表のパーティションを削除する パーティションにデータおよび参照整合性制約索引が含まれる場合、表のパーティションを削除するには次のいずれかの方法を使います。

1. 整合性制約を使用禁止にし、ALTER TABLE DROP PARTITION 文を発行してから、整合性制約を使用可能にする。

```
ALTER TABLE sales  
  DISABLE CONSTRAINT dname_sales1;  
ALTER TABLE sales DROP PARTITION dec94;
```

```
ALTER TABLE sales
  ENABLE CONSTRAINT dname_sales1;
```

削除するパーティションに、その表の全データのうち大部分が含まれるような大規模な表の場合には、この方法が最適です。

2. ALTER TABLE DROP PARTITION 文を発行する前に、DELETE コマンドを発行し、パーティションからすべての行を削除する。DELETE コマンドで参照整合性制約が施行され、さらにトリガーが起動され、再実行 (REDO) および取消し (UNDO) ログが生成されます。

```
DELETE FROM sales WHERE TRANSID < 10000;
ALTER TABLE sales DROP PARTITION dec94;
```

これは、小さい表の場合、または削除されるパーティションにその表の全データのうちのごく一部分だけが含まれるような大規模な表の場合には最適な方法です。

索引のパーティションの削除

ローカルな索引のパーティションは、明示的には削除できません。ローカルな索引のパーティションを削除できるのは、パーティションをその基礎である表から削除するときだけです。

グローバルな索引のパーティションが空の場合は、明示的に ALTER INDEX DROP PARTITION 文を発行することによって削除できます。

グローバルな索引のパーティションにデータが含まれている場合に、そのパーティションを削除すると、次の最高位パーティションが使用不可となります。たとえば、DBA が索引のパーティション P1 を削除したい場合に、P2 が次の最高位パーティションであるとする、DBA は、次の文を発行しなければなりません。

```
ALTER INDEX npr DROP PARTITION P1;
ALTER INDEX npr REBUILD PARTITION P2;
```

注意: グローバルな索引では、最高位のパーティションは削除できません。

パーティションを切り捨てる

表のパーティションからすべての行を削除したいときは、ALTER TABLE TRUNCATE PARTITION 文を使います。索引のパーティションの切捨てはできません。しかし、ALTER TABLE TRUNCATE PARTITION 文により、ローカルな各索引の中の一一致するパーティションは切り捨てられます。

パーティション表の切捨て

ALTER TABLE TRUNCATE PARTITION 文を使って、空白を再生したり再生せずに表のパーティションからすべての行を削除することができます。この表にローカルな索引が定義され

ている場合は、ALTER TABLE TRUNCATE PARTITION により、各ローカル索引からも一致するパーティションが切り捨てられます。

データおよびグローバルな索引を含む表のパーティションを切り捨てる ただし、パーティションにデータとグローバルな索引が含まれる場合、表のパーティションを切り捨てるには次のいずれかの方法を使います。

1. グローバルな索引は、ALTER TABLE TRUNCATE PARTITION 文中の正しい位置にあるようにしておく。

注意： ALTER TABLE TRUNCATE PARTITION文はすべてのグローバルな索引パーティションを使用不可とマーク設定するだけでなく、すべてのパーティション化されていない索引を使用禁止状態にします。パーティション化された索引全体は1つの文で再構成できないので、次の文の `sal1` はパーティション化されていない索引です。

```
ALTER TABLE sales TRUNCATE PARTITION dec94;  
ALTER INDEX sales_area_ix REBUILD sal1;
```

切り捨てるパーティションに、その表の全データのうち大部分が含まれるような大規模な表の場合には、この方法が最適です。

2. ALTER TABLE TRUNCATE PARTITION 文を発行する前に、DELETE コマンドを発行し、パーティションからすべての行を削除する。DELETE コマンドでグローバルな索引が更新され、さらにトリガーが起動されて、再実行 (REDO) および取消し (UNDO) ログが生成されます。

これは、小さい表の場合、または切り捨てるパーティションにその表の全データのうちのく一部分だけが含まれるような大規模な表の場合には最適な方法です。

データおよび参照整合性制約を含む表のパーティションを切り捨てる パーティションにデータおよび参照整合性制約索引が含まれる場合、表のパーティションを切り捨てるには次のいずれかの方法を使います。

1. 整合性制約を使用禁止にし、ALTER TABLE TRUNCATE PARTITION 文を発行してから、整合性制約を再び使用可能にする。

```
ALTER TABLE sales  
  DISABLE CONSTRAINT dname_sales1;  
ALTER TABLE sales TRUNCATE PARTITION dec94;  
ALTER TABLE sales  
  ENABLE CONSTRAINT dname_sales1;
```

切り捨てるパーティションに、その表の全データのうち大部分が含まれるような大規模な表の場合には、この方法が最適です。

2. ALTER TABLE TRUNCATE PARTITION 文を発行する前に、DELETE コマンドを発行し、パーティションからすべての行を削除する。DELETE コマンドで参照整合性制約が施行され、さらにトリガーが起動され、再実行 (REDO) および取消し (UNDO) ログが生成されます。

注意： すべてのパーティションの行を削除する前に、パーティションの NOLOGGING 属性を設定する (ALTER TABLE...MODIFY PARTITION...NOLOGGING) ことにより、ロギングの量を大幅に削減できます。

```
DELETE FROM sales WHERE TRANSID < 10000;
ALTER TABLE sales TRUNCATE PARTITION dec94;
```

これは、小さい表の場合、または切り捨てるパーティションにその表の全データのうごく一部分だけが含まれるような大規模な表の場合には最適な方法です。

パーティションを分割する

ALTER TABLE/INDEX のこの形式は、1 つのパーティションを 2 つのパーティションに分割します。パーティションが大規模になりすぎて、バックアップ操作またはリカバリ操作、メンテナンス操作に時間がかかる原因となっている場合には、SPLIT PARTITION 句を使えます。I/O ロードを再分配するのにも、SPLIT PARTITION 句を使えます。

表のパーティションの分割

表のパーティションを分割するには、ALTER TABLE SPLIT PARTITION 文を発行します。表にローカルな索引が定義されている場合は、この文によって、各ローカル索引の中の一一致するパーティションも分割されます。新しい索引のパーティションには、Oracle によってシステム生成名とデフォルトの物理記憶領域属性が割り当てられるため、索引の分割後には、これらの索引を改名または変更するとよいでしょう。

分割するパーティションにデータが含まれている場合は、ALTER TABLE SPLIT PARTITION 文により、各ローカル索引の中の一一致するパーティション (2 個) と、グローバルな索引のすべてのパーティションが使用不可にされます。これらの索引のパーティションは、ALTER TABLE SPLIT PARTITION 文を発行した後で、再構築する必要があります。

シナリオ：表のパーティションの分割 このシナリオでは、"VET_cats" という表に "fee_katy" というパーティションがあります。この表には、JAF1 というローカル索引があります。また、この表には VET というグローバル索引があります。VET には、VET_parta と VET_partb という 2 つのパーティションが含まれています。

パーティション "fee_katy" を分割し、索引のパーティションを再構築するために DBA が発行する文は、次のとおりです。

```
ALTER TABLE vet_cats SPLIT PARTITION
    fee_katy at (100) INTO ( PARTITION
        fee_katy1 ..., PARTITION fee_katy2 ... );
ALTER INDEX JAF1 REBUILD PARTITION SYS_P00067;
ALTER INDEX JAF1 REBUILD PARTITION SYS_P00068;
ALTER INDEX VET REBUILD PARTITION VET_parta;
ALTER INDEX VET REBUILD PARTITION VET_partb;
```

注意： 新しいローカル索引のパーティションに割り当てられた名前を調べるには、データ・ディクショナリを調べる必要があります。このシナリオの場合は、SYS_P00067 と SYS_P00068 です。改名もできます。

また、すでにパーティション fee_katy1 および fee_katy2 が JAF1 に含まれていないかぎり、この分割で生成されたローカル索引パーティションに割り当てられた名前は対応する実表パーティションの名前に合致します。

索引のパーティションの分割

ローカルな索引のパーティションは、明示的には分割できません。ローカル索引のパーティションを分割できるのは、その基礎となる表のパーティションを分割するときだけです。

グローバルな索引のパーティションが空の場合、それを分割するには、ALTER INDEX SPLIT PARTITION 文を発行します。

次の文は、データを含む索引のパーティション QUON1 を分割します。

```
ALTER INDEX quon1 SPLIT
    PARTITION canada AT VALUES LESS THAN ( 100 ) INTO
    PARTITION canada1 ..., PARTITION canada2 ... );
ALTER INDEX quon1 REBUILD PARTITION canada1;
ALTER INDEX quon1 REBUILD PARTITION canada2;
```

パーティションをマージする

パーティションをマージするのに明示的な MERGE 文はありませんが、DROP PARTITION 句か EXCHANGE PARTITION 句のどちらかを使ってマージできます。

表のパーティションをマージする

次の方針のどちらかによって、表のパーティションをマージできます。

パーティション OSU1 にデータがあり、その表 OH にグローバル索引も参照整合性制約もない場合は、表のパーティション OSU1 を次の最高位パーティション OSU2 にマージできます。

パーティション OSU1 をパーティション OSU2 にマージする手順

1. データを OSU1 からエクスポートする。
2. 次の文を発行する。

```
ALTER TABLE OH DROP PARTITION OSU1;
```

3. ステップ 1 からのデータをパーティション OSU2 へインポートします。

注意： 対応するローカルな索引のパーティションもマージされます。

パーティション OSU1 をパーティション OSU2 にマージする別の方法

1. 表 OH のパーティション OSU1 を、ダミー表 COLS と交換する。
2. 次の文を発行する。

```
ALTER TABLE OH DROP PARTITION OSU1;
```

3. 「ダミー」表から SELECT としてデータを挿入して、OSU1 から OSU2 へデータを移動する。

パーティション索引のマージ

ローカルな索引のパーティションをマージする唯一の方法は、その基礎となる表のパーティションをマージすることです。

索引のパーティション BUCKS が空であれば、次の文を発行することによってグローバルな索引のパーティション BUCKS を次の最高位のパーティション GOOSU にマージできます。

```
ALTER INDEX BUCKEYES DROP PARTITION BUCKS;
```

索引のパーティション BUCKS にデータが含まれている場合は、次の文を発行します。

```
ALTER INDEX BUCKEYES DROP PARTITION BUCKS;  
ALTER INDEX BUCKEYES REBUILD PARTITION GOOSU;
```

最初の文でパーティション GOOSU は使用不可になり、2 番目の文で再度有効になります。

表のパーティションの交換

表およびパーティションのデータ（および索引）セグメントを交換することによって、パーティションをパーティション化されていない表に変換したり、表をパーティション表のパーティションに変換したりできます。表のパーティションを交換することは、パーティション化されていない表を使うアプリケーションがあって、そのような表をパーティション表のパーティションに変換したい場合に便利です。たとえば、パーティション表に移行するパーティション・ビューがすでにあるかもしれません。

シナリオ：表の隣接パーティションのマージ

ここでは、表の 2 つの隣接するパーティションをマージする方法を説明します。FEB95 というパーティションのデータを MAR95 というパーティションに移動することによって、SALES という表の 2 つのパーティション (FEB95 と MAR95) をマージする必要があるとします。

2 つの表のパーティションをマージする手順

1. FEB95 パーティションのデータを入れるための一時表を作る。

```
CREATE TABLE sales_feb95 (...)  
TABLESPACE ts_temp STORAGE (INITIAL 2);
```

2. FEB95 パーティションのセグメントを表 SALES_FEB95 と交換する。

```
ALTER TABLE sales  
EXCHANGE PARTITION feb95 WITH TABLE  
sales_feb95 WITHOUT VALIDATION;
```

これで、SALES_FEB95 表のプレースホルダ・セグメントが FEB95 パーティションに付加されました。

3. FEB95 パーティションを削除する。これにより、もともと SALES_FEB95 表の所有だったセグメントが解放されます。

```
ALTER TABLE sales DROP PARTITION feb95;
```

4. INSERT 文を使って、SALES_FEB95 表のデータを MAR95 パーティションに移動する。

```
INSERT INTO sales PARTITION (mar95)  
SELECT * FROM sales_feb95;
```

ここで拡張された表名を使用すると、さらに効率が良くなります。行が属するパーティションを計算しようとせずに、Oracle では指定されたパーティションに行が属しているかどうかを確認します。

5. SALES_FEB95 表を削除し、もともと FEB95 パーティションに対応付けられていたセグメントを解放する。

```
DROP TABLE sales_feb95;
```

6. (オプション) MAR95 パーティションを改名する。

```
ALTER TABLE sales RENAME PARTITION mar95 TO  
feb_mar95;
```

関連項目：索引のメンテナンス遅延の詳細は、『Oracle8 Server SQL リファレンス』の ALTER SESSION SET SKIP_UNUSABLE_INDEXES 文を参照してください。

シナリオ：パーティション・ビューをパーティション表に変換する

ここでは、パーティション・ビュー（「マニュアル・パーティション」）をパーティション表に変換する方法を説明します。パーティション・ビューは、次のように定義されます。

```
CREATE VIEW accounts
  SELECT * FROM accounts_jan95
  UNION ALL
  SELECT * FROM accounts_feb95
  UNION ALL
  ...
  SELECT * FROM accounts_dec95;
```

パーティション・ビューをパーティション表に段階的に移行する手順

1. パーティション表を作成することにより、最後の2つのパーティション ACCOUNTS_NOV95 と ACCOUNTS_DEC95 だけをビューから表に移行する。各パーティションは、2つのブロックの一時セグメント（ブレースホルダ）を取得する。

```
CREATE TABLE accounts_new (...)
  TABLESPACE ts_temp STORAGE (INITIAL 2)
  PARTITION BY RANGE (opening_date)
  (PARTITION jan95 VALUES LESS THAN ('950201'),
   ...
   PARTITION dec95 VALUES LESS THAN ('960101'));
```

2. EXCHANGE コマンドを使って、表をそれに対応するパーティションに移行する。

```
ALTER TABLE accounts_new
  EXCHANGE PARTITION nov95 WITH TABLE
  accounts_95 WITH VALIDATION;

ALTER TABLE accounts_new
  EXCHANGE PARTITION dec95 WITH TABLE
  accounts_dec95 WITH VALIDATION;
```

これで、NOV95 パーティションと DEC95 パーティションに対応付けられたブレースホルダ・データ・セグメントが、ACCOUNTS_NOV95 表と ACCOUNTS_DEC95 表に対応付けられたデータ・セグメントと交換されました。

3. ACCOUNTS ビューを再定義する。

```
CREATE OR REPLACE VIEW accounts
  SELECT * FROM accounts_jan95
  UNION ALL
  SELECT * FROM accounts_feb_95
  UNION ALL
  ...
  UNION ALL
  SELECT * FROM accounts_new PARTITION (nov95)
```

```
UNION ALL
SELECT * FROM accounts_new PARTITION (dec95);
```

4. ACCOUNTS_NOV95 表と ACCOUNTS_DEC95 表を削除する。これらは、もともと NOV95 パーティションと DEC95 パーティションに付加されていたプレースホルダ・セグメントの所有者です。
5. UNIONALL ビューの中のすべての表がパーティションに変換されたら、ビューと、ビューとして改名されたパーティション表とを削除する。

```
DROP VIEW accounts;
RENAME accounts_new TO accounts;
```

関連項目：ここで説明したそれぞれの文の構文および使用方法の詳細は、『Oracle8 Server SQL リファレンス』を参照してください。

索引のパーティションの再構築

ALTER TABLE DROP PARTITION などのいくつかの操作では、グローバル索引のすべてのパーティションが使用不可になります。グローバル索引のパーティションを再構築するには、次の 2 つの方法があります。

1. ALTER INDEX REBUILD PARTITION 文を発行することによって、各パーティションを再構築する（再構築は並行実行できる）。
2. 一度索引を削除し、再度作る。

注意： この方法は表を一回走査するだけなので、この方法のほうが効率的です。

履歴表での時間枠の移動

「履歴」表は、ある期間にわたる企業のビジネス取引（トランザクション）を記述するものです。履歴表は、売上げ、小切手、注文などの基礎情報を含むもので、「基礎」表になります。履歴表は、「まとめ」表、つまり、GROUP BY または AVERAGE、COUNT のような操作によって基礎情報から導出されたサマリー情報を取り込むものともなります。

履歴表の時間間隔は、ロール表示窓のようなものです。DBA は、定期的に最も古いトランザクションを記述する一連の行を削除し、最近のトランザクションを記述する一連の行に領域を割り当てます。たとえば、1995 年 4 月 30 日の業務の締めで、DBA は 1994 年 4 月のトランザクションの行（およびそれをサポートする索引項目）を削除し、1995 年 4 月のトランザクションのために領域を割り当てます。

履歴表の時間枠を移動する手順 ここで、特定の例を考えてみます。13 か月分（一年分の注文の履歴データと今月のデータ）を内容とする ORDER という表があるとします。月ごとに 1 つのパーティションがあり、各パーティションは ORDER_yymm という名前です。

ORDER 表には 2 つのローカル索引が含まれています。1 つは ORDER_IX_ONUM で、これは、注文番号に関するローカルな、接頭辞付き一意 (UNIQUE) 索引です。もう 1 つは ORDER_IX_SUPP で、これは業者番号に関するローカルな、接頭辞なし索引です。ローカル索引のパーティションの名前には、基礎となる表と一致する接尾辞が付いています。また、顧客名のための、グローバルな一意索引 ORDER_IX_CUST もあります。ORDER_IX_CUST には、アルファベット順に分けられた 3 つのパーティションが含まれています。1994 年 10 月 31 日には、次のようにして ORDER の時間枠を変更することになります。

1. 最も古い時間間隔のデータをバックアップする。

```
ALTER TABLESPACE ORDER_9310 BEGIN BACKUP;  
ALTER TABLESPACE ORDER_9310 END BACKUP;
```

2. 最も古い時間間隔のパーティションを削除する。

```
ALTER TABLE ORDER DROP PARTITION ORDER_9310;
```

3. 最近の時間間隔のパーティションを追加する。

```
ALTER TABLE ORDER ADD PARTITION ORDER_9411;
```

4. グローバル索引を削除し、作りなおす。

```
ALTER INDEX ORDER DROP PARTITION ORDER_IX_CUST;  
ALTER INDEX REBUILD PARTITION ORDER_IX_CUST;
```

複数ステップのメンテナンス操作中のアプリケーション静止

通常、Oracle は、ALTER TABLE DROP PARTITION のような個別の DDL 文を妨げる操作 (DML、DDL、ユーティリティ) が何もないようにするためのロックを取得します。しかし、パーティションのメンテナンス操作に複数ステップが必要な場合、DBA はアプリケーション (またはその他のメンテナンス操作) が、進行中の複数ステップの操作を妨げないようにしなければなりません。

たとえば、表 ORDER に参照整合性制約があり、パーティションを削除するために、それを使用禁止にしたい場合は、そのかわりに、前の章からのステップ 2 を次のものと置き換えます。

```
DELETE FROM ORDER WHERE ODATE < TO_DATE( '01-NOV-93' );  
ALTER TABLE ORDER DROP PARTITION ORDER_9310;
```

すべてのアプリケーションで使われている APPLICATION ロールからアクセス権限を取り消すことによって、DELETE ステップと複数の DROP PARTITION ステップの間で新しい行を ORDER に挿入できないようにできます。毎晩または毎週、正しく定義されたバッチ枠の中で、ユーザー・レベルのアプリケーションをすべて終了させることもできます。

12

表の管理

この章では、表を管理する方法について説明します。トピックは次のとおりです。

- 表を管理するためのガイドライン
- 表の作成
- 表の変更
- 表の記憶領域を手動で割り当て
- 表の削除
- 索引構成表

この章で説明する作業を実行する前に、第 10 章「スキーマ・オブジェクトを管理するためのガイドライン」の内容をよく理解しておいてください。

表を管理するためのガイドライン

ここでは、表の管理のガイドラインについて説明します。トピックは次のとおりです。

- 作成前の表の設計
- データ・ブロック領域の使用方法を指定する
- トランザクション・エントリ・パラメータを指定する
- 各表の位置を指定する
- 表作成を並行化する
- 回復不能表作成に関する考慮事項
- 表のサイズの見積もりと記憶領域パラメータの設定
- 大規模な表の計画
- 表の制限

表の管理をできる限り簡単にするために、ここに示すガイドラインに従ってください。

作成前の表の設計

通常、アプリケーション開発者は、表を含む、アプリケーションの要素を設計しなければなりません。データベース管理者は、アプリケーション開発者から得た、アプリケーションの動作と、予想されるデータのタイプに関する情報に基づいて、記憶領域パラメータを設定し、表のクラスタを定義しなければなりません。

表が次の要素を含むように、アプリケーション開発者とともに慎重に各表を作ってください。

- 表を正規化する。
- 各列は適当なデータ型を持っている。
- 記憶領域を節約するために、NULL を許す列が最後に定義される。
- 記憶領域を節約し、SQL 文のパフォーマンスを最適化するために、適切であればいつでも表をクラスタ化する。

データ・ブロック領域の使用方法を指定する

各表を作るときに、PCTFREE パラメータと PCTUSED パラメータを指定することによって、データ・セグメントのデータ・ブロック内の領域使用の効率、および現行データ更新時の予約域を調整できます。

関連項目：PCTFREE と PCTUSED の指定の詳細は、10-2 ページの「データ・ブロックの領域管理」を参照してください。

トランザクション・エントリ・パラメータを指定する

各表を作るときに INITRANS パラメータと MAXTRANS パラメータを指定することによって、表のデータ・セグメントのデータ・ブロック内のトランザクション・エントリに、最初に割り当てられる領域とその後割り当てられる領域を調整できます。

関連項目：INITRANS と MAXTRAN の指定の詳細は、10-7 ページの「記憶領域パラメータの設定」を参照してください。

各表の位置を指定する

適当な権限と表領域割当て制限を持っていると、現在、オンライン状態の表領域内に新しい表を作成できます。そのため、新しい表を格納する予定の表領域を識別するために、CREATE TABLE 文に TABLESPACE オプションを指定すべきです。

CREATE TABLE 文で表領域を指定しない場合、自分のデフォルト表領域内に表が作られます。

新しい表を含む表領域を指定するとき、その選択が意味することを必ず理解しておいてください。各表を作るときに表領域を適切に指定することによって、次のようなことができます。

- データベース・システムのパフォーマンスを向上させる。
- データベース管理に必要な時間を短縮する。

次のように、スキーマ・オブジェクトの記憶位置を誤ると、データベースに悪影響が及びます。

- ユーザーのオブジェクトを SYSTEM 表領域に作ると、データ・ディクショナリ・オブジェクトとユーザー・オブジェクトの両方が、同じデータ・ファイルを求めて競合し、Oracle のパフォーマンスが低下する可能性がある。
- アプリケーションに関係する表をいろいろな表領域に無計画に格納すると、データベース管理者がアプリケーションのデータ管理操作（バックアップやリカバリなど）に要する時間が増大する可能性がある。

関連項目：表領域指定の詳細は、8-3 ページの「ユーザーに表領域割当て制限を割り当てる」を参照してください。

表作成を並行化する

パラレル問合せオプションをインストールしている場合、CREATE TABLE コマンドの副問合せを開いて表作成を並行化できます。複数のプロセスが同時に動作して表を作るため、表を作るときのパフォーマンスが向上します。

関連項目：パラレル問合せオプションとパラレル表作成の詳細は、『Oracle8 Server チューニング』を参照してください。

CREATE TABLE コマンドの詳細は、『Oracle8 Server SQL リファレンス』を参照してください。

回復不能表作成に関する考慮事項

表を回復不能として作ると、表をアーカイブ済みログ・ファイルから回復できません（回復不能の表作成では、必要な REDO ログ・レコードが生成されないため）。したがって、表が失われると問題がある場合は、表を作った後にバックアップする必要があります。一時的に使うために作る表など、注意が必要ない場合もあります。

CREATE TABLE AS SELECT 文の副問合せを使って表を作るときに UNRECOVERABLE を指定すると、表を回復不能として作成できます。ただし、それ以降に挿入された行は回復できます。実際、その文の実行が完了した後は、それ以降に実行される文はすべて完全に回復できます。

表を回復不能として作ると、次の利点があります。

- REDO ログ・ファイルの領域を節約できる。
- 表を作るのに要する時間が削減できる。
- 大規模な表のパラレル作成のパフォーマンスが向上する。

一般に、表を回復不能として作ったとき、小規模な表の場合より大規模な表の方が相対的にパフォーマンスの向上が大きくなります。小規模な表を回復不能として作っても、表作成に要する時間にほとんど影響はありません。一方、大規模な表では、特に表作成を並行化したときにパフォーマンスが著しく向上します。

表のサイズの見積もりと記憶領域パラメータの設定

次のような理由で、表を作る前に表のサイズを見積もると有効です。

- 索引、ロールバック・セグメント、REDO ログ・ファイルの見積もりと合わせた表の見積もりサイズを使って、作るデータベースを保持するために必要なディスク容量を決定できる。この見積もりを利用して適切なハードウェアを購入できます。
- 個々の表の見積もりサイズを使うと、表が使うディスク領域をよりよく管理できる。表を作るとき、適切な記憶領域パラメータを設定し、その表を使うアプリケーションの I/O パフォーマンスを改善できます。

たとえば、表を作る前に、表の最大サイズを見積もることができる場合を想定します。表を作るときに記憶領域パラメータを設定すると、その表のデータ・セグメントに割り当てるエクステントを少なくできます。そのため、表のデータがディスク領域の連続した部分に格納されやすくなります。これによって、この表を呼び出すディスク I/O 操作に要する時間が短くなります。

付録 A に、表のサイズの見積もりに役立つ公式が示されています。表を作る前に表サイズを見積もるかどうかにかかわらず、クラスタ化されてない表を作るときは記憶領域パラメータを明示的に設定できます。(クラスタ化した表はクラスタの記憶領域パラメータを自動的に使います。) 表を作るとき、または表を変更するときに記憶領域パラメータを設定しないと、その表が常駐する表領域に設定されたデフォルト記憶領域パラメータが自動的に使われます。

表のデータ・セグメントのエクステントに記憶領域パラメータを明示的に設定する場合、エクステントを小さくしてその数を増やすよりも、エクステントを大きくしてその数を減らすように、表のデータを格納すべきであることに注意してください。

大規模な表の計画

表とエクステントの物理的なサイズには制限はありません。MAXEXTENTS にキーワード UNLIMITED を指定することにより、大きなオブジェクトの計画を簡単にし、無駄な領域や断片化を少なくして、領域の再使用率を向上させることができます。ただし、Oracle にはエクステントの数に制限はありませんが、表の中のエクステントの数があまり多くなると、その表を必要とする操作を実行するときのパフォーマンスに影響するので注意してください。

注意： 許容されるブロックの最大値よりも MAXEXTENTS が大きくなるように、データ・ディクショナリ表を変更できません。

データベース内にこのような表がある場合、次の推奨事項について検討してください。

表と索引の分離 索引は、他のオブジェクトとは別の表領域に配置し、さらにできれば、別のディスク上に配置してください。かなり大規模な表に対して索引を削除して作りなおす必要がある場合(制約を使用可能、使用禁止にしたり、表を作りなおすときなど)、索引を別々の表領域に分離することによって、他のオブジェクトと同じ表領域に配置した場合よりも、容易に連続領域が見つかるようになります。

十分な一時領域の割当て かなり大規模な表のデータをアクセスするアプリケーションが、大規模なソートを実行する場合、大きな一時セグメントに使用可能な領域が十分であり、ユーザーがこの領域をアクセスできることを確認してください（なお、一時セグメントは、常にその表領域のデフォルトの STORAGE を使います）。

表の制限

表を作成する前に、次の制限を理解しておいてください。

- 新しいオブジェクト型を含む表を Oracle8 以前のデータベースにインポートできません。
- オリジナルのデータがデータベースにまだ存在するときは、型とエクステンツ表を異なるスキーマに移動できません。
- エクスポートされた表を異なるスキーマで同じ名前の付いた既存の表にマージできません。

表の作成

自分のスキーマに新しい表を作るには、CREATE TABLE システム権限を持っていなければなりません。別のユーザーのスキーマに表を作るには、CREATE ANY TABLE システム権限を持っていなければなりません。さらに、表の所有者は、その表を含む表領域に対する割当て制限、または UNLIMITED TABLESPACE システム権限を持っていなければなりません。

表は SQL コマンド CREATE TABLE を使用して作成します。たとえば、ユーザー SCOTT が次の文を発行すると、クラスタ化されてない表 EMP が SCOTT のスキーマに作られ、表領域 USERS に格納されます。

```
CREATE TABLE      emp (
    empno          NUMBER(5) PRIMARY KEY,
    ename          VARCHAR2(15) NOT NULL,
    job            VARCHAR2(10),
    mgr            NUMBER(5),
    hiredate       DATE DEFAULT (sysdate),
    sal            NUMBER(7,2),
    comm          NUMBER(7,2),
    deptno         NUMBER(3) NOT NULL
                  CONSTRAINT dept_fkey REFERENCES dept)

PCTFREE 10
PCTUSED 40
TABLESPACE users
STORAGE ( INITIAL 50K
          NEXT 50K
          MAXEXTENTS 10
          PCTINCREASE 25 );
```

整合性制約がいくつかの列に定義され、いくつかの記憶設定が表に明示的に指定されていることに注目してください。

関連項目：システム権限の詳細は、第 21 章「ユーザー権限とロールの管理」を参照してください。表領域割当て制限の詳細は、第 20 章「ユーザーとリソースの管理」を参照してください。

表の変更

表を変更するためには、その表が自分のスキーマに含まれているか、その表の ALTER オブジェクト権限または ALTER ANY TABLE システム権限のどちらかを持っていなければなりません。

Oracle データベース内の表は、次の理由で変更できます。

- 表に 1 つ以上の新しい列を追加する。
- 表に 1 つ以上の整合性制約を追加する。
- 既存の列の定義（データ型および長さ、デフォルト値、NOT NULL 整合性制約）を修正する。
- データ・ブロック領域使用パラメータ（PCTFREE、PCTUSED）を修正する。
- トランザクション・エントリ設定（INITRANS、MAXTRANS）を修正する。
- 記憶領域パラメータ（NEXT、PCTINCREASE）を修正する。
- 表に対応付けられた整合性制約、またはトリガーを使用可能にする、あるいは使用禁止にする。
- 表に対応付けられた整合性制約を削除する。

既存の列の長さを拡張できます。しかし、表の中に行がない場合を除いて、列の長さの縮小はできません。さらに、データ型 CHAR の列長を拡張するために表を修正している場合、特に表内の行数が多いのであれば、この操作は時間を浪費する可能性があり、さらに相当な追加記憶領域を必要とする可能性があることに注意してください。これは、各行の CHAR 値には空白を埋めて、新しい列長に合わせなければならないためです。

表のデータ・ブロック領域使用パラメータ（PCTFREE と PCTUSED）を変更するときには、すでに割り当てられているブロックと今後割り当てられるブロックを含めて、その表が使うすべてのデータ・ブロックに新しい設定が適用されます。しかし、すでに割り当てられているブロックは、領域使用パラメータが変更されてただちに再編成されるのではなく、変更した後で必要に応じて再編成されます。

表のトランザクション・エントリ設定（INITRANS、MAXTRANS）を変更するときには、MAXTRANS の新しい設定が表のすべてのデータ・ブロック（すでに割り当てられたブロックとその後割り当てられるブロック）に適用される一方、INITRANS の新しい設定はその後表に割り当てられるブロックだけに適用されることに注意してください。

記憶領域パラメータ INITIAL と MINEXTENTS は変更できません。他の記憶領域パラメータ（たとえば NEXT、PCTINCREASE）の新しい設定はすべて、その後割り当てられるエクステンツにしか影響しません。割り当てられる次のエクステンツのサイズは、NEXT と PCTINCREASE の現行値によって決まり、前の値に基づいて決まるわけではありません。

表は SQL コマンド ALTER TABLE を使って変更します。次の文は、EMP 表を変更します。

```
ALTER TABLE emp
  PCTFREE 30
  PCTUSED 60;
```

警告： 表を変更する前に、表を変更した結果についてよく理解しておいてください。

新しい列を表に追加した当初、列は NULL となります。NOT NULL 制約付きの列を追加できるのは、表に行がまったく含まれていないときだけです。

ビューまたは PL/SQL プログラム・ユニットが実表に依存する場合、実表を変更すると、依存するオブジェクトに影響が及ぶ可能性があります。

関連項目： オブジェクトの依存性を管理する方法の詳細は、17-22 ページの「オブジェクトの依存性の管理」を参照してください。

表の記憶領域を手動で割り当て

Oracle は、必要に応じて表のデータ・セグメントに追加のエクステントを動的に割り当てます。ただし、表に追加のエクステントを明示的に割り当てることもできます。たとえば、Oracle Parallel Server を使っているとき、表のエクステントを特定のインスタンスに明示的に割り当てることができます。

新しいエクステントは、ALLOCATE EXTENT オプションを指定した SQL コマンド ALTER TABLE を使って表に割り当てることができます。

関連項目： ALLOCATE EXTENT オプションの詳細は、『Oracle8 Parallel Server 概要および管理』を参照してください。

表の削除

表を削除するためには、その表が自分のスキーマに含まれているか、または DROP ANY TABLE システム権限を持っている必要があります。

SQL コマンド DROP TABLE を使って、必要ではなくなった表を削除できます。次の文は、EMP 表を削除します。

```
DROP TABLE emp;
```

削除しようとする表に、他の表の外部キーが参照している主キー、または一意キーが含まれていて、その子表の FOREIGN KEY 制約を削除するのであれば、次のように DROP TABLE コマンドに CASCADE オプションを指定してください。

```
DROP TABLE emp CASCADE CONSTRAINTS;
```

警告： 表を削除する前に、表を削除した結果についてよく理解しておいてください。

- 表を削除すると、その表定義はデータ・ディクショナリから削除される。その結果、表のすべての行はアクセスできなくなる。
 - 表に対応付けられている索引とトリガーはすべて削除される。
 - 削除する表に依存しているビューと PL/SQL プログラム・ユニットはすべてそのまま残り、無効になる（使えない）。オブジェクトの依存性を管理する方法の詳細は、17-22 ページの「オブジェクトの依存性の管理」を参照してください。
 - 削除する表のシノニムはすべてそのまま残るが、使うとエラーが戻される。
 - クラスタ化されてない表を削除する場合、それに割り当てられているエクステントは表領域の空き領域にすべて戻され、新しいエクステントまたは新しいオブジェクトを必要とするその他のオブジェクトによって再使用される。クラスタ化した表に対応する行はすべて、そのクラスタのブロックから削除される。
-

索引構成表

ここでは、索引構成表の管理について説明します。トピックは次のとおりです。

- 索引構成表とは何か
- 索引構成表を作成する
- 索引構成表をメンテナンスする
- 索引構成表を標準的な表に変換する

索引構成表とは何か

索引構成表は、主キーに従ってグループ化されたデータ行を持つ表です。このクラスタ化は、B*tree 索引を使用して行います。B*tree 索引は、主キーと非キー列の両方を格納しているという点で、正規の表の B ツリー索引とは異なる特殊なタイプの索引ツリーです。索引構成表の属性は、索引の物理データ構造体内にすべて格納されています。言い換えると、索引構成表は SQL 文で索引を定義してアクセスするための論理構成体です。

なぜ索引構成表を使うか

索引構成表は、完全一致検索および範囲検索を呼び出す問合せに対して、表データへのより高速なキーベースのアクセスができます。新しい行の追加、行の更新、行の削除などにより表データを変更すると、別の表記憶領域がないために索引構造体の更新だけが実行されます。

また、キー列が表と索引の両方で重複しないため、記憶領域要件も少なくなります。残りの非キー列は、索引構造体に格納されます。

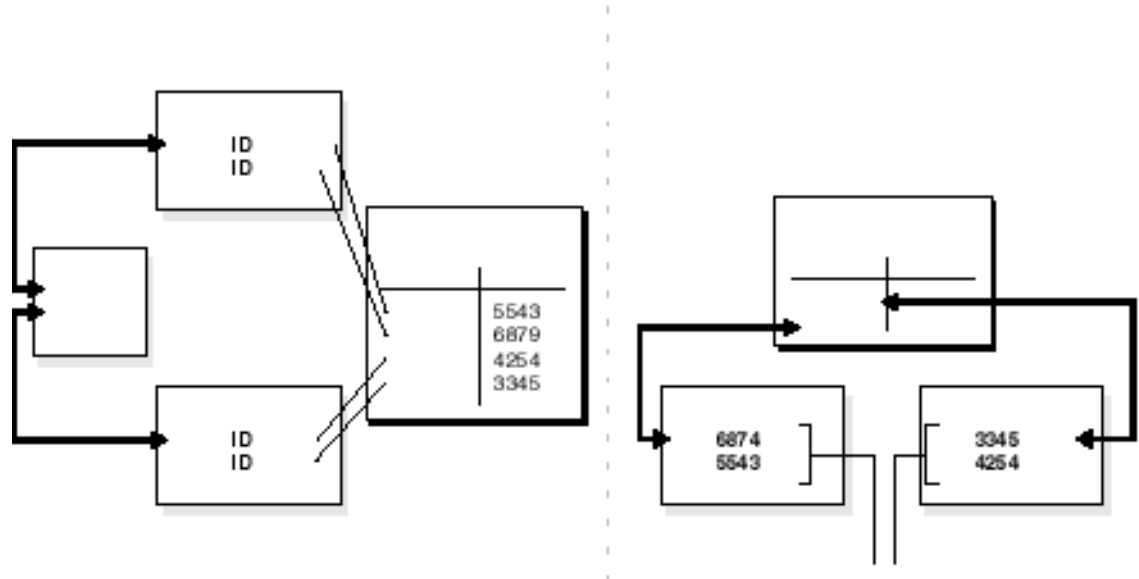
索引構成表は特に、主キーに基づいてデータを検索しなければならないアプリケーションを使用しているときに有効です。また、索引構成表はアプリケーション固有の索引構造体をモデル化するのに適しています。たとえば、テキスト、イメージ、オーディオ・データを含むコンテンツ・ベースの情報検索アプリケーションは、索引構成表を使用して有効にモデル化できる反転した索引が必要です。

関連項目：索引構成表の詳細は、『Oracle8 Server 概要』を参照してください。

索引構成表と標準的な表との違い

索引構成表は、1 つの列またはいくつかの列に索引を持つ標準的な表に似ています。ただし、索引構成表は表と B*tree の索引のための 2 つの別の記憶領域をメンテナンスせずに、表の主キーとその他の列の値を含む 1 つの B*tree の索引だけをメンテナンスします。

図 12-1 標準的な表と索引構成表の構造体



索引構成表は、主キーを使ったアクセス、または主キーに対する有効な接頭辞のキーを使ったデータ・アクセスに適しています。また、キー値と ROWID を含む別の索引構造体は作成

されないため、キーの重複はありません。表 12-1 は、索引構成表と標準的な表との違いをまとめたものです。

表 12-1 索引構成表と標準的な表との比較

標準的な表	索引構成表
ROWID により行が一意に識別される。主キーはオプションとして指定できる。	主キーにより行が一意に識別される。主キーは必ず指定。
暗黙 ROWID 列により物理的な 2 次索引を構築できる。	暗黙 ROWID 列はないため、物理的な 2 次索引は不可能。
ROWID ペースのアクセス、キー、または走査。	主キー・ベースのアクセスまたは走査。
順次走査ですべての行が戻される。	表全体走査ですべての行が主キーの順に戻される。
その他の列で UNIQUE 制約が許されている。	その他の列で UNIQUE 制約が許されていない。
その他の列で Triggers 制約が許されている。	その他の列で Triggers 制約が許されている。
別の表を含むクラスタ内にも表を格納できる。	索引構成表はクラスタ内には格納できない。
分散 SQL、レプリケーション、パーティション化のサポート。	分散 SQL とレプリケーションのサポート。パーティション化のサポートなし。

索引構成表を作成する

索引構成表を作るには、CREATE TABLE 文を使います。ただし、次の追加情報を指定する必要があります。

- ORGANIZATION INDEX 修飾子（索引構成表であることを示すもの）。
- 主キーは、単一列主キーの場合は列制約句、複数列主キーの場合は表制約句によって指定します。索引構成表では必ず主キーを指定しなければなりません。
- オプションの行オーバーフロー仕様句は、指定したしきい値を超える行列値を別のオーバーフロー・データ・セグメントに格納することにより、B*tree 索引の稠密なクラスタを保ちます。

行オーバーフロー表領域は、ブロック・サイズの割合として定義されます。行のサイズが指定したしきい値 (PCTTHRESHOLD) より大きい場合、非キー列値はオーバーフロー表領域に格納されます。言い換えると、行は列の境界で、先頭の断片と後尾の断片などの 2 つの断片に分けられます。先頭の断片は指定したしきい値に収まり、索引のリーフ・ブロック内にキーとともに格納されます。後尾の断片は、1 つ以上の行断片としてオーバーフロー領域に格納されます。したがって、索引エントリには、キー値、指定したしきい値に収まる非キー列値、行の残りの部分へのポインタが含まれています。

次の例は、索引構成表を作成するときに与える情報を示したものです。

```
SVRMGR> CREATE TABLE docindex
  ( token char(20),
    doc_oid NUMBER,
    token_frequency NUMBER,
    token_occurrence_data varchar2(512),
    CONSTRAINT pk_docindex
    PRIMARY KEY (token, doc_oid))
  ORGANIZATION INDEX TABLESPACE text_col
  PCTTHRESHOLD 20
  OVERFLOW TABLESPACE text_col_overflow;
```

この例では、ORGANIZATION INDEX 修飾子が索引構成表であることを示しており、キー列と非キー列は表の主キー (TOKEN, DOC_ID) を指す列に対して定義される索引の中にあります。

索引構成表は、オブジェクト型を格納できます。たとえば、次のように、この例の目的であるオブジェクト型の mytype の列を含む索引構成表を作成できます。

```
CREATE TABLE iot (c1 NUMBER primary key, c2 mytype)
  ORGANIZATION INDEX;
```

ただし、オブジェクト型の索引構成表は作成できません。たとえば、次の文は無効です。

```
CREATE TABLE iot of mytype ORGANIZATION INDEX;
```

関連項目：CREATE INDEX 文の詳細は、『Oracle8 Server SQL リファレンス』を参照してください。

OVERFLOW 句を使用する

前の例で指定した OVERFLOW 句では、ブロック・サイズの 20% を超える行の非キー列が表領域 TEXT_COLLECTION_OVER に入れられることが示されています。キー列は、指定したしきい値に収まらなければなりません。

非キー列を更新したために行のサイズが小さくなる場合、その更新が適用できる行断片（先頭または後尾）が識別され、その断片が書き直されます。

非キー列を更新したために行のサイズが大きくなる場合、その更新が適用できる行断片（先頭または後尾）が識別され、その断片が書き直されます。更新のターゲットが先頭の断片であることがわかった場合、この断片が再度 2 つに分かれ、指定したしきい値以下に行のサイズを保つことに注意してください。

索引リーフ・ブロックに収まる非キー列は行の先頭断片として格納され、その断片にはオーバーフロー・データ・セグメントに格納された次の行断片に先頭断片をリンクする ROWID フィールドが含まれています。オーバーフロー領域に格納されている列は、収まらない列だけです。

しきい値の選択と監視 キー列とともに最初のいくつかの非キー列が頻繁にアクセスされる場合はその非キー列を取り込めるしきい値を選択しなければなりません。

しきい値を選択した後、指定した値が適切な値であることを確認するために表を監視できます。ANALYZE TABLE LIST CHAINED ROWS 文を使用して、しきい値を超える行の数と個別性を判断できます。

索引構成表を分析するには、索引構成表ごとに別の CHAINED ROWS 表と、索引構成表の主キー記憶領域を取り込むためにすべての索引構成表を作成する必要があります。SQL スクリプト DBMSIOTC.SQL と PRVTIOTC.PLB を使用して BUILD_CHAIN_ROWS_TABLE パッケージ定義を定義し、次にこの手順を実行して索引構成表のための IOT_CHAINED_ROWS 表を作成することができます。

パッケージ定義を作成するには、'SYS' スキーマで DBMSIOTC.SQL と PRVTIOTC.PLB の両方を実行する必要があります。PUBLIC ユーザーはパッケージで定義されている手順の EXECUTE 権限を持っていたため、スキーマ内のユーザーはその手順を使用して LIST_CHAIN_ROW 表を作成できます。

関連項目： ANALYZE コマンドと SQL スクリプトの詳細は、『Oracle8 Server SQL リファレンス』を参照してください。

INCLUDING 句を使用する PCTTHRESHOLD を指定する他、INCLUDING <COLUMN_NAME> 句を使用してキー列でどの非キー列を格納するかを制御できます。Oracle では、索引リーフ・ブロック内に INCLUDING 句で指定した列まですべての非キー列を取り込むことができます。ただし、その列が指定したしきい値を超えないとします。INCLUDING 句で指定した列を超えるすべての非キー列は、オーバーフロー領域に格納されます。

たとえば、索引構成表を作成した直前の例を変更して、その表の TOKEN_OCCURRENCE_DATA 列値が常にオーバーフロー領域に格納されるようにすることができます。

```
SVRMGR> CREATE TABLE docindex
  ( token char(20),
    doc_oid NUMBER,
    token_frequency NUMBER,
    token_occurrence_data varchar2(512),
    CONSTRAINT pk_docindex
    PRIMARY KEY (token, doc_oid))
  ORGANIZATION INDEX TABLESPACE text_col
  PCTTHRESHOLD 20
  INCLUDING token_frequency
  OVERFLOW TABLESPACE text_col_overflow;
```

ここには、索引リーフ・ブロック内のキー列値で TOKEN_FREQUENCY までの非キー列だけ（この場合は 1 つの列だけ）を格納します。

索引構成表をメンテナンスする

INSERT 文および SELECT 文、DELETE 文、UPDATE 文では、標準的な表のかわりに索引構成表を使えます。ここで、索引構成表の行は B*tree に格納されており、行 ID(ROWID) が無いことに注意してください。そのため、索引構成表では、ROWID による検索はできません。

注意： 索引のパーティション化はできません。

索引構成表を変更する

索引構成表は、物理的な構成だけが異なります。論理的には、索引構成表は標準的な表と同じように処理されます。したがって、標準的な表のように索引構成表を処理します。ただし、ALTER TABLE 文を使用する場合には違いが 1 つあります。その他の定義される句の他に、次の句を使用できます。

- PCTTHRESHOLD

ブロック・サイズの割合としてしきい値を指定する整数値。

- ADD OVERFLOW

オーバーフロー・データ・セグメント（しきい値を超えているデータ行が設定されている領域）の物理属性を指定します。

索引構成表のしきい値を変更できるのは、その表が空の場合、または指定されたしきい値が現行のしきい値より大きい場合だけです。

関連項目： ALTER TABLE 文の詳細は、『Oracle8 Server SQL リファレンス』を参照してください。

シナリオ：索引構成表で ORDER BY 句を使用する場合

ORDER BY 句が主キー列またはその第 1 列だけを参照する場合、行は主キー列でソートされた状態で返されるので、オプティマイザはソートのオーバーヘッドを回避します。

たとえば、次の表を作成します。

```
CREATE TABLE EMPLOYEES (DEPT_ID INTEGER, E_ID INTEGER, E_NAME
    VARCHAR2, PRIMARY KEY (DEPT_ID, E_ID)) ORGANIZATION INDEX;
```

データはすでに主キーについてソートされているので、次の 2 つの問合せはソートのオーバーヘッドを回避します。

```
SELECT * FROM EMPLOYEES ORDER BY (DEPT_ID, E_ID);
SELECT * FROM EMPLOYEES ORDER BY (DEPT_ID);
```

ただし、主キー列の末尾列に ORDER BY 句がある場合、追加のソートがおこなわれます。

```
SELECT * FROM EMPLOYEES ORDER BY (E_ID);
SELECT * FROM EMPLOYEES ORDER BY (E_NAME);
```

シナリオ：キー列を更新する

キー列の更新は、論理的には、古いキー値を持つ行を削除し、主キーの順序を保つために適切な位置に新しいキー値を持つ行を挿入することと同じです。

次の例では、DEPT_ID 20 と E_ID 10 の社員行が削除され、DEPT_ID 23 と E_ID 10 の社員行が挿入されます。

```
UPDATE EMPLOYEES
SET DEPT_ID=23
WHERE DEPT_ID=20 and E_ID=10;
```

索引構成表を標準的な表に変換する

索引構成表から標準的な表に変換するには、Oracle IMPORT/EXPORT ユーティリティまたは CREATE TABLE AS SELECT 文を使います。

索引構成表を標準的な表に変換する手順

- 索引構成表のデータを従来型パスを使ってエクスポートする。
- 同じ定義で、標準的な表の定義を作る。
- 索引構成表のデータを、IGNORE=y（オブジェクト存在エラーを無視する）を確認してインポートする。

注意： 索引構成表を標準的な表に変換する前に、Oracle8 以前の Export ユーティリティで索引構成表をエクスポートできないことに注意してください。

関連項目： IMPORT/EXPORT の使用方法の詳細は、『Oracle8 Server ユーティリティ』を参照してください。

ビュー、順序、シノニムの管理

この章では、ビューの管理について説明します。トピックは次のとおりです。

- ビューの管理
- 順序の管理
- シノニムの管理

この章で説明する作業を実行する前に、第 10 章「スキーマ・オブジェクトを管理するためのガイドライン」の内容をよく理解しておいてください。

ビューの管理

ビューとは、1つ以上の表（または他のビュー）に含まれているデータの専用表示であり、問合せの出力をとり、それを表として扱います。ビューは、「格納された問合せ」と考えることも、「仮想表」と考えることもできます。表が使えるところでは、ほとんどの場合、ビューも使えます。

ここでは、ビューを管理する方法を説明します。トピックは次のとおりです。

- ビューを作成する
- 結合ビューを変更する
- ビューを置き換える
- ビューを削除する

ビューを作成する

ビューを作るには、次に示す要件を満たさなければなりません。

- 自分のスキーマにビューを作るには、CREATE VIEW システム権限を持っていなければなりません。別のユーザーのスキーマにビューを作るには、CREATE ANY VIEW システム権限を持っていなければなりません。これらの権限は明示的に獲得するか、またはロールを介して獲得できます。
- ビューの所有者はビュー定義で参照するすべてのオブジェクトにアクセスするための権限を明示的に付与されていなければなりません。所有者は、ロールによってはそれらの権限を獲得できません。また、ビューの機能はビューの所有者の権限に依存します。たとえば、ビューの所有者に Scott の EMP 表の INSERT 権限しかない場合、EMP 表に新しい行を挿入するためにはビューを使えますが、このビューの行を選択 (SELECT)、更新 (UPDATE)、削除 (DELETE) するためには使えません。
- ビューの所有者がビューにアクセスする権限を他のユーザーに付与しようとする場合、基盤となるオブジェクトの GRANT OPTION 付きのオブジェクト権限、または ADMIN OPTION 付きのシステム権限を受け取っていなければなりません。

ビューは、SQL コマンド CREATE VIEW を使って作ります。ビューはそれぞれ、表またはスナップショット、その他のビューを参照する問合せによって定義されます。ビューを定義する問合せには、ORDER BY 句や FOR UPDATE 句を指定できません。たとえば、次の文は、EMP 表のデータのサブセットに対してビューを作ります。

```
CREATE VIEW sales_staff AS
  SELECT empno, ename, deptno
  FROM emp
  WHERE deptno = 10
  WITH CHECK OPTION CONSTRAINT sales_staff_cnst;
```

SALES_STAFF ビューを定義する問合せは、部門番号が 10 の行しか参照しません。さらに、CHECK OPTION によって作られるビューは、そのビューに対して発行される INSERT 文と

UPDATE 文の結果がビューが選択できないような行になってはならないという制約のあるビューです。たとえば、次の INSERT 文では、SALES_STAFF ビューによって EMP 表に行が正常に挿入されます。それらの行はすべて部門番号が 10 の行です。

```
INSERT INTO sales_staff VALUES (7584, 'OSTER', 10);
```

しかし、次の INSERT 文は、SALES_STAFF ビューを使って選択できない部門番号 30 の行を挿入しようとするため、その文はロールバックされ、エラーが戻されます。

```
INSERT INTO sales_staff VALUES (7591, 'WILLIAMS', 30);
```

次の文は、EMP 表と DEPT 表のデータを結合するビューを作ります。

```
CREATE VIEW division1_staff AS
  SELECT ename, empno, job, dname
  FROM emp, dept
  WHERE emp.deptno IN (10, 30)
  AND emp.deptno = dept.deptno;
```

DIVISION1_STAFF ビューは、EMP 表と DEPT 表の情報を結合するものです。このビューの CREATE VIEW 文には CHECK OPTION が指定されていません。

ビュー作成時の問合せ定義の展開

ANSI/ISO 規格に従って、ビューが作られるときに、Oracle はトップレベル・ビュー問合せのワイルドカードを列リストに展開し、結果の問合せをデータ・ディクショナリに格納します。副問合せはそのまま残されます。展開された列リスト中の列名は、基盤となるオブジェクトの列がもともと引用符付きで入力された可能性があり、問合せの構文を正しいものにするためには引用符が必要である、ということを示すために引用符で囲まれています。

たとえば、DEPT ビューが次のように作られる場合を想定します。

```
CREATE VIEW dept AS SELECT * FROM scott.dept;
```

Oracle は、DEPT ビューを定義している問合せを次のように格納します。

```
SELECT "DEPTNO", "DNAME", "LOC" FROM scott.dept
```

エラー付きで作られたビューではワイルドカードは展開されません。エラーなしでビューがコンパイルされると、Oracle は定義の問合せのワイルドカードを展開します。

エラー付きのビューを作る

CREATE VIEW 文に構文エラーがない場合、そのビューを定義している問合せを実行できなくても、Oracle はビューを作成できます。ビューは、「エラー付きで作られた」と見なされます。たとえば、存在しない表や既存の表の無効な列を参照するビューを作るとき、またはビューの所有者が必要な権限を持っていないときにも、ビューを作り、データ・ディクショナリに登録できます。ただし、そのビューは使えません。

エラー付きのビューを作るには、CREATE VIEW コマンドの FORCE オプションを指定しなければなりません。

```
CREATE FORCE VIEW AS ....;
```

特に何も指定しないと、ビューはエラー付きで作られません。ビューがエラー付きで作られると、そのことを示すメッセージが戻されます。エラー付きで作ったビューの状態は INVALID です。状況が変化して無効なビューの問合せを実行できるようになると、ビューは再コンパイルされ、有効（使用可能）になります。

関連項目：条件とビューに対する条件の影響の変更の詳細は、17-22 ページの「オブジェクトの依存性の管理」を参照してください。

結合ビューを変更する

変更可能な結合ビューとは、SELECT 文の上位の FROM 句に複数の表を含んでいて、かつ次のいずれも含まないビューのことです。

- DISTINCT 演算子
- 集約ファンクション：AVG、COUNT、GLB、MAX、MIN、STDDEV、SUM、VARIANCE
- 集合演算：UNION、UNION、ALL、INTERSECT、MINUS
- GROUP BY 句または HAVING 句
- START WITH 句または CONNECT BY 句
- ROWNUM 疑似列

制限がいくつかありますが、結合を伴うビューを変更できます。ビューが別のネストされたビュー上の結合である場合は、そのネストされたビューが上位のビューにマージできなければなりません。

次の項の例では、EMP 表と DEPT 表を使います。これらの例は、この 2 つの表に主キーと外部キーを明示的に定義するか、一意の索引を定義する場合にだけ当てはまります。次に示すのは、制約に適切に従った EMP と DEPT の表定義です。

```
CREATE TABLE dept (  
    deptno      NUMBER(4) PRIMARY KEY,  
    dname       VARCHAR2(14),  
    loc         VARCHAR2(13));
```

```
CREATE TABLE emp (  
    empno       NUMBER(4) PRIMARY KEY,  
    ename       VARCHAR2(10),  
    job         varchar2(9),  
    mgr         NUMBER(4),  
    sal         NUMBER(7,2),  
    comm        NUMBER(7,2),
```

```
deptno          NUMBER(2),
FOREIGN KEY(deptno) REFERENCES DEPT(deptno));
```

また、前述の主キーと外部キーの制約を省略し、DEPT (DEPTNO) に UNIQUE INDEX を作って、次の例を適用することもできます。

関連項目：マージ可能なビューの詳細は、『Oracle8 Server チューニング』を参照してください。

キー保存表

キー保存表の概念は、結合ビューを更新する上での制限を理解するために重要なものです。表のすべてのキーが結合の結果のキーでもある場合、その表はキー保存です。つまり、キー保存表は、結合後もそのキーを保存します。

注意： 表の 1 つまたは複数のキーをキー保存として選択する必要はありません。1 つまたは複数のキーが選択され、そのキーが結合の結果のキーであれば十分です。

表のキー保存特性は、表の実際のデータには依存しません。これはそのスキーマの特性であって、表内のデータの特性ではありません。たとえば、EMP 表で各部門に最高でも 1 人の従業員しか含まれていない場合、EMP と DEPT の結合の結果では DEPT.DEPTNO は一意ですが、DEPT はキー保存表ではありません。

EMP_DEPT_VIEW からすべての行を選択すると、結果は次のようになります。

EMPNO	ENAME	DEPTNO	DNAME	LOC
7782	CLARK	10	ACCOUNTING	NEW YORK
7839	KING	10	ACCOUNTING	NEW YORK
7934	MILLER	10	ACCOUNTING	NEW YORK
7369	SMITH	20	RESEARCH	DALLAS
7876	ADAMS	20	RESEARCH	DALLAS
7902	FORD	20	RESEARCH	DALLAS
7788	SCOTT	20	RESEARCH	DALLAS
7566	JONES	20	RESEARCH	DALLAS

8 rows selected.

このビューでは、EMPNO は表 EMP のキーであり、結合結果のキーでもあるので、EMP はキー保存表となります。DEPT がキー保存表でないのは、DEPTNO が DEPT 表のキーであっても結合のキーではないからです。

DML 文および結合ビュー

結合ビューにおいて、どのような UPDATE または INSERT、DELETE 文でも、基礎を形成する実表を 1 つだけ変更できます。

UPDATE 文 次の例は、EMP_DEPT ビューを正常に変更する UPDATE 文を示したものです。

```
UPDATE emp_dept
  SET sal = sal * 1.10
  WHERE deptno = 10;
```

次に示す UPDATE 文は、EMP_DEPT ビューには使えません。

```
UPDATE emp_dept
  SET loc = 'BOSTON'
  WHERE ename = 'SMITH';
```

この文は基礎を形成する DEPT 表を変更しようとしませんが、DEPT 表は EMP_DEPT ビューに保存されたキーではないため、ORA-01779 エラー「複数表にマップする列を変更できません。」が発生し、文を実行できません。

一般に、結合ビューの変更可能な列はすべて、キー保存表の列にマップしなければなりません。ビューの定義に WITH CHECK OPTION 句が使われている場合は、繰り返される表の結合列および列はいずれも変更できません。

したがって、たとえば EMP_DEPT ビューの定義に WITH CHECK OPTION が使われている場合、次に示す UPDATE 文は失敗します。

```
UPDATE emp_dept
  SET deptno = 10
  WHERE ename = 'SMITH';
```

結合列を更新しようとするため、この文は失敗となります。

DELETE 文 結合の中にキー保存表が 1 つしかない場合には、結合ビューから削除できます。

次の DELETE 文は、EMP_DEPT ビューを対象にしたものです。

```
DELETE FROM emp_dept
  WHERE ename = 'SMITH';
```

EMP_DEPT ビューに対するこの DELETE 文は、実表 EMP に対する DELETE 操作に変換でき、表 EMP は結合ビュー内の唯一のキー保存表であるため、この文は正当です。

次のビューでは E1 と E2 がともにキー保存表であるため、このビューに対して DELETE 操作を実行できません。

```
CREATE VIEW emp_emp AS
  SELECT e1.ename, e2.empno, deptno
  FROM emp e1, emp e2
  WHERE e1.empno = e2.empno;
```

ビューの定義に WITH CHECK OPTION 句が使われていて、キー保存表が繰り返される場合、そのビューから行を削除できません。

```
CREATE VIEW emp_mgr AS
```

```
SELECT e1.ename, e2.ename mname
FROM emp e1, emp e2
WHERE e1.mgr = e2.empno
WITH CHECK OPTION;
```

このビューはキー保存される表の内部結合を伴うので、このビューに対して削除を実行できません。

INSERT 文 EMP_DEPT ビューに対する次の INSERT 文は実行されます。

```
INSERT INTO emp_dept (ename, empno, deptno)
VALUES ('KURODA', 9010, 40);
```

変更されるキー保存実表は 1 つだけであり (EMP) 40 は DEPT 表の有効な DEPTNO であるため、(したがって EMP 表に対する FOREIGN KEY 整合性制約を満たすので) この文は実行されます。

次に示すような INSERT 文は、実表 EMP に対する UPDATE が失敗するのと同じ理由、つまり、EMP 表に対する FOREIGN KEY 整合性制約違反のために、実行できません。

```
INSERT INTO emp_dept (ename, empno, deptno)
VALUES ('KURODA', 9010, 77);
```

次の INSERT 文は ORA-01776 エラー「結合ビューを介して複数の基本表を変更できません。」によって失敗します。

```
INSERT INTO emp_dept (empno, ename, loc)
VALUES (9010, 'KURODA', 'BOSTON');
```

INSERT は、暗黙的にも明示的にも、非キー保存表の列を参照できません。結合ビューの定義に WITH CHECK OPTION 句が使われている場合は、その結合ビューに対して INSERT を実行できません。

UPDATABLE_COLUMNS ビューを使う

表 13-1 で説明するビューは、結合ビューを変更する際に使うと便利です。

表 13-1 UPDATABLE_COLUMNS ビュー

ビュー名	説明
USER_UPDATABLE_COLUMNS	ユーザーのスキーマ内で変更可能なすべての表とビューの列をすべて表示する。
DBA_UPDATABLE_COLUMNS	DBA スキーマ内で変更可能なすべての表とビューの列をすべて表示する。
ALL_UPDATABLE_VIEWS	変更可能なすべての表とビューの列をすべて表示する。

ビューを置き換える

ビューを置き換えるには、ビューの削除および作成に必要なすべての権限を持っていなければなりません。ビューの定義を変更しなければならない場合は、そのビューを置き換えなければなりません。つまり、ビューの定義を変更できず、次の方法でビューを置き換えることができます。

- ビューを削除してから再作成します。

警告： ビューを削除するときに、ロールおよびユーザーに付与された対応するオブジェクト権限はすべて取り消されます。ビューを作りなおしてから、権限を再付与しなければなりません。

- OR REPLACE オプションを含む CREATE VIEW 文によって、ビューを再定義します。OR REPLACE オプションは、ビューの現行の定義を置き換え、現行のセキュリティ認可を保存します。たとえば、先の例で示されたように、SALES_STAFF ビューを作り、いくつかのオブジェクト権限をロールと他のユーザーに付与する場合を想定します。ただし、ここでは SALES_STAFF ビューを再定義して、WHERE 句に指定されている部門番号を変更する必要があります。次の文によって、SALES_STAFF ビューの現行バージョンを置き換えることができます。

```
CREATE OR REPLACE VIEW sales_staff AS
    SELECT empno, ename, deptno
    FROM emp
    WHERE deptno = 30
    WITH CHECK OPTION CONSTRAINT sales_staff_cnst;
```

ビューを置き換える前に、次の影響を検討してください。

- ビューを置き換えることによって、データ・ディクショナリ内のビュー定義が置き換えられる。ビューによって参照される基礎となるオブジェクトは影響を受けません。
- 以前 CHECK OPTION に制約が定義されていたが、新しいビュー定義には指定されない場合、その制約は削除される。
- 置き換えられたビューに依存するビューと PL/SQL プログラム・ユニットはすべて無効（使用不可能）になる。Oracle でオブジェクトの依存性を管理する方法に関する詳細は、17-22 ページの「オブジェクトの依存性の管理」を参照してください。

ビューを削除する

自分のスキーマにあるビューはすべて削除できます。別のユーザーのスキーマ内のビューを削除するには、DROP ANY VIEW システム権限を持っていなければなりません。ビューは SQL コマンド DROP VIEW を使って削除します。たとえば、次の文は SALES_STAFF ビューを削除します。

```
DROP VIEW sales_staff;
```

順序の管理

ここでは、順序を管理する方法について説明します。トピックは次のとおりです。

- 順序を作成する
- 順序を変更する
- 順序に影響を及ぼす初期化パラメータ
- 順序を削除する

順序を作成する

自分のスキーマに順序を作成するには、CREATE SEQUENCE システム権限を持っていなければなりません。別のユーザーのスキーマに順序を作成するには、CREATE ANY SEQUENCE 権限を持っていなければなりません。順序は、SQL コマンド CREATE SEQUENCE を使って作ります。たとえば、次の文は、EMP 表の EMPNO 列に対して従業員番号を生成するために使う順序を作ります。

```
CREATE SEQUENCE emp_sequence  
    INCREMENT BY 1  
    START WITH 1  
    NOMAXVALUE  
    NOCYCLE  
    CACHE 10;
```

CACHE オプションは順序番号をより速くアクセスできるように、順序番号の集合をメモリーに事前に割り当て、維持します。キャッシュ内の最後の順序番号が使われると、別の順序の集合がキャッシュ内に読み込まれます。

順序番号の集合をキャッシュする場合、順序番号がスキップされる可能性があることを承知しておいてください。たとえば、インスタンスが異常停止すると（たとえばインスタンス障害が発生したり、SHUTDOWN ABORT 文が発行されたりすると）、キャッシュされているが使われていない順序番号は失われます。また、使われたにもかかわらず保存されなかった順序番号も失われます。また、エクスポートとインポートの後、Oracle がキャッシュされた順序番号をスキップすることもあります。詳細は、*Oracle8 Server ユーティリティ*を参照してください。

関連項目：Oracle Parallel Server がどのようにキャッシュされた順序番号に影響を与えるかの詳細は、『Oracle8 Parallel Server 概要および管理』を参照してください。

順序番号をキャッシュに格納する際のパフォーマンスの詳細は、『Oracle8 Server チューニング』を参照してください。

順序を変更する

順序を変更するには、その順序が自分のスキーマに含まれているか、または ALTER ANY SEQUENCE システム権限を持っていないければなりません。順序を変更することにより、順序番号の生成方法を定義するパラメータを変更できます。ただし、順序の開始番号は変更できません。順序の開始点を変更するには、順序を削除してから、作りなおさなければなりません。順序番号で DDL を実行すると、キャッシュ値が削除されます。

順序は SQL コマンド ALTER SEQUENCE を使って変更します。たとえば、次の文は、EMP_SEQUENCE を変更します。

```
ALTER SEQUENCE emp_sequence
    INCREMENT BY 10
    MAXVALUE 10000
    CYCLE
    CACHE 20;
```

順序に影響を及ぼす初期化パラメータ

初期化パラメータ SEQUENCE_CACHE_ENTRIES は、キャッシュされる順序の数を設定します。監査が使用可能になっている場合は、監査セッション番号を識別するために追加の順序を 1 つ見越しておくといよいでしょう。

SEQUENCE_CACHE_ENTRIES の値が小さすぎる場合、次のようなシナリオで順序値がスキップされる可能性があります。キャッシュされる順序を 5 つ使い、キャッシュは満杯で、SEQUENCE_CACHE_ENTRIES は 4 に設定されているとします。現在 4 つの順序がキャッシュされている場合、5 番目の順序は、キャッシュ内で最も前に使用された順序に置き換わり、追い出された順序の中に残っていた値はすべて（キャッシュされた最後の順序番号まで）失われます。

順序を削除する

自分のスキーマ内の順序はどれでも削除できます。別のスキーマ内の順序を削除するには、DROP ANY SEQUENCE システム権限を持っていないければなりません。必要なくなった順序は、SQL コマンド DROP SEQUENCE を使って削除できます。たとえば、次の文は ORDER_SEQ 順序を削除します。

```
DROP SEQUENCE order_seq;
```

順序が削除されると、その定義がデータ・ディクショナリから削除されます。シノニムはそのまま残りますが、参照時にエラーが戻されます。

シノニムの管理

パブリック・シノニムとプライベート・シノニムの両方を作成できます。パブリック・シノニムは PUBLIC という名前の特別なユーザー・グループによって所有され、データベース内のすべてのユーザーがアクセスできます。プライベート・シノニムは、特定のユーザーのスキーマ内に含まれ、そのユーザーとそのユーザーの権限受領者だけが使えます。

ここでは、次のようなシノニム管理の情報について説明します。

- シノニムを作成する
- シノニムを削除する

シノニムを作成する

自分のスキーマ内にプライベート・シノニムを作るには、CREATE SYNONYM 権限を持っていなければなりません。また、別のユーザーのスキーマにプライベート・シノニムを作るには、CREATE ANY SYNONYM 権限を持っていなければなりません。パブリック・シノニムを作るには、CREATE PUBLIC SYNONYM システム権限を持っていなければなりません。

シノニムは、SQL コマンド CREATE SYNONYM を使って作ります。たとえば、次の文は JWARD のスキーマに含まれる EMP 表のパブリック・シノニム PUBLIC_EMP を作ります。

```
CREATE PUBLIC SYNONYM public_emp FOR jward.emp;
```

シノニムを削除する

自分のスキーマ内のプライベート・シノニムはどれでも削除できます。別のユーザーのスキーマ内のプライベート・シノニムを削除するには、DROP ANY SYNONYM システム権限を持っていなければなりません。またパブリック・シノニムを削除するには、DROP PUBLIC SYNONYM システム権限を持っていなければなりません。

SQL コマンド DROP SYNONYM を使って、必要なくなったシノニムを削除できます。プライベート・シノニムを削除するには、PUBLIC キーワードを指定しないでください。またパブリック・シノニムを削除するには、PUBLIC キーワードを指定してください。

たとえば、次の文はプライベート・シノニム EMP を削除します。

```
DROP SYNONYM emp;
```

次の文はパブリック・シノニム PUBLIC_EMP を削除します。

```
DROP PUBLIC SYNONYM public_emp;
```

シノニムが削除されると、その定義がデータ・ディクショナリから削除されます。削除されたシノニムを参照するオブジェクトはすべてそのまま残ります。ただしそれらは無効（使用不可能）になります。

関連項目：シノニムを削除することにより他のスキーマ・オブジェクトに対する影響の詳細は、17-22 ページの「オブジェクトの依存性の管理」を参照してください。

この章では、索引の管理について説明します。トピックは次のとおりです。

- 索引を管理するためのガイドライン
- 索引の作成
- 索引の変更
- 索引の領域使用を監視
- 索引の削除

この章で説明する作業を実行する前に、第 10 章「スキーマ・オブジェクトを管理するためのガイドライン」の内容をよく理解しておいてください。

索引を管理するためのガイドライン

ここでは、索引の管理のガイドラインについて説明します。トピックは次のとおりです。

- 表データ挿入後に索引を作成する
- 表あたりの索引の数を制限する
- 各索引の表領域を指定する
- トランザクション・エントリ・パラメータを指定する
- 索引ブロックの領域使用を指定する
- 索引作成を並行化する
- 回復不能索引作成に関する考慮事項
- 索引サイズの見積もりと記憶領域パラメータの設定

索引とは、表とクラスタに対応付けられたオプションの構造体であり、明示的に作ることによって、表に対する SQL 文の実行速度を上げることができます。このマニュアルに索引が付いていることによって情報を速く検索できるのと同じように、Oracle の索引は表データに対する高速アクセス経路を提供します。

索引の有無によって SQL 文を変更する必要はありません。索引はデータに対する高速アクセス経路を提供するだけであり、実行の速度にだけ影響します。データ値に索引が付いている場合、その索引によって、その値を含む行の位置が直接指示されます。

索引は、対応付けられた表内のデータから論理的にも物理的にも独立しています。索引は、実表またはその他の索引に影響を与えることなく、いつでも作成または削除できます。索引を削除してもアプリケーションはすべて実行を続けますが、それまで索引が付いていたデータのアクセスは遅くなります。索引は、独立した構造体であり、記憶領域を必要とします。

索引の作成後は、Oracle は自動的に索引をメンテナンスし、使います。新しい行の追加、行の更新、行の削除といったデータに対する変更は、関連するすべての索引に自動的に反映され、ユーザーは何もする必要はありません。

関連項目：索引の作成のパフォーマンスの意味に関する詳細は、『Oracle8 Server チューニング』を参照してください。

索引の詳細は、『Oracle8 Server 概要』を参照してください。

表データ挿入後に索引を作成する

表の索引は、(SQL*Loader または Import ユーティリティを使って) データを表に挿入またはロードした後で作ります。索引のない表にデータ行を挿入してから、それ以降のアクセスのために索引を作るほうが効率的です。表データをロードする前に索引を作ると、表に行が挿入されるたびに索引すべてを更新しなければなりません。また、クラスタの索引は、そのクラスタにデータを挿入する前に作らなければなりません。

すでにデータを持っている表に索引を作るには、Oracle はソート領域を使わなければなりません。Oracle は索引の作成者に割り当てられたメモリー内のソート領域（初期化パラメータ SORT_AREA_SIZE で決まるユーザーあたりの容量）を使いますが、索引作成のために割り当てられた一時セグメントとソート情報もスワップしなければなりません。

索引が極端に大きい場合は、次の手順を実行すると便利です。

大きな索引を管理する手順

1. 一時セグメント表領域を新しく作る。
2. 索引作成者の一時セグメント表領域を変更する。
3. 索引を作る。
4. 一時セグメント表領域を削除し、必要であれば、作成者の一時セグメント表領域を再指定する。

関連項目：特定の条件下では、SQL*Loader の「ダイレクト・パス・ロード」を使ってデータを表にロードできるので、データをロードしながら索引を作成できます。詳細は、『Oracle8 Server ユーティリティ』を参照してください。

表あたりの索引の数を制限する

表は、かなり多数の索引を持つことができます。ただし、索引の数が多いほど、表を修正するときに発生するオーバーヘッドが増加します。特に、行を挿入したり削除したりするときは、その表の索引もすべて更新しなければなりません。また、列を更新するときには、その列を含む索引もすべて更新しなければなりません。

したがって、表からデータを検索する速度とその表を更新する速度の間に妥協点があります。たとえば、表が主に読取り専用であれば、索引を増やすと有効ですが、表がかなり頻繁に更新されるのであれば、索引を少なくするほうが望ましいでしょう。

トランザクション・エントリ・パラメータを指定する

各索引を作るときに INITRANS パラメータと MAXTRANS パラメータを指定することによって、索引のセグメントのデータ・ブロック内のトランザクション・エントリに、最初に割り当てられる領域とその後割り当てられる領域を調整できます。

関連項目：これらのパラメータの詳細は、10-7 ページの「記憶領域パラメータの設定」を参照してください。

索引ブロックの領域使用を指定する

表の索引が作られるとき、その索引のデータ・ブロックはその表にある既存の値を使って PCTFREE まで満たされます。索引ブロック用に PCTFREE によって確保されている領域は、表に新しい行が挿入されたため、その行に対応する索引エントリを正しい索引ブロック（前の索引エントリと次の索引エントリの間）に入れなければならないときにだけ使われます。該当する索引ブロックにそれ以上領域がない場合は、索引の値は別の索引ブロックに入れら

れます。対応する索引ブロック内に領域がない場合、索引を付けられた値は別の索引ブロックに入れられます。そのため、索引を付けられた表に多くの行を挿入する予定であれば、PCTFREE は新しい索引値を収容するために大きくする必要があります。また、あまり挿入されず、表が相対的に静的である場合、索引データを保持するために必要なブロックが少なくなるように、対応する索引の PCTFREE を小さくできます。

関連項目：PCTUSED は索引には指定できません。PCTFREE パラメータの詳細は、10-2 ページの「データ・ブロックの領域管理」を参照してください。

各索引の表領域を指定する

索引はどの表領域にも作成できます。索引は、その索引を付けた表と同じ表領域、または異なる表領域に作成できます。

表とその索引に対して同じ表領域を使うと、データベースのメンテナンス（表領域バックアップまたはファイル・バックアップ、アプリケーション可用性や更新など）が容易になるかもしれません。この場合、関連するすべてのデータが常にまとめてオンラインになります。

表とその索引に対して異なる表領域（異なるディスク上）を使うと、表と索引を同じ表領域に格納するよりもディスクの競合が低減するために、よりよいパフォーマンスが得られます。

表とその索引に対して異なる表領域を使い、一方の（データまたは索引のいずれかを含む）表領域がオフラインになっている場合には、その表を参照している文が作動する保証はありません。

索引作成を並行化する

パラレル問合せオプションをインストールしている場合は、索引の作成を並行化できます。複数のプロセスが同時に動作して索引を作るため、1つのサーバー・プロセスが順に索引を作る場合よりも Oracle の索引作成の速度が速くなります。

索引を並行して作る場合、問合せサーバー・プロセスごとに別々の記憶領域パラメータが使われます。したがって、INITIAL が 5M、PARALLEL DEGREE が 12 で作られた索引は、索引を作るときに 60MB 以上の記憶領域を使います。

関連項目：パラレル問合せオプションとパラレル索引作成の詳細は、『Oracle8 Server チューニング』を参照してください。

回復不能索引作成に関する考慮事項

CREATE INDEX 文で UNRECOVERABLE を指定すると、REDO ログ・レコードを生成せずに索引を作成できます。

注意： 回復不能として作られた索引はアーカイブされないため、索引を作った後でバックアップする必要があります。

索引を回復不能として作ると、次のような利点があります。

- REDO ログ・ファイルの領域を節約できる。
- 索引を作るのに要する時間が削減できる。
- 大規模な索引の平行作成のパフォーマンスが向上する。

一般に、表を回復不能として作ったとき、小規模な表の場合より大規模な表の方が相対的にパフォーマンスの向上が大きくなります。小規模な表を回復不能として作っても、表作成に要する時間にほとんど影響はありません。一方、大規模な表では、特に表作成を並行化したときに、パフォーマンスが著しく向上します。

索引サイズの見積もりと記憶領域パラメータの設定

付録 A「スキーマ・オブジェクトの領域の見積もり」に、索引サイズの見積もりに役立つ公式が示されています。

次のような理由で、索引を作る前に索引のサイズを見積もると有効です。

- 表、ロールバック・セグメント、REDO ログ・ファイルの見積もりと合わせた表の見積もりサイズを使って、作るデータベースを保持するために必要なディスク容量を決定できる。この見積もりを利用して適切なハードウェアを購入したりできます。
- 個々の索引の見積もりサイズを使うと、索引が使うディスク領域をよりよく管理できる。索引を作るときに、適切な記憶領域パラメータを設定し、その索引を使うアプリケーションの I/O パフォーマンスを向上させることができます。

たとえば、表を作る前に、表の最大サイズを見積もることができる場合を想定します。表を作るときに記憶領域パラメータを設定すると、その表のデータ・セグメントに割り当てるエクステントを少なくできます。そのため、表のデータすべてが比較的ディスク領域の連続した部分に格納されます。これによって、この表を呼び出すディスク I/O 操作に要する時間が短くなります。

単一の索引エントリの最大サイズは、データ・ブロック・サイズからある適当なオーバーヘッドを除いた分のおよそ 2 分の 1 程度になります。

表と同じように、索引を作るときにも、記憶領域パラメータを明示的に設定できます。索引の記憶領域パラメータを明示的に設定する場合、エクステントを小さくしてその数を増やすよりも、エクステントを大きくしてその数を少なくするように、索引のデータを格納すべきであることに注意してください。

関連項目：記憶領域パラメータの詳細は、10-7 ページの「記憶領域パラメータの設定」を参照してください。

索引サイズを見積もる方法の詳細は、付録 A「スキーマ・オブジェクトの領域の見積もり」を参照してください。

制約を使用禁止または削除する前の検討

一意キーと主キーには対応する索引があるため、UNIQUE キー制約や PRIMARY KEY 制約を使用禁止または削除するかどうかを検討するときには、索引の削除と作成にかかわるコストを分析してください。また、UNIQUE キー制約や PRIMARY KEY 制約に対する索引がかなり大きい場合には、その索引を削除して作りなおすよりも、その制約を使用可能な状態のままにして時間を節約できます。

索引の作成

ここでは、索引を作る方法について説明します。トピックは次のとおりです。

- 制約に対応付けられる索引を作成する
- 索引を明示的に作成する
- 既存の索引を作成しなおす

(対応付けられる索引を作る) UNIQUE キーまたは PRIMARY KEY を使用可能にするには、表の所有者に、索引を収録する表領域に対する割当て制限、または UNLIMITED TABLESPACE システム権限が必要です。

LONG 列および LONG RAW 列に索引を付けることはできません。

Oracle は、一意キーまたは主キーに一意索引をすることによって、UNIQUE キー整合性制約または PRIMARY KEY 整合性制約を施行します。制約を使用可能にすると、この索引は Oracle によって自動的に作られます。つまり、CREATE TABLE 文や ALTER TABLE 文を発行するユーザーは、索引を作るために何もする必要はありません。制約を定義し使用可能にすると、および定義したが使用禁止にしていた制約を使用可能にするときの両方で、このようなことがあてはまります。

一般に、一意性を施行するには、CREATE UNIQUE INDEX 構文を使うよりも、制約を作るほうがよいでしょう。制約に対応する索引は、常にその制約の名前を想定しています。つまり、制約索引に固有の名前は指定できません。

索引に記憶オプション (INITIAL と NEXT など) を指定しないと、ホスト表領域のデフォルトの記憶オプションが自動的に使われます。

制約に対応付けられる索引を作成する

USING INDEX オプションを指定した ENABLE 句を使うと、UNIQUE キー制約または PRIMARY KEY 制約に対応する索引の記憶オプションを指定できます。次の文は、PRIMARY KEY 制約を定義し、対応する索引の記憶オプションを指定します。

```
CREATE TABLE emp (
    empno NUMBER(5) PRIMARY KEY, . . . )
    ENABLE PRIMARY KEY USING INDEX
    TABLESPACE users
    PCTFREE 0;
```

索引を明示的に作成する

SQL コマンド CREATE INDEX を使って、索引を明示的に（整合性制約外に）作ります。次の文は、EMP 表の ENAME 列に索引 EMP_ENAME を作ります。

```
CREATE INDEX emp_ename ON emp(ename)
    TABLESPACE users
    STORAGE (INITIAL 20K
    NEXT 20k
    PCTINCREASE 75)
    PCTFREE 0;
```

複数の記憶設定が、索引に対して明示的に指定されています。

既存の索引を作成しなおす

既存の索引をデータ・ソースとして使って索引を作成できます。このようにして索引を作ると、記憶特性の変更や新たな表領域への移動ができます。既存のデータ・ソースに基づいて索引を作りなおすと、ブロック内の断片化もなくなります。実際には、索引を削除して CREATE INDEX コマンドを使うより、既存の索引を作りなおしたほうがパフォーマンスはよくなります。

既存の索引を作りなおすには、次の文を発行します。

```
ALTER INDEX index name REBUILD;
```

REBUILD 句は索引名の直後、他のオプションの前になければなりません。また、REBUILD 句は DEALLOCATE STORAGE 句と一緒に使えません。

関連項目： ALTER INDEX コマンドとオプションの句の詳細は、『Oracle8 Server SQL リファレンス』を参照してください。

索引の変更

索引を変更するには、その索引が自分のスキーマに含まれているか、または ALTER ANY INDEX システム権限を持っていなければなりません。索引は、トランザクション・エントリ・パラメータを変更したり、記憶領域パラメータを変更する場合以外は変更できません。索引の列構造は変更できません。

主キーと一意キーの整合性制約を施行するために、Oracle によって作られる索引も含めて、どの索引の記憶領域パラメータも、SQL コマンド ALTER INDEX を使って変更できます。たとえば、次の文は EMP_ENAME 索引を変更します。

```
ALTER INDEX emp_ename
  INITRANS 5
  MAXTRANS 10
  STORAGE (PCTINCREASE 50);
```

索引のトランザクション・エントリ設定 (INITRANS、MAXTRANS) を変更するときには、MAXTRANS の新しい設定が索引のすべてのデータ・ブロック (すでに割り当てられたブロックとその後割り当てられるブロック) に適用される一方、INITRAN の新しい設定はその後割り当てられるブロックだけに適用されます。

記憶領域パラメータ INITIAL と MINEXTENTS は変更できません。他の記憶領域パラメータの新しい設定はすべて、その後索引に割り当てられるエクステンツにしか影響しません。

整合性制約をインプリメントする索引では、USING INDEX オプションを指定した ENABLE 句を含む ALTER TABLE 文を発行することによって、記憶領域パラメータの調整もできます。たとえば、次の文は先に定義した索引の記憶オプションを変更します。

```
ALTER TABLE emp
  ENABLE PRIMARY KEY USING INDEX
  PCTFREE 5;
```

索引の領域使用を監視

索引のキー値を頻繁に挿入、更新、削除する場合、索引がその獲得した領域を効果的に何度も使うかどうかはわかりません。そこで、最初に索引の構造を分析することによって、定期的な間隔で索引の領域使用の効率を監視した後、次のように INDEX_STATS ビューを問い合わせるとよいでしょう。

```
SELECT pct_used FROM sys.index_stats WHERE name = 'indexname';
```

索引の領域使用の割合は、どれくらい頻繁に索引キーが更新され、挿入され、削除されるかによって変化します。次の一連の操作を何度も実行することによって、索引の領域使用の平均の効率の履歴を作ってください。つまり、索引を検証する、および PCT_USED をチェックする、索引を削除してから作りなおす操作を繰り返します。索引の領域使用がその平均を下回るときは、索引を削除してから作りなおす (再構築する) ことによって、索引の領域を圧縮できます。

関連項目：索引の構造の分析の詳細は、17-3 ページの「表、索引、クラスタの分析」を参照してください。

索引の削除

索引を削除するには、その索引が自分のスキーマに含まれているか、または DROP ANY INDEX システム権限を持っていなければなりません。

索引を削除する理由には、次のようなものがあります。

- 索引が必要ではなくなった。
- 対応する表に対して発行した問合せで、索引が予想されたパフォーマンス改善をもたらしていない。(たとえば、表が非常に小さい、または表には多くの行があるが、索引エントリが非常に少ないなど)
- アプリケーションに索引を使う問合せが含まれない。
- 索引が無効になり、再構築する前に削除しなければならない。
- 索引がかなり断片化し、再構築する前に削除しなければならない。

索引が削除されると、その索引のセグメントのエクステントはすべて、索引を含んでいる表領域に戻され、表領域内の他のオブジェクトで利用できます。

索引を削除する方法は、CREATE INDEX コマンドによって索引を明示的に作ったか、または表にキー制約を定義することによって索引を暗黙に作ったかという索引の作成方法に依存します。

注意： 表が削除されると、対応する索引はすべて自動的に削除されます。

使用可能になっている UNIQUE キー制約や PRIMARY KEY 制約に対応付けられた索引だけの削除はできません。制約に対応付けられた索引を削除するには、制約自体を使用禁止にするか、または削除しなければなりません。

```
DROP INDEX emp_ename;
```

関連項目：索引の分析の詳細は、17-3 ページの「表、索引、クラスタの分析」を参照してください。

制約に対応する索引の削除の詳細は、17-12 ページの「整合性制約の管理」を参照してください。

クラスタの管理

この章では、クラスタ（クラスタ化された表と索引を含む）を管理する方法について説明します。トピックは次のとおりです。

- クラスタを管理するためのガイドライン
- クラスタの作成
- クラスタの変更
- クラスタの削除

この章で説明する作業を実行する前に、第 10 章「スキーマ・オブジェクトを管理するためのガイドライン」の内容をよく理解しておいてください。

クラスタを管理するためのガイドライン

クラスタは、表データを格納するオプションとしての方法を提供します。クラスタは同じデータ・ブロックを共有する表のグループで構成されており、そのデータブロックは共通の列を共有し、一緒に使われることが多いためにグループ化されています。たとえば、EMP 表および DEPT 表は DEPTNO 列を共有しています。EMP 表および DEPT 表をクラスタ化する場合（図 15-1 を参照）EMP 表および DEPT 表の各部門の行はすべて、物理的に同じデータ・ブロックに格納されます。別々にアクセスされることの多い表には、クラスタを使わないでください。

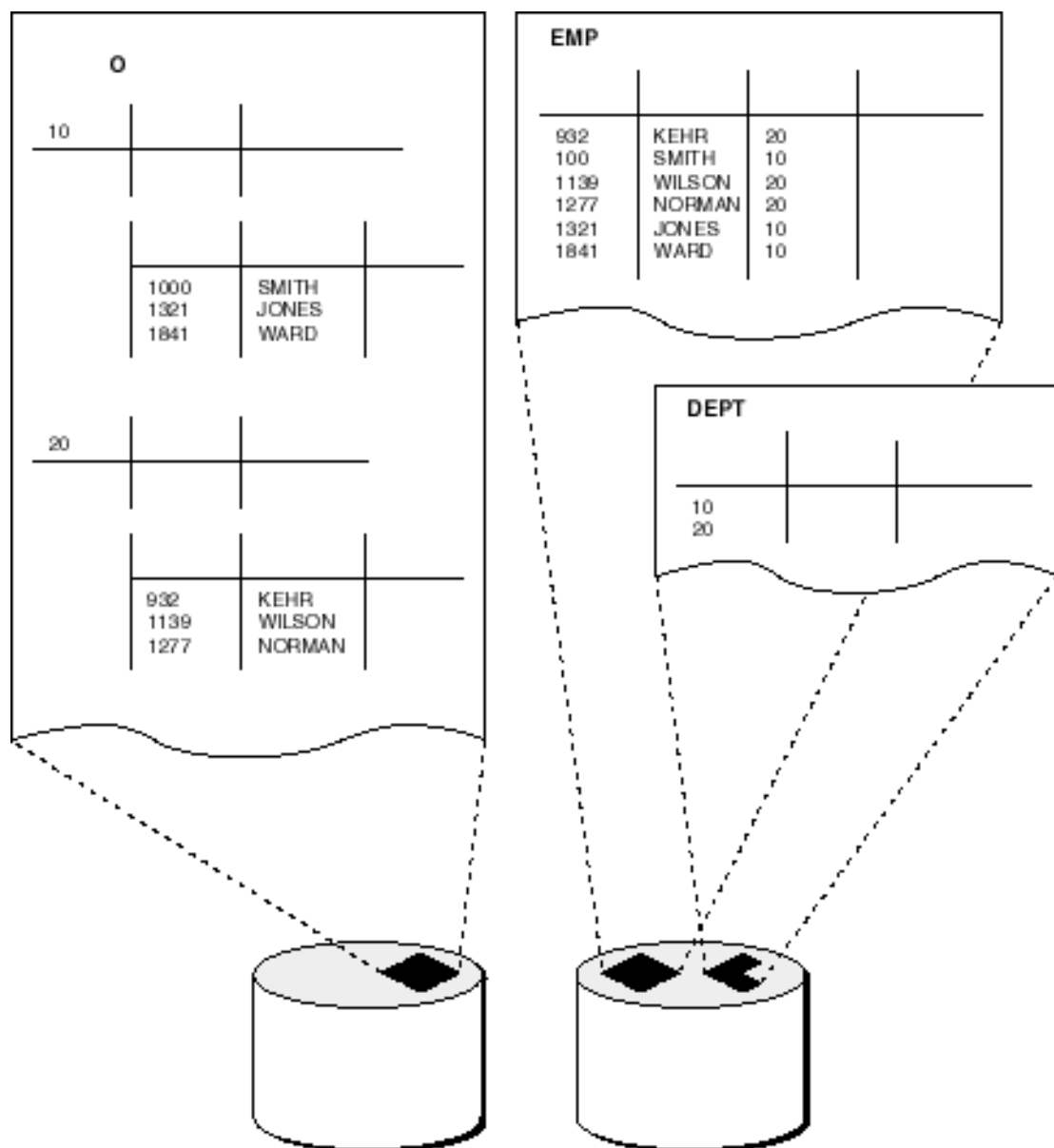
クラスタは別々の表の関連する行を同じデータ・ブロックに格納するため、クラスタを正しく使うと利点として主に次の 2 つがあります。

- クラスタ化した表の結合のため、ディスク I/O が減少し、アクセス時間が改善される。
- クラスタ・キーは、クラスタ化した表が共通に持っている列または列のグループである。まずクラスタを作るときに、クラスタ・キーの列を指定します。その後、そのクラスタに追加する表を作るたびに同じ列を指定します。各クラスタ・キーの値は、異なる表のどれだけ多くの行にその値が含まれているかに関係なく、クラスタとクラスタ索引のそれぞれに 1 度だけ格納されます。

このため、関連する表および索引データをクラスタに格納するために必要な記憶領域は、クラスタ化されてない表形式の場合に比べて少なくて済みます。これは、たとえば各クラスタ・キー（各 DEPTNO）が EMP 表および DEPT 表の両方の中の同じ値を含んでいる多数の行について 1 度だけ格納される、ということを考えればよくわかります。

クラスタを作った後で、そのクラスタ内に表を作れます。しかし、クラスタ化した表に行を挿入する前には、クラスタ索引を作らなければなりません。クラスタを使っても、クラスタ化した表に対してさらに索引を作ることは影響しません。普通に作成、削除できます。

図 15-1 クラスタ化表データ



次の部分では、クラスタを管理する際に考慮すべきガイドラインを説明します。トピックは次のとおりです。

- 適切な表のクラスタ化
- クラスタ・キーに対する適切な列の選択
- データ・ブロック領域使用の指定
- 平均クラスタ・キーとその対応行が必要とする領域の指定
- 各クラスタとクラスタ索引の位置の指定
- クラスタ・サイズの見積もりおよび記憶領域パラメータの設定

関連項目：クラスタの詳細は、『Oracle8 Server 概要』を参照してください。

適切な表のクラスタ化

問合せを主とした（挿入または更新をあまりしない）1 つ以上の表を格納するため、その問合せが頻繁にクラスタ内の複数の表のデータを結合したり、またはその単一の表から関連したデータを検索したりする場合に、クラスタを使ってください。

クラスタ・キーに対する適切な列の選択

クラスタ・キー列を慎重に選択してください。表を結合する問合せで、複数の列が使われる場合、クラスタ・キーを複合キーにしてください。一般に、よいクラスタ索引を表す特性は、よい索引を表す特性と同じです。

各キー値に対応している行のグループがおおよそ1つのデータ・ブロックを満杯にするように、よいクラスタ・キーは十分な一意の値を持ちます。クラスタ・キー値あたりの行が少なすぎると、領域を浪費し、パフォーマンスが低下する結果となります。少しの行だけが共通の値を共有するような特性のクラスタ・キーは、クラスタを作るときに小さなサイズを指定しない限り、ブロック内の領域を浪費する可能性があります（下記参照）。

クラスタ・キー値あたりの行が多すぎると、そのキーに対する行を見つけるために余分な検索が起こる可能性があります。あまりに一般的な値（たとえば性別）に対するクラスタ・キーは、過度の検索を生じ、クラスタ化していないものよりもパフォーマンスが低下する可能性があります。

クラスタ索引は一意にできません。また、LONG として定義した列を含むことはできません。

関連項目：よい索引の特性の詳細は、14-2 ページの「索引を管理するためのガイドライン」を参照してください。

データ・ブロック領域使用の指定

クラスタを作るときに、PCTFREE パラメータと PCTUSED パラメータを指定することによって、領域使用、および現在行への更新のために、クラスタのデータ・セグメントのデータ・ブロック内に確保される領域を変化させることができます。クラスタ内の表に設定された PCTFREE パラメータと PCTUSED パラメータは、無視されることに注意してください。つまり、クラスタ化した表はクラスタの設定を自動的に使います。

関連項目：PCTFREE と PCTUSED の指定の詳細は、10-2 ページの「データ・ブロックの領域管理」を参照してください。

平均クラスタ・キーとその対応行が必要とする領域の指定

CREATE CLUSTER コマンドにはオプションの引数 SIZE があります。これは、平均クラスタ・キーとそれに対応する行に必要なバイト数の見積もりです。Oracle では、次の処理を実行するときに、SIZE パラメータが使われます。

- クラスタ化したデータ・ブロック内に収めることのできるクラスタ・キー（および対応する行）の数を見積もる。
- クラスタ化したデータ・ブロック内に置かれるクラスタ・キーの数を制限する。これによって、クラスタ内のキーの記憶効率を最大にします。

SIZE はクラスタ・キーが使える領域を制限しません。たとえば、2 つのクラスタ・キーが 1 つのデータ・ブロックに収まるようにサイズが設定された場合、どちらのクラスタ・キーも、その利用可能なデータ・ブロック領域をいくらかでも（すべてであっても）使えます。

デフォルトでは、Oracle は 1 つのクラスタ・キーとそれに対応付けられている行をそのクラスタのデータ・セグメントの各データ・ブロックに格納します。ブロック・サイズはオペレーティング・システムによって違いますが、クラスタ化した表が他のマシン上の他のデータベースにインポートされる时候にも、ブロックあたり 1 つのキーというルールは守られます。

クラスタ・キー値に対するすべての行を、1 つのブロック内に収めることができない場合、ブロックはまとめて連鎖され、キーによるすべての値へのアクセス速度を向上させます。クラスタ索引は、各クラスタ・キー値に対応する行を内容とするブロックの連鎖の始点を示します。クラスタのサイズが、複数のキーが 1 つのブロックに収まる大きさの場合、ブロックが複数の連鎖に属する可能性があります。

各クラスタとクラスタ索引の位置の指定

適当な権限と表領域割当て制限を持っていると、現在、オンライン状態の表領域内に新しいクラスタを作成できます。新しいクラスタまたは索引を格納する表領域を識別するためには、CREATE CLUSTER/INDEX 文に TABLESPACE オプションを必ず指定してください。

クラスタとそのクラスタ索引は異なる表領域内に作成できます。実際、クラスタとその索引を、それぞれ異なる記憶デバイス上に格納された異なる表領域内に作ると、ディスク競合を最小限にして表データと索引データを同時に検索できます。

クラスタ・サイズの見積もりおよび記憶領域パラメータの設定

クラスタを作る前にクラスタのサイズを見積もる利点は、次のとおりです。

- 索引、ロールバック・セグメント、REDO ログ・ファイルの見積もりと合わせたクラスタの見積もりサイズを使って、作るデータベースを保持するために必要なディスク容量を決定できる。この見積もりを利用して適切なハードウェアを購入したりできます。
- 個々のクラスタの見積もりサイズを使うと、クラスタが使うディスク領域をよりよく管理できる。クラスタを作るときに、適切な記憶領域パラメータを設定し、そのクラスタを使うアプリケーションの I/O パフォーマンスを改善できます。

表を作る前に表サイズを見積もるかどうにかかわらず、クラスタ化されてない表を作るときは記憶領域パラメータを明示的に設定できます。表を作るとき、または表を変更するときに記憶領域パラメータを設定しないと、その表が常駐する表領域に設定されたデフォルト記憶領域パラメータが自動的に使われます。クラスタ化した表はクラスタの記憶領域パラメータも自動的に使います。

関連項目：スキーマ・オブジェクトのサイズの見積もりの詳細は、付録 A「スキーマ・オブジェクトの領域の見積もり」を参照してください。

クラスタの作成

ここでは、クラスタを作る方法について説明します。トピックは次のとおりです。

- クラスタ化された表を作成する
- クラスタ索引を作成する

自分のスキーマにクラスタを作るには、CREATE CLUSTER システム権限とそのクラスタを含む予定の表領域に対する割当て制限を持っているか、または UNLIMITED TABLESPACE システム権限を持っていないければなりません。

別のユーザーのスキーマにクラスタを作るには CREATE ANY CLUSTER システム権限を持っていないければならず、また、所有者はそのクラスタを含む予定の表領域に対する割当て制限を持っているか、または UNLIMITED TABLESPACE システム権限を持っていないければなりません。

SQL コマンド CREATE CLUSTER を使って、クラスタを作ります。次の文は、DEPTNO 列によってクラスタ化された、EMP 表と DEPT 表を格納するクラスタ EMP_DEPT を作ります。

```
CREATE CLUSTER emp_dept (deptno NUMBER(3))
    PCTUSED 80
    PCTFREE 5
    SIZE 600
    TABLESPACE users
    STORAGE (INITIAL 200k
        NEXT 300K
        MINEXTENTS 2
        MAXEXTENTS 20
        PCTINCREASE 33);
```

クラスタ化された表を作成する

クラスタに表を作るには、CREATE TABLE システム権限または CREATE ANY TABLE システム権限のどちらかを持っていなければなりません。なお、クラスタ内に表を作るには、表領域割当て制限または UNLIMITED TABLESPACE システム権限を必要としません。

CLUSTER オプションを指定した SQL コマンド CREATE TABLE を使って、クラスタに表を作成できます。たとえば、次の文を使って、EMP 表と DEPT 表を EMP_DEPT クラスタに作成できます。

```
CREATE TABLE dept (  
    deptno NUMBER(3) PRIMARY KEY, . . . )  
    CLUSTER emp_dept (deptno);
```

```
CREATE TABLE emp (  
    empno NUMBER(5) PRIMARY KEY,  
    ename VARCHAR2(15) NOT NULL,  
    . . .  
    deptno NUMBER(3) REFERENCES dept)  
    CLUSTER emp_dept (deptno);
```

注意 : CREATE TABLE 文で、クラスタ化した表に対してスキーマを指定できます。クラスタ化した表は、そのクラスタを含むスキーマとは別のスキーマに存在させることもできます。

クラスタ索引を作成する

クラスタ索引を作るには、次の条件の 1 つが真でなければなりません。

- 自分のスキーマ内にクラスタが含まれ、さらに CREATE INDEX システム権限を持っている。
- CREATE ANY INDEX システム権限を持っている。

どちらの場合も、クラスタ索引を収録する表領域に対する割当て制限、または UNLIMITED TABLESPACE システム権限を持っていなければなりません。

クラスタ化した表に行を挿入する前に、クラスタ索引を作らなければなりません。次の文は、EMP_DEPT クラスタに対するクラスタ索引を作ります。

```
CREATE INDEX emp_dept_index  
    ON CLUSTER emp_dept  
    INITTRANS 2  
    MAXTRANS 5  
    TABLESPACE users  
    STORAGE (INITIAL 50K  
        NEXT 50K  
        MINEXTENTS 2  
        MAXEXTENTS 10)
```

```
PCTINCREASE 33)
PCTFREE 5;
```

クラスタ・キーにより、クラスタ内の表の関連が確立されます。いくつかの記憶設定が、クラスタとクラスタ索引に対して明示的に指定されています。

関連項目：システム権限の詳細は、第 21 章「ユーザー権限とロールの管理」を、表領域割当て制限の詳細は、第 20 章「ユーザーとリソースの管理」を参照してください。

クラスタの変更

既存のクラスタを変更して、次の設定を変更できます。

- データ・ブロック領域使用パラメータ (PCTFREE、PCTUSED)
- 平均のクラスタ・キー・サイズ (SIZE)
- トランザクション・エントリの設定 (INITRANS、MAXTRANS)
- 記憶領域パラメータ (NEXT、PCTINCREASE)

クラスタを変更するには、そのクラスタが自分のスキーマに含まれているか、または ALTER ANY CLUSTER システム権限を持っていないなりません。

クラスタのデータ・ブロック領域使用パラメータ (PCTFREE と PCTUSED) またはクラスタ・サイズ・パラメータ (SIZE) を変更するときには、そのクラスタにすでに割り当てられているブロックと今後割り当てられるブロックを含め、そのクラスタが使うすべてのデータ・ブロックに対して新しい設定が適用されます。すでに表に割り当てられているブロックは、必要なときに (ただちにではなく) 再編成されます。

クラスタのトランザクション・エントリ設定 (INITRANS、MAXTRANS) を変更するときには、MAXTRANS の新しい設定がクラスタのすべてのデータ・ブロック (すでに割り当てられたブロックとその後割り当てられるブロック) に適用される一方、INITRAN の新しい設定はその後クラスタに割り当てられるブロックだけに適用されます。

記憶領域パラメータ INITIAL と MINEXTENTS は変更できません。他の記憶領域パラメータの新しい設定はすべて、その後クラスタに割り当てられるエクステントにしか影響しません。

クラスタを変更するには、SQL コマンド ALTER CLUSTER を使います。次の文は EMP_DEPT クラスタを変更します。

```
ALTER CLUSTER emp_dept
PCTFREE 30
PCTUSED 60;
```

クラスタ化した表とクラスタ索引を変更する

SQL コマンド ALTER TABLE を使って、クラスタ化した表を変更できます。ただし、クラスタ化した表に対して ALTER TABLE 文でどのようなデータ・ブロック領域パラメータ、またはトランザクション・エントリ・パラメータ、記憶領域パラメータを設定しても、エラー・メッセージ (ORA-01771 「クラスタ表に対するオプションが無効です。」) が出されます。これは、Oracle はそのクラスタのパラメータをすべてのクラスタ化した表に対して使うからです。そのため、ALTER TABLE コマンドは、クラスタ化した表に対して、列の追加や修正、または整合性制約やトリガーの追加、削除、使用可能または使用禁止にするためにだけ使えます。

注意： クラスタ索引のサイズを見積もるときには、索引は実際の行ではなく各クラスタ・キーに付いていることに注意してください。各キーは索引内に 1 度しか現れません。

クラスタに記憶領域を手動で割り当てる

Oracle は、必要に応じてクラスタのデータ・セグメントに追加のエクステントを動的に割り当てます。ただし、状況によっては、クラスタに追加のエクステントを明示的に割り当てることもできます。たとえば、Oracle Parallel Server を使っているとき、クラスタのエクステントを特定のインスタンスに明示的に割り当てることができます。

新しいエクステントは、ALLOCATE EXTENT オプションを指定した SQL コマンド ALTER CLUSTER を使って、クラスタに割り当てることができます。

関連項目：表の変更の詳細は、12-7 ページの「表の変更」を参照してください。

クラスタ索引は、他の索引と同じように変更します。詳細は、14-8 ページの「索引の変更」を参照してください。

ALTER CLUSTER コマンドの CLUSTER パラメータの詳細は、『Oracle8 Parallel Server 概要および管理』を参照してください。

クラスタの削除

ここでは、クラスタの削除について説明します。トピックは次のとおりです。

- クラスタ化した表を削除する
- クラスタ索引を削除する

クラスタ内の表が不要になった場合、そのクラスタを削除できます。クラスタを削除すると、そのクラスタ内にある表と対応するクラスタ索引も削除されます。クラスタのデータ・セグメントとクラスタ索引の索引セグメントの両方に属するすべてのエクステントは、それらを含んでいる表領域に戻され、その表領域内の他のセグメントで利用できるようになります。

クラスタ化した表を削除する

クラスタを削除するには、そのクラスタが自分のスキーマに含まれているか、または DROP ANY CLUSTER システム権限を持っていなければなりません。クラスタ化した表をそのクラスタの所有者が所有していなくても、その表を含むクラスタを削除するために特別の権限は必要ありません。

クラスタ化した表は、その表のクラスタ、または他のクラスタ化した表、クラスタ索引に影響を及ぼすことなく、個別に削除できます。クラスタ化した表は、クラスタ化されてない表を削除する場合と同じように、SQL コマンドの DROP TABLE を使って削除します。

注意： 単一の表をクラスタから削除するとき、Oracle は表の各行を個々に削除します。クラスタ全体を削除するときに効率を最大にするには、INCLUDING TABLES オプション付きの DROP CLUSTER コマンドを使って、すべての表も含めて、そのクラスタを削除してください。クラスタの残りの部分をそのままにしておくときだけ、(DROP TABLE コマンドを使って) クラスタから個々の表を削除してください。

関連項目： 表の削除の詳細は、12-8 ページの「表の削除」を参照してください。

クラスタ索引を削除する

クラスタ索引は、クラスタまたはそのクラスタ化した表に影響を及ぼさずに削除できます。ただし、クラスタ索引が存在しないと、クラスタ化した表を使えません。クラスタへのアクセスを可能にするには、クラスタ索引を作りなおさなければなりません。断片化したクラスタ索引を再構築する手順の一部として、クラスタ索引が削除されることがあります。

表を含まないクラスタとそのクラスタ索引を削除するには、SQL コマンドの DROP CLUSTER を使います。たとえば、次の文は空のクラスタ EMP_DEPT を削除します。

```
DROP CLUSTER emp_dept;
```

クラスタに 1 つ以上のクラスタ化された表が含まれているときにその表も削除する場合は、DROP CLUSTER コマンドの INCLUDING TABLES オプションを次のように追加してください。

```
DROP CLUSTER emp_dept INCLUDING TABLES;
```

INCLUDING TABLES オプションを指定していないのに、クラスタに表が含まれているとエラーが戻されます。

クラスタ内の 1 つ以上の表が、そのクラスタ外に存在する表の FOREIGN KEY 制約によって参照される主キーまたは一意キーを含んでいる場合、その依存関係を持つ FOREIGN KEY 制約が削除されない限り、そのクラスタを削除できません。この削除は、DROP CLUSTER コマンドの CASCADE CONSTRAINTS オプションを使うと簡単に実行できます。次に例を示します。

```
DROP CLUSTER emp_dept INCLUDING TABLES CASCADE CONSTRAINTS;
```

制約が存在しているのに、CASCADE CONSTRAINTS オプションを使わないと、Oracle はエラーを戻します。

関連項目 : 索引の削除の詳細は、14-9 ページの「索引の削除」を参照してください。

ハッシュ・クラスタの管理

この章では、ハッシュ・クラスタを管理する方法について説明します。トピックは次のとおりです。

- ハッシュ・クラスタを管理するためのガイドライン
- ハッシュ・クラスタの変更
- ハッシュ・クラスタの削除

関連項目：この章で説明する作業を実行する前に、第 10 章「スキーマ・オブジェクトを管理するためのガイドライン」の内容をよく理解しておいてください。

ハッシュ・クラスタを管理するためのガイドライン

ここでは、ハッシュ・クラスタの管理を実行する前に考慮すべきガイドラインについて説明します。トピックは次のとおりです。

- ハッシングの長所
- ハッシングの短所
- ハッシュ・クラスタが必要とするサイズを見積もり、記憶領域パラメータを設定する

ハッシュ・クラスタに表を格納することは、データ検索のパフォーマンスを改善するための1つの選択肢です。ハッシュ・クラスタは、索引または索引クラスタを持つクラスタ化しない表に対して選択肢を提供します。索引付きの表または索引クラスタでは、別個の索引に格納されるキー値を使って、表内の行の位置を決定します。ハッシングを使うには、ハッシュ・クラスタを作り、そこに表をロードします。表の行は物理的にはハッシュ・クラスタに格納され、ハッシュ関数の結果に従って検索します。

Oracle はハッシュ関数を使って、特定のクラスタ・キー値に基づく、ハッシュ値と呼ばれる数値の分布を生成します。ハッシュ・クラスタのキーは、索引クラスタのキーと同じように単一列でも複合キー（複数列キー）でもかまいません。ハッシュ・クラスタ内の行の検索または格納を行う場合、Oracle は行のクラスタ・キー値にハッシュ関数を適用します。結果として生成されるハッシュ値はクラスタ内のデータ・ブロックに対応しており、以後、Oracle は、発行された文のかわりにその読み込みや書き込みをします。

索引付きの表またはクラスタ内の行の検索または格納のためには、少なくとも2回（通常はそれ以上）のI/Oが実行されなければなりません。

- 1回以上のI/Oによる、索引内のキー値の検索または格納の実行
- 別のI/Oによる、表またはクラスタ内の行の読み取りまたは書き込みの実行

一方、Oracle はハッシュ関数を使ってハッシュ・クラスタ内の行の位置を突き止めるので、I/Oは必要ありません。結果として、ハッシュ・クラスタ内の行の読み込みや書き込みに必要とされるのは最小限のI/O操作だけになります。

ハッシングの長所

ハッシングではなく索引を使う場合は、表を個別に格納するのか、またはクラスタの一部として格納するのかを考慮してください。

ハッシングは、次のような状況で最も有効です。

- ほとんどの問合せが次のようなクラスタ・キーとの等式を含んでいる場合。

```
SELECT . . . WHERE cluster_key = . . . ;
```

このような場合、等価条件を満たすクラスタ・キーがハッシュされ、対応するハッシュ・キーは通常一度の読み込みで見つかります。索引付きの表では、キー値を索引内で見つけなければならず（通常複数回の読み込み）その後で行が表から読み込まれます（さら別の読み込み）。

- ハッシュ・クラスタ内の表のサイズが初めに固定されていて、行数とそのクラスタ内の表が必要とする領域を決定できる場合。ハッシュ・クラスタ内の表でそのクラスタの初期割当てより多くの領域が必要な場合、オーバーフロー・ブロックが必要となるためにパフォーマンスがかなり低下する可能性があります。

ハッシングの短所

ハッシングは次のような状況では有効ではありません。

- ほとんどの問合せがクラスタ・キー値全体にわたって行を検索する場合。たとえば、全表走査、または次のような問合せでは、ハッシュ関数は特定のハッシュ・キーの位置を決定するためには使えません。これにかわって、全表走査と同等の機能を実行して問合せの行を取り出す必要があります。

```
SELECT . . . WHERE cluster_key < . . . ;
```

索引では、キー値はその索引内で順序付けられており、問合せの WHERE 句を満たすクラスタ・キー値を、比較的少ない I/O で見つけることができます。

- 表が絶えず大きくなる場合。表が無制限に大きくなる場合、表（そのクラスタ）の存続期間にわたって必要な領域を事前に定義できません。
- アプリケーションが表に対して頻繁に全表走査を実行し、表が散在している場合。この状況でハッシングを使うと、処理に必要な時間が長くなります。
- 最終的にハッシュ・クラスタが必要とする領域を事前に割り当てることができない場合。

関連項目：ハッシュ・クラスタの作成とハッシュ関数の指定の詳細は、『Oracle8 Server SQL リファレンス』を参照してください。

ハッシュ関数とユーザー定義のハッシュ関数の指定の詳細は、『Oracle8 Server 概要』を参照してください。

ハッシングを使う場合でも、クラスタ・キーを含む表のどの列にも異なる索引が付いている可能性があります。その他の推奨事項の詳細は、『Oracle8 Server アプリケーション開発者ガイド』を参照してください。

ハッシュ・クラスタが必要とするサイズを見積もり、記憶領域パラメータを設定する

索引クラスタの場合と同じように、ハッシュ・クラスタ内のデータに必要な記憶領域を見積もることは重要です。

Oracle は、領域の初期割当てが設定 SIZE と HASHKEYS によってハッシュ表を格納するのに十分であることを保証します。記憶領域パラメータ INITIAL、NEXT、MINEXTENTS の設定がハッシュ表サイズを計算していない場合、少なくとも SIZE*HASHKEYS に達するまで増分（追加の）エクステンツが割り当てられます。たとえば、データ・ブロック・サイズが 2KB、利用可能なデータ領域がブロックあたりおおよそ 1900 バイト（データ・ブロック・

サイズからオーバーヘッドを引く)と想定し、CREATE CLUSTER コマンドに STORAGE パラメータと HASH パラメータを指定すると次のようになります。

```
STORAGE (INITIAL 100K
        NEXT 150K
        MINEXTENTS 1
        PCTINCREASE 0)
SIZE 1500
HASHKEYS 100
```

この例では、データ・ブロックあたり、ハッシュ・キーを1つだけ割り当てることができます。そのため、ハッシュ・クラスタが必要とする初期領域は少なくとも 200KB(100*2KB) です。記憶領域パラメータの設定はこの要件を考慮していません。そのため、100KB の初期のエクステントと 150KB の増分のエクステントがハッシュ・クラスタに割り当てられます。

一方、HASH パラメータが次のように指定されるとします。

```
SIZE 500 HASHKEYS 100
```

この場合、データ・ブロックあたり、3つのハッシュ・キーが割り当てられます。そのため、ハッシュ・クラスタが必要とする初期領域は少なくとも 68KB(34*2KB) です。記憶領域パラメータの初期設定は、この要件を満たしています(100KBの初期エクステントがハッシュ・クラスタに割り当てられます)。

関連項目: ハッシュ・クラスタのサイズを見積もるには、A-10 ページの「クラスタに必要な領域の見積もり」で説明されている手順に従ってください。

ハッシュ・クラスタを作成する

ハッシュ・クラスタが作られると、表をそのクラスタ内に作成できます。ハッシュ・クラスタは、SQL コマンド CREATE CLUSTER を使って作成できます。たとえば、次の文は TRIAL 表を格納するクラスタ TRIAL_CLUSTER を作り、TRIALNO 列によってクラスタ化します。

```
CREATE CLUSTER trial_cluster (trialno NUMBER(5,0))
    PCTUSED 80
    PCTFREE 5
    TABLESPACE users
    STORAGE (INITIAL 250K      NEXT 50K
             MINEXTENTS 1      MAXEXTENTS 3
             PCTINCREASE 0)
    HASH IS trialno HASHKEYS 150;

CREATE TABLE trial (
    trialno          NUMBER(5,0) PRIMARY KEY,
    ...)
    CLUSTER trial_cluster (trialno);
```

次の部分では、ハッシュ・クラスタに固有の CREATE CLUSTER コマンドのパラメータの設定について説明します。

関連項目：クラスタ内に表を作る方法、CREATE CLUSTER コマンドの他のパラメータの設定のガイドライン、ハッシュ・クラスタを作るために必要な権限の詳細は、15-6 ページの「クラスタの作成」を参照してください。

ハッシュ・クラスタ内の領域使用を制御する

ハッシュ・クラスタを作るときは、パフォーマンスと使用領域が最適になるようにクラスタ・キーを正しく選択し、HASH IS、SIZE、HASHKEYS の各パラメータを設定することが重要です。次に示すガイドラインでは、これらのパラメータを設定する方法について説明します。

キーを選択する

正しいクラスタ・キーの選択は、クラスタ化した表に対して最もよく発行される問合せのタイプに依存しています。たとえば、ハッシュ・クラスタ内の EMP 表について検討します。問合せが従業員番号によって行を頻繁に選択する場合、EMPNO 列をクラスタ・キーにするとういでしょう。問合せが部門番号によって行を頻繁に選択する場合には、DEPTNO 列をクラスタ・キーにすべきです。単一の表を含むハッシュ・クラスタの場合、通常クラスタ・キーをその含まれる表の主キー全体にします。

ハッシュ・クラスタのキーは、索引クラスタのキーと同じように単一列でも複合キー（複数列キー）でもかまいません。複合キーによるハッシュ・クラスタは、Oracle の内部ハッシュ関数を使わなければなりません。

HASH IS パラメータを設定する

HASH IS パラメータを指定するのは、クラスタ・キーが NUMBER データ型の単一の列であり、その内容が、均一に分布した整数である場合だけにしてください。上の条件が当てはまる場合、それぞれの一意のクラスタ・キー値が衝突なしで一意ハッシュ値にハッシュするように、クラスタ内に行を分布させることができます。上の条件が当てはまらない場合は、このオプションを指定しないで内部ハッシュ関数を使ってください。

SIZE を設定する

SIZE は、ハッシュ・キーに対してすべての行を保持するために必要な領域の平均サイズに設定してください。そのため、SIZE を適切に決定するには、自分のデータの特性をよく理解していなければなりません。

- ハッシュ・クラスタが表を 1 つだけ収録し、その表の行のハッシュ・キー値が一意（値あたり 1 行）であれば、SIZE はクラスタ内の平均の行サイズに設定できる。
- ハッシュ・クラスタが複数の表を含むのであれば、SIZE は代表ハッシュ値に対応付けられているすべての行を保持するために必要領域の平均サイズに設定できる。

関連項目：SIZE の事前の値を見積もるには、付録 A「スキーマ・オブジェクトの領域の見積もり」の手順に従ってください。SIZE の事前の値が小さい場合（データ・ブロックあたり 4 より多くのハッシュ・キーを割り当てることができます）、CREATE CLUSTER コマンドの SIZE にこの値を使えます。

しかし、SIZE の値が大きいく（たとえば、データ・ブロックあたり 5 未満のハッシュ・キーしか割り当てることができない）場合、次のように、予想される衝突の頻度、および自分にとって重要なのはデータ検索のパフォーマンスか、または領域使用の効率かについても検討すべきです。

- ハッシュ・クラスタが、内部ハッシュ関数を使わず（HASH IS を指定した場合）、衝突がわずかであるか、または衝突のないことが予想される場合、見積もりのとおりに SIZE を設定できる。この場合、衝突は発生せず、領域は可能な限り効果的に使われます。
- 挿入に関して頻繁な衝突が予想される場合、行を格納するためにオーバーフロー・ブロックが割り当てられる可能性は高くなる。衝突が頻繁な場合にブロックのオーバーフローの可能性を軽減し、パフォーマンスを最大にするには、表 16-1 に従って SIZE を大きくする必要があります。

表 16-1 SIZE の増加チャート

ブロックあたりの使用可能領域 / 計算された SIZE	SIZE の設定
1	算出された SIZE
2	算出された SIZE + 15%
3	算出された SIZE + 12%
4	算出された SIZE + 8%
>4	算出された SIZE

ただし、SIZE の値を過大に見積もると、クラスタ内の未使用の領域を増やすことになっていきます。領域効率がデータ検索のパフォーマンスよりも重要であれば、上の調整を無視して SIZE に見積もりの値を使ってください。

HASHKEYS を設定する

ハッシュ・クラスタの行の配分が最大のときは、必ず HASHKEYS を素数にしてください。

たとえば、EMP 表を DEPTNO によってクラスタ化し、そして値 10、20、...1000 を持つ DEPTNO が 100 あるとします。内部ハッシュ関数を無視し、100 の HASHKEYS を持つクラスタを作ると想定すると、部門 10 は 10、部門 20 は 20、...、部門 110 は 10(110 を 100 で割った余り)、部門 120 は 20、のようにハッシュします。10、20、... のハッシュ値に対して 10 のエントリが存在しますが、1、2、... に対しては何も存在しないことに注意してください。その結果として、浪費されている領域が大量に存在し、衝突のために多数のオーバーフロー・ブロックが存在する可能性があります。一方、HASHKEYS が 101 に設定されると、各部門番号は一意的ハッシュ・キー値をハッシュします。

ハッシュ・クラスタ内の使用領域を制御する：例

次に示す例では、クラスタ・キーを正しく選択し、HASH IS、SIZE、HASHKEYS の各パラメータを設定する方法を示します。すべての例について、データ・ブロック・サイズは 2KB であり、平均して各ブロックの 1950 バイトが利用可能なデータ領域です（ブロック・サイズからオーバーヘッドを引く）。

例 1 ハッシュ・クラスタに EMP 表をロードします。ほとんどの問合せは、従業員番号によって従業員レコードを検索します。常に EMP 表の最大行数は 10000 であり、平均の行サイズが 55 バイトとして見積もられています。

この場合、EMPNO をクラスタ・キーにすべきです。この列は一樣に分布する整数を持っていますので、内部ハッシュ関数は無視できます。SIZE は平均の行サイズ (55 バイト) に設定できます。データ・ブロックあたり、34 のハッシュ・キーが割り当てられることに注目してください。HASHKEYS は、表の行数 (10000) より大きい最初の素数 (10001) に切り上げたものに設定できます。

```
CREATE CLUSTER emp_cluster (empno  
NUMBER)  
...  
SIZE 55  
HASH IS empno HASHKEYS 10001;
```

例 2 先の例と同様の条件を考えます。ただし、この場合、行は通常部門番号によって検索されます。多くて、部門あたり平均 10 名の従業員を持つ部門が 1000 存在します。部門番号が 10 ずつ増加する (0、10、20、30、...) ことに注目してください。

この場合、EMPNO をクラスタ・キーにすべきです。この列は一樣に分布する整数を持っていますので、内部ハッシュ関数は無視できます。事前に見積もった SIZE（部門あたりのすべての行を保持するために必要な領域の平均サイズ）は 55×10 バイト、つまり 550 バイトです。SIZE にこの値を使って、データ・ブロックあたり、3 つのハッシュ・キーだけを割り当てることができます。いくつかの衝突を予想し、データ検索の最大パフォーマンスが必要であれば、オーバーフロー・ブロックを必要とすることから衝突を防ぐために、見積もりの SIZE を少し変更してください。SIZE を 12% だけ調整して 620 バイトにすることによって (SIZE の設定に関する先の節を参照してください)、データ・ブロックあたり 3 つのハッシュ・キーだけが割り当てられ、予想される衝突の行のためにより多くの領域を残します。

HASHKEYS は、一意の部門番号の数 (1000) より大きい最初の素数 (1009) に切り上げたものに設定できます。

```
CREATE CLUSTER emp_cluster (deptno NUMBER)
. . .
SIZE 620
HASH IS deptno HASHKEYS 1009;
```

ハッシュ・クラスタの変更

ハッシュ・クラスタは、SQL コマンド ALTER CLUSTER を使って変更します。

```
ALTER CLUSTER emp_dept . . . ;
```

ハッシュ・クラスタの変更に関係する問題は、索引クラスタを変更する場合と同じです。ただし、SIZE、HASHKEYS、HASH IS の各パラメータは ALTER CLUSTER 文に指定できません。クラスタを作りなおしてこれらのパラメータを変更し、元のクラスタからデータをコピーしなければなりません。

関連項目：索引クラスタの変更の詳細は、15-8 ページの「クラスタの変更」を参照してください。

ハッシュ・クラスタの削除

ハッシュ・クラスタは、SQL コマンド DROP CLUSTER を使って削除します。

```
DROP CLUSTER emp_dept;
```

ハッシュ・クラスタ内の表が、SQL コマンド DROP TABLE を使って削除されます。ハッシュ・クラスタとハッシュ・クラスタ内の表の削除に關係する問題は、索引クラスタの場合と同じです。

関連項目：クラスタの削除の詳細は、15-9 ページの「クラスタの削除」を参照してください。

スキーマ・オブジェクトの一般的な管理

この章では、第 10 章～第 15 章までで扱わなかった、スキーマ・オブジェクトの一般的な管理について説明します。トピックは次のとおりです。

- 一度の操作で複数の表やビューを作成
- オブジェクトの改名
- 表、索引、クラスタの分析
- 表とクラスタの削除
- トリガーの使用可能、使用禁止
- 整合性制約の管理
- オブジェクトの依存性の管理
- オブジェクト名の名前の変換の管理
- データ・ディクショナリの記憶領域パラメータの変更
- スキーマ・オブジェクトについての情報を表示

一度の操作で複数の表やビューを作成

スキーマ・オブジェクトを作成するには、それに伴う操作に必要な権限を持っていなければなりません。たとえば、CREATE SCHEMA コマンドを使用して複数の表を作成するには、表を作成するために必要な権限を持っていなければなりません。

SQL コマンド CREATE SCHEMA を使用して、一度の操作で複数の表やビューを作成し、権限を付与できます。確実に一度の操作で複数の表やビューを作成して権限を付与するには、CREATE SCHEMA コマンドが便利です。個々の表やビューの作成が失敗したり、権限付与が失敗したりすると、文全体がロールバックされます。オブジェクトは作成されず、権限も付与されません。次の文は、2 つの表とそれらのデータを結合するビューを作成します。

```
CREATE SCHEMA AUTHORIZATION scott
CREATE TABLE dept (
    deptno NUMBER(3,0) PRIMARY KEY,
    dname VARCHAR2(15),
    loc VARCHAR2(25)
CREATE TABLE emp (
    empno NUMBER(5,0) PRIMARY KEY,
    ename VARCHAR2(15) NOT NULL,
    job VARCHAR2(10),
    mgr NUMBER(5,0),
    hiredate DATE DEFAULT (sysdate),
    sal NUMBER(7,2),
    comm NUMBER(7,2),
    deptno NUMBER(3,0) NOT NULL
CONSTRAINT dept_fkey REFERENCES dept)
CREATE VIEW sales_staff AS
    SELECT empno, ename, sal, comm
    FROM emp
    WHERE deptno = 30
    WITH CHECK OPTION CONSTRAINT sales_staff_cnst
GRANT SELECT ON sales_staff TO human_resources;
```

CREATE SCHEMA コマンドは、ANSI の CREATE TABLE コマンドと CREATE VIEW コマンドを拡張した Oracle 独自の機能をサポートしていません。これには、STORAGE 句が含まれます。

オブジェクトの改名

オブジェクトを改名するには、そのオブジェクトを所有していなければなりません。スキーマ・オブジェクトは、次のどちらかの方法で改名できます。

- オブジェクトを削除して作成しなおす。
- SQL コマンド RENAME を使用して改名する。

オブジェクトを削除して作成しなおす場合、そのオブジェクトの権限付与はすべて失われます。オブジェクトを作成しなおすときに、権限を再付与しなければなりません。一方、

RENAME コマンドを使用して、表、ビュー、順序、プライベート・シノニムを改名できます。RENAME コマンドを使用すると、そのオブジェクトの権限付与は新しい名前に引き継がれます。たとえば、次の文は SALES_STAFF ビューを改名します。

```
RENAME sales_staff TO dept_30;
```

注意： ストアド PL/SQL プログラム・ユニット、パブリック・シノニム、索引、クラスタは改名できません。そのようなオブジェクトを改名するには、削除してから作成しなおします。

スキーマ・オブジェクトを改名する前に、次のような影響を理解する必要があります。

- 改名されたオブジェクトに依存しているすべてのビューと PL/SQL プログラム・ユニットは、無効になる次の使用の前に再コンパイルしなければならない。
- 改名されたオブジェクトのシノニムは使われるとエラーを戻す。

関連項目： Oracle でオブジェクトの依存性を管理する方法に関する詳細は、17-22 ページの「オブジェクトの依存性の管理」を参照してください。

表、索引、クラスタの分析

ここでは、表、索引、クラスタを分析する方法について説明します。トピックは次のとおりです。

- 表、索引、クラスタの統計を使用する
- 表、索引、クラスタの妥当性を検査する
- 表とクラスタの連鎖された行をリストする

表、索引、クラスタを分析して、それについてのデータを収集したり、その記憶形式の妥当性を検証したりできます。表、クラスタ、索引を分析するには、その表、クラスタ、索引を所有しているか、または ANALYZE ANY システム権限を持っていなければなりません。

これらのスキーマ・オブジェクトを分析して、特定のオブジェクトについての統計の収集や更新もできます。DML 文が発行されると、参照されるオブジェクトの統計を使用して、その文の最も効率的な実行計画が決定されます。この最適化を「コストベースの最適化」と呼びます。統計はデータ・ディクショナリ内に格納されます。

表、索引、クラスタを分析して、オブジェクトの構造の妥当性を検査できます。たとえば、ハードウェアやその他のシステムに障害が発生したような場合、索引が破壊され、正しく機能しなくなる可能性があります。索引の妥当性検査をするとき、索引の中のすべてのエントリが対応付けられた表の正しい行を示していることを確かめることができます。スキーマ・オブジェクトが破壊された場合、そのオブジェクトを削除して再作成できます。

表やクラスタの連鎖された行に関する情報を収集するために、その表やクラスタを分析できます。この結果は、行の更新のための領域が十分であるかどうかを判断する上で役に立ちます。たとえば、この情報によって、PCTFREE が表やクラスタに対して適切に設定されているかどうかを知ることができます。

関連項目：パフォーマンスの統計表示とオプティマイザの表、索引、クラスタの詳細は、『Oracle8 Server チューニング』を参照してください。

表、索引、クラスタの統計を使用する

STATISTICS オプションを指定した SQL コマンド統計表示 ANALYZE を使用して、表、索引、クラスタの物理記憶特性に関する統計を収集し、データ・ディクショナリに格納できます。コストベースの最適化によって、分析したオブジェクトにアクセスする SQL 文の最も効率的な実行計画を選択するときに、Oracle はこのような統計を使用できます。また、このコマンドによって生成される統計を使用して、分析されたオブジェクトにアクセスする効率的な SQL 文を書くこともできます。

COMPUTE STATISTICS オプションまたは ESTIMATE STATISTICS オプションを指定した ANALYZE コマンドを使用して、統計の計算や見積もりができます。

COMPUTE STATISTICS	統計を計算するとき、オブジェクト全体が、オブジェクトに関するデータを収集するために走査されます。Oracle では、オブジェクトに関する厳密な統計値を計算するために、このデータを使用します。計算された統計では、オブジェクトにわたってわずかな変動がみられます。計算された統計の情報を収集するために、オブジェクト全体が走査されるので、オブジェクトのサイズが大きいほど、必要な情報を収集するために必要な作業が多くなります。
ESTIMATE STATISTICS	統計を見積もるとき、Oracle はオブジェクトの一部から代表的な情報を収集します。この情報のサブセットにより、オブジェクトに関して合理的に統計を見積もることができます。統計を見積もる際の正確さは、Oracle がサンプリングの代表値をどのように使用するかに依存します。見積もる統計の情報の収集ではオブジェクトの一部だけが走査されるので、オブジェクトを高速に分析できます。Oracle が見積もる上で使用するべき行の数や割合は、オプションで指定できます。

注意： 表やクラスタの統計を計算するとき、その計算に必要な一時領域の容量は指定された行の数に関連します。COMPUTE STATISTICS の場合、表全体を格納し、ソートするために十分な一時領域に、行ごとの多少のオーバーヘッドを加えた領域が必要となります。また、ESTIMATE STATISTICS の場合は、要求されたサンプル行を格納し、ソートするために十分な一時領域と行ごとの多少のオーバーヘッドを加えた領域が必要となります。索引の場合は、分析用に一時領域は必要ありません。

関連項目： SQL コマンド ANALYZE の詳細は、『Oracle8 Server SQL リファレンス』を参照してください。

統計を含むデータ・ディクショナリ・ビューの詳細は、『Oracle8 Server リファレンス・マニュアル』を参照してください。

オブジェクトの統計を表示する

オブジェクトの統計が計算されたもの、または見積もられたものでも、統計はデータ・ディクショナリ内に格納されます。次のデータ・ディクショナリ・ビューを使用して、統計を問い合わせることができます。

- USER_INDEXES、ALL_INDEXES、DBA_INDEXES
- USER_TABLES、ALL_TABLES、DBA_TABLES
- USER_TAB_COLUMNS、ALL_TAB_COLUMNS、DBA_TAB_COLUMNS

注意： これらのビューの中の行では、統計が収集された索引、表、クラスタの統計表示列にエントリが入ります。オブジェクトを ANALYZE するたびにそのオブジェクトに対するエントリが更新されます。

表の統計 表に関して、次に示す統計を収集できます。

注意： * 記号は、統計の計算時に数値が必ず正確な値であることを示しています。

- 行数
- 使われたブロック数 *
- 使われていないブロック数
- 利用可能な平均空き領域

- 連鎖行の数
- 平均行長
- 列あたりの重複しない値の数
- 列あたりの 2 番目に小さい値 *
- 列あたりの 2 番目に大きい値 *

注意： 表が分析されるときに、表に対応付けられたすべての索引の統計が自動的に収集されます。

索引の統計 索引に関して、次に示す統計を収集できます。

- 索引レベル *
- リーフ・ブロックの数
- 重複しないキーの数
- 平均のリーフ・ブロック / キーの数
- 平均のデータ・ブロック / キーの数
- クラスタ化の係数
- 最小のキー値 *
- 最大のキー値 *

クラスタの統計 クラスタに関して収集可能な唯一の統計は平均のクラスタ・キー連鎖長であり、この統計は見積もりまたは計算ができます。クラスタ内の表とクラスタ化した表に対応付けた（クラスタ・キー索引を含む）すべての索引に対する統計は、クラスタが統計に対して分析されるときに自動的に収集されます。

注意： ANALYZE 文が発行されるとき、指定されたオブジェクトに対する統計がデータ・ディクショナリに含まれていると、新しい統計がデータ・ディクショナリ内の古い統計に置き換わります。

統計を計算する

次の文は、EMP 表の統計を計算します。

```
ANALYZE TABLE emp COMPUTE STATISTICS;
```

次の問合せは、1064 行のデフォルト統計サンプルを使用して、EMP 行に関する統計を見積もります。

```
ANALYZE TABLE emp ESTIMATE STATISTICS;
```

Oracle が使用する統計サンプルを指定するには、ESTIMATE STATISTICS オプションとともに SAMPLE オプションを指定します。行や索引値の数、または表における行や索引値の百分率を示す整数を指定できます。次に、各オプションの指定例を示します。

```
ANALYZE TABLE emp
  ESTIMATE STATISTICS
    SAMPLE 2000 ROWS;
ANALYZE TABLE emp
  ESTIMATE STATISTICS
    SAMPLE 33 PERCENT;
```

どちらの場合も、50 より大きな百分率や、そのオブジェクトの中の行または索引値の数の 50% を超える行または索引値の数を指定すると、Oracle は見積もりではなく、正確な統計を計算します。

スキーマ・オブジェクトの統計を削除する

表、索引、クラスタの統計は、DELETE STATISTICS オプションを指定した ANALYZE コマンドを使用して、データ・ディクショナリから削除できます。あるオブジェクトに関する文に対して、コストベースの最適化を使わない場合、そのオブジェクトの統計も削除できません。次の文は、EMP 表の統計をデータ・ディクショナリから削除します。

```
ANALYZE TABLE emp DELETE STATISTICS;
```

共有 SQL と統計分析

表、クラスタ、索引を分析すると、(共有プール内にある) 現行の共有 SQL 文に影響が及びます。統計を更新、削除するためにオブジェクトが分析されるときはいつでも、文の次の実行が新しい統計を利用できるように、分析されたオブジェクトを参照するすべての共有 SQL 文はメモリーからフラッシュされます。

次のプロシージャをコールできます。

DBMS_UTILITY.
ANALYZE_SCHEMA()

このプロシージャには 2 つの引数、つまりスキーマの名前と分析方法 ('COMPUTE' または 'ESTIMATE'、'DELETE') が必要です。そのスキーマ内の全オブジェクトに関する統計が収集されます。

DBMS_DDL.
ANALYZE_OBJECTS()

このプロシージャには、4 つの引数、つまりオブジェクトのタイプ ('CLUSTER' または 'TABLE'、'INDEX')、およびオブジェクトのスキーマ、オブジェクトの名前、分析方法 ('COMPUTE' または 'ESTIMATE'、'DELETE') が必要です。そのオブジェクトに関する統計が収集されます。

これらのプロシージャを定期的にコールして、統計を更新するとよいでしょう。

表、索引、クラスタの妥当性を検査する

表、索引、クラスタ、スナップショットの構造の整合性を確認するためには、VALIDATE STRUCTURE オプションを指定した ANALYZE オプションを使います。構造が妥当である場合、エラーは戻されません。しかし、構造が破壊されているとエラー・メッセージが戻されます。表、または索引、クラスタが破壊された場合には、それを削除して作成しなおしてください。スナップショットが破壊された場合、完全リフレッシュを実行し、それで問題が解決したことを確認してください。問題が解決していなければ、スナップショットを削除し、作成しなおします。

次の文は、EMP 表を分析します。

```
ANALYZE TABLE emp VALIDATE STRUCTURE;
```

CASCADE オプションを含めると、オブジェクト、および関連するすべてのオブジェクトの妥当性を検査できます。次の文は、EMP 表と対応付けられているすべての索引の妥当性を検査します。

```
ANALYZE TABLE emp VALIDATE STRUCTURE CASCADE;
```

表とクラスタの連鎖された行をリストする

表またはクラスタの連鎖された行と移行された行は、LIST CHAINED ROWS オプションを指定した ANALYZE コマンドを使用して検出できます。このコマンドの結果は、LIST CHAINED ROWS オプションによって戻される情報を受け取るものとして明示的に作成して指定した表に格納されます。

ANALYZE... LIST CHAINED ROWS 文によって戻されるデータを受け取るための適当な表を作成するには、Oracle とともに提供される UTLCHAIN.SQL スクリプトを使用してください。UTLCHAIN.SQL スクリプトは、このスクリプトを実行するユーザーのスキーマ内に表 CHAINED_ROWS を作成します。

CHAINED_ROWS 表が作成されたなら、ANALYZE コマンドを使用するときにその表を指定できます。たとえば、次の文は CHAINED_ROWS 表に EMP_DEPT クラスタ内の連鎖された行に関する情報を含む行を挿入します。

```
ANALYZE CLUSTER emp_dept LIST CHAINED ROWS INTO chained_rows;
```

関連項目：UTLCHAIN.SQL スクリプトの名前と位置は、オペレーティング・システムによって異なります。オペレーティング・システム固有の Oracle マニュアルを参照してください。

表とクラスタ内の連鎖行と移行行の数を減らす方法の詳細は、『Oracle8 Server チューニング』を参照してください。

表とクラスタの削除

表（またはクラスタ）は存在したままで、完全に空になるように、表のすべての行、またはクラスタ化した表のグループのすべての行を削除できます。たとえば、毎月のデータを含む表があり、各月の終わりにそのデータをアーカイブした後で、表を空にする（すべての行を削除する）必要があります。

表からすべての行を削除するには、3つのオプションがあります。

1. DELETE コマンドを使用する。

DELETE コマンドを使用して表の行を削除できます。たとえば、次の文は EMP 表からすべての行を削除します。

```
DELETE FROM emp;
```

2. DROP コマンドと CREATE コマンドを使用する。

表を削除してから表を作成しなおします。たとえば、次の文は EMP 表を削除した後で作成しなおします。

```
DROP TABLE emp;  
CREATE TABLE emp ( . . . );
```

3. TRUNCATE を使用する。

SQL コマンド TRUNCATE を使用して表のすべての行を削除できます。たとえば、次の文は EMP 表を TRUNCATE します。

```
TRUNCATE TABLE emp;
```

DELETE を使用する

DELETE コマンドを使用するときに、表またはクラスタに多数の行が存在していると、それらの行を削除する際にかなりのシステム・リソースが消費されます。たとえば、その表と対応付けられた索引の CPU 時間、REDO ログ領域、ロールバック・セグメント領域はリソースを必要とします。また、各行が削除されるときに、トリガーが起動される可能性があります。その結果空になる表またはクラスタに前もって割り当てられていた領域は、そのオブジェクトに対応付けられたままです。

DROP と CREATE を使用する

表やクラスタを削除してから作成しなおすと、対応付けられた索引、整合性制約、トリガーもすべて削除され、削除される表またはクラスタ化した表に依存するオブジェクトはすべて無効になります。また、削除される表またはクラスタ化した表に対する権限付与もすべて削除されます。

TRUNCATE を使用する

TRUNCATE コマンドは、表またはクラスタからすべての行を削除するための高速で効率的な方法を提供します。TRUNCATE 文はロールバック情報を生成しないで、すぐにコミットします。つまり、それは DDL 文であり、ロールバックできません。TRUNCATE 文は、TRUNCATE される表に対応付けられている構造（制約、トリガー）または許可に影響を及ぼしません。また、TRUNCATE 文は、TRUNCATE の後で、表に現在割り当てられている領域を、その表を含む表領域に戻すかどうかも指定します。

ユーザーの対応するスキーマ内の表またはクラスタを TRUNCATE できます。また、DROP ANY TABLE システム権限を持っているユーザーは、どのスキーマ内の表またはクラスタでも TRUNCATE できます。

親キーを含む表またはクラスタ化表を TRUNCATE する前に、別の表の中の参照しているすべての外部キーを使用禁止にしなければなりません。自己参照制約を使用禁止にする必要はありません。

TRUNCATE 文が表から行を削除する場合、表に対応付けたトリガーは起動されません。また、TRUNCATE 文は、監査が使用可能であっても、DELETE 文に対応するどのような監査情報も生成しません。そのかわりに、単一の監査レコードが、発行された TRUNCATE 文に対して生成されます。

ハッシュ・クラスタは TRUNCATE では削除できません。また、ハッシュ・クラスタまたは索引クラスタ内の表を個々に TRUNCATE できません。索引クラスタを TRUNCATE すると、そのクラスタ内のすべての表からすべての行が削除されます。個々のクラスタ化した表からすべての行を削除しなければならない場合には、DELETE コマンドを使用するか、または表を削除してから作成しなおしてください。

TRUNCATE コマンドの REUSE STORAGE オプション、または DROP STORAGE オプションは、TRUNCATE の後に、表（またはクラスタ）に現在割り当てられている領域を、その表を含む表領域に戻すかどうかを制御します。デフォルトのオプション DROP STORAGE は、結果表に割り当てられているエクステントの数を少なくして、MINEXTENTS の元の設定にします。解放されたエクステントはシステムに戻され、他のオブジェクトによって使用できます。

一方、REUSE STORAGE オプションは、表またはクラスタに対して現在割り当てられているすべての領域を、それらに割り当てたままにすることを指定します。たとえば、次の文は EMP_DEPT クラスタを TRUNCATE し、クラスタに対して前から割り当てられているすべてのエクステントを、今後の挿入と削除のためにそのまま残します。

```
TRUNCATE CLUSTER emp_dept REUSE STORAGE;
```

REUSE オプションまたは DROP STORAGE オプションは、対応付けられたどの索引にも適用されます。表またはクラスタが TRUNCATE されると、対応付けられた索引もすべて TRUNCATE されます。また、TRUNCATE された表またはクラスタ、対応付けられた索引に対する記憶領域パラメータは、TRUNCATE の結果として変化しません。

関連項目：監査の詳細は、第 22 章「データベース使用の監査」を参照してください。

トリガーの使用可能、使用禁止

ここでは、データベース・トリガーの管理について説明します。トピックは次のとおりです。

- トリガーを使用可能にする
- トリガーを使用禁止にする

Oracle では、対応付けられた表に対して INSERT 文または UPDATE 文、DELETE 文が発行されたときに暗黙的に実行される、データベース・トリガーと呼ばれるプロシージャを定義できます。

トリガーには、次の 2 つの異なるモードがあります。

使用可能	トリガー文を発行し、トリガー制限が TRUE と評価された場合、使用可能トリガーによってトリガー本体が実行されます。
使用禁止	トリガー文を発行し、トリガー制限が TRUE と評価された場合でも、使用禁止トリガーはトリガー本体を実行しません。

ALTER TABLE コマンドを使用してトリガーを使用禁止または使用可能にするには、表を所有しているか、表の ALTER オブジェクト権限があるか、または ALTER ANY TABLE システム権限があることが必要です。また、ALTER TRIGGER コマンドを使用してトリガーを使用禁止または使用可能にするには、トリガーを所有しているか、または ALTER ANY TRIGGER システム権限を持っていなければなりません。

トリガーを使用可能にする

使用禁止のトリガーを使用可能にするには、ENABLE オプションを指定した ALTER TRIGGER コマンドを使用します。たとえば、INVENTORY 表の REORDER という名前の使用禁止のトリガーを使用可能にするには、次の文を入力します。

```
ALTER TRIGGER reorder ENABLE;
```

ALTER TABLE コマンドの ENABLE 句と ALL TRIGGERS オプションを使用すれば、ある表に定義してあるトリガーをすべて使用可能にできます。たとえば、INVENTORY 表に定義されているトリガーをすべて使用可能にするには、次の文を入力します。

```
ALTER TABLE inventory
  ENABLE ALL TRIGGERS;
```

トリガーを使用禁止にする

次の条件の 1 つが真である場合には、一時的にトリガーを使用禁止にできます。

- トリガーの参照するオブジェクトが使用可能になっていない。

- 大規模なデータ・ロードを実行する必要がある、トリガーを実行しないでデータ・ロードを迅速に処理する。
- トリガーが適用される表にデータをロードしている。

トリガーを作成した当初は、デフォルトによって使用可能に設定されます。これを変更するには ALTER TRIGGER コマンドの DISABLE オプションを使用します。たとえば、INVENTORY 表のトリガー REORDER を使用禁止にするには、次の文を入力します。

```
ALTER TRIGGER reorder DISABLE;
```

ALTER TRIGGER コマンドの DISABLE 句の ALL TRIGGERS オプションを使用すれば、表に関連するトリガーをすべて使用禁止にできます。たとえば、表 INVENTORY のトリガーをすべて使用禁止にするには、次のように入力します。

```
ALTER TABLE inventory  
  DISABLE ALL TRIGGERS;
```

整合性制約の管理

整合性制約とは、データベース内のデータに関するルールまたは文です。制約は、データベース内で入力または更新されたときにデータをチェックし、制約のルールに従わないデータが入力されないようにします。制約は、整合性を保証したり、マスター / ディテールの関連を保ったり、式に従っているかどうかをチェックしたり、NULL を入力できないようにすることができます。

それらのルールや文は、制約が使用可能になっているときは常に真です。しかし、制約を使用禁止にする（または「使用可能であることが確認できない」と、整合性に違反したデータがデータベース内に存在する可能性があるため、この文が真であるかどうか分かりません。ここでは、整合性制約の管理のメカニズムと手順について説明します。

- 整合性制約の状態
- 制約チェックを延期する
- 対応付けられた索引をもつ制約を管理する
- 定義時に整合性制約を使用禁止、または妥当性検査なしで使用可能、使用可能にする
- 既存の整合性制約を使用可能、使用禁止にする
- 整合性制約を削除する
- 制約例外をレポートする

関連項目：制約を使用可能にするときに、特定の整合性制約に対する例外を指定できます。17-20 ページの「制約例外をレポートする」を参照してください。

整合性制約の状態

表で定義する整合性制約は、次の 3 つの状態の 1 つとすることができます。

使用禁止	<p>制約が使用禁止の場合、制約により定義したルールは、制約で指定した列のデータ値に施行されません。ただし、制約の定義は、データ・ディクショナリ内に格納されます。</p> <p>このモードは、データ・ウェアハウス・ロールアップまたはロードを実行していて、ロード処理のスピードを速くする場合に役立ちます。</p>
妥当性検査なしで使用可能	<p>施行された制約がある表には、無効なデータが入っていることがあります。無効なデータの新たな追加はできません。</p> <p>妥当性検査ありの使用可能を使用して表の中のデータの妥当性検査を行う前に即時性のある状態として有効です。このため、新しいデータが制約に違反することはできず、妥当性検査なしの使用可能から妥当性検査ありの使用可能までの制約をとるときにロックは保持されません。</p> <p>このモードは、例外に関してチェックする制約を使用可能にしない場合に役立ちます（たとえばデータ・ウェアハウス・ロードの後など）。</p>
使用可能で妥当性検査済み	<p>使用可能な制約が施行され、有効（表のデータの有効性がチェックされる）として認識されます。制約の定義は、データ・ディクショナリ内に格納されます。</p> <p>これは、制約処理の標準的な運用状態です。このモードは、正規の OLTP 処理中の無効なデータ入力を防ぐのに役立ちます。</p>

制約を使用禁止にする

整合性制約によって定義したルールを施行するには、その制約を使用可能にしなければなりません。しかし、状況によっては、次のパフォーマンス上の理由から、表の整合性制約を一時的に使用禁止にすることが望ましい場合があります。

- 表に大量のデータをロードする場合
- 表に対して大規模な変更を加えるバッチ操作を実施する場合（たとえば、既存の番号に 1000 を加えてすべての従業員番号を変更する場合）
- 一度に 1 つの表をインポートまたはエクスポートする場合

上記の 3 つの場合には、整合性制約を一時的に使用禁止にして操作のパフォーマンスを改善できます（特にデータ・ウェアハウス構成において）。

制約が使用禁止である間は、その制約に違反するデータを入力できます。したがって、上記に示した操作を終了した後は制約を使用可能にする必要があります。

妥当性検査なしで制約を使用可能にする

制約が妥当性検査なしで使用可能な状態にある場合、それ以後の文はすべて、制約に適合するかどうかチェックされます。ただし、表の中の既存のデータはチェックされません。施行された制約がある表には、無効なデータが入っていることがあります。無効なデータの新たな追加はできません。妥当性検査なしの状態では制約を使用可能にすることは、有効な OLTP データをアップロードしているデータ・ウェアハウス構成で便利です。

制約を使用可能にする場合、妥当性検査は不要です。妥当性検査なしで制約を使用可能にするほうが、妥当検査ありで制約を使用可能にするよりはるかに高速です。また、すでに使用可能になっている制約を妥当性検査する場合、妥当性検査中の DML ロックは不要です（すでに使用禁止にした制約を妥当性検査する場合とは違う）。施行によって、妥当性チェック中の違反が持ち込まれないことが保証されます。したがって、妥当性検査なしで使用可能にすれば、制約を使用可能にすることに一般に関連するダウンタイムを少なくすることができます。

制約を使用可能にする

制約が使用可能になっている場合には、制約に違反する行は表に挿入されません。しかし、制約が使用禁止になっていると、そのような行も挿入できます。その行は制約に対する例外と呼ばれます。制約が妥当性検査なしの使用可能状態にある場合、制約が使用禁止になっていた間に入力したデータから生ずる違反が残ります。制約に違反する行は、制約を使用可能にするために更新、または削除しなければなりません。

制約に違反するすべての行は、EXCEPTIONS 表で調べることができます。

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

整合性制約の状態：プロシージャと利点

整合性制約の状態を次の順序で使用すると、最高の利点が得られます。

1. 使用禁止状態
2. 操作（ロード、エクスポート、インポート）を実行する
3. 妥当性検査なしで使用可能状態
4. 使用可能状態

制約をこの順序で使用する利点は、次の通りです。

- ロックが保持されない
- すべての制約が同時に使用可能状態へジャンプできる
- 制約を使用可能にする操作がパラレルで行われる

- 表で同時のアクティビティが許される

制約チェックを延期する

Oracle が制約をチェックして、制約が満たされていない場合はエラーのシグナルが出されます。トランザクションが終わるまで、制約の有効性のチェックを延期できます。

SET CONSTRAINTS 文を発行すると、SET CONSTRAINTS モードはトランザクションの期間中、または別の SET CONSTRAINTS 文によりモードが設定しなおされるまで続きます。

注意： SET CONSTRAINT 文は、トリガーの内部では発行できません。

関連項目： SET CONSTRAINTS 文の詳細は、『Oracle8 Server SQL リファレンス』を参照してください。

制約に関する一般的な情報は、『Oracle8 Server 概要』を参照してください。

制約チェックを延期する方法

適切なデータを選択する 処理中のデータが次のような特性のものである場合、UNIQUE キーと FOREIGN キーの制約チェックを延期した方がよいかもしれません。

- 表がスナップショットである。
- 表に入っている大量のデータが別のアプリケーションによって操作されており、そのアプリケーションは同じ順序でデータを戻すかどうかわからない。
- FOREIGN キーでカスケード操作を更新する

外部のアプリケーションで操作中のデータを処理する場合、トランザクションの終わりまで有効性に関する制約のチェックを延期できます。

制約を延期可能として作成する 適切な表を識別し選択した後、表の FOREIGN キーと UNIQUE キーの制約が延期可能として作成されるようにしてください。そのためには、次のような文を発行します。

```
CREATE TABLE dept (  
    deptno NUMBER PRIMARY KEY,  
    dname VARCHAR2 (30)  
);  
  
CREATE TABLE emp (  
    empno NUMBER,  
    ename VARCHAR2 (30),  
    deptno NUMBER REFERENCES (dept),  
    CONSTRAINT epk PRIMARY KEY (empno),  
    CONSTRAINT efk FOREIGN KEY (deptno)  
REFERENCES (dept. deptno) DEFERABLE);
```

```
INSERT INTO dept VALUES (10, 'Accounting');
INSERT INTO dept VALUES (20, 'SALES');
INSERT INTO emp VALUES (1, 'Corleone', 10);
INSERT INTO emp VALUES (2, 'Costanza', 20);
COMMIT;
```

```
SET CONSTRAINT efk DEFERRED;
UPDATE dept SET deptno = deptno + 10
WHERE deptno = 20;
```

```
SELECT * from emp ORDER BY deptno;
```

EMPNO	ENAME	DEPTNO
1	Corleone	10
2	Costanza	20

```
UPDATE emp SET deptno = deptno + 10
WHERE deptno = 20;
```

```
SELECT * FROM emp ORDER BY deptno;
```

EMPNO	ENAME	DEPTNO
1	Corleone	10
2	Costanza	30

```
COMMIT;
```

すべての制約を延期に設定する データを操作するのに使われるアプリケーション内では、実際にデータの処理を始める前にすべての制約を延期に設定することが必要です。すべての制約を延期に設定するには、次の DML 文を使用します。

```
SET CONSTRAINTS ALL DEFERRED;
```

注意：SET CONSTRAINTS 文は、現行のトランザクションにだけ適用されます。制約を作成したときに指定したデフォルトでは、その制約が存在する限りそのままです。ALTER SESSION SET CONSTRAINTS 文は、現行のセッションにしか適用されません。

コミットをチェックする (オプション) COMMIT の発行直前に SET ALL CONSTRAINTS IMMEDIATE 文を発行することによって、制約違反をチェックできます。制約になにか問題がある場合、この文は失敗し、エラーの原因となっている制約が識別されます。制約違反のままコミットすると、トランザクションはロールバックされ、エラー・メッセージが表示されます。

対応付けられた索引をもつ制約を管理する

UNIQUE キーまたは PRIMARY キーを作成するときに、Oracle はその制約を施行するのに既存の索引を使用できるかどうかを調べます。そのような索引がない場合、Oracle は索引を作成します。

一意の索引と対応付けられた制約が削除されたり使用禁止にされたりすると、その索引は削除されます。Oracle では、一意でない索引を使用して UNIQUE キーと PRIMARY キーの制約を施行できます。UNIQUE 索引を自動的に作成できる場合、UNIQUE 索引と対応付けられた制約が削除されたり使用禁止にされたりすると、その索引は削除されます。

使用可能な外部キーが PRIMARY キーまたは UNIQUE キーを参照している場合、PRIMARY キーが UNIQUE キーの制約または索引を削除したり使用禁止にしたりはできません。

注意： 延期できる UNIQUE キーと PRIMARY キーはすべて、一意でない索引を使用する必要があります。常に、使用禁止状態で UNIQUE キーと PRIMARY キーの制約を作成してください。次に、表に索引を作成し、制約を使用可能にしてください。そうすると、制約を使用禁止にしても索引は削除されません。

定義時に整合性制約を使用禁止、または妥当性検査なしで使用可能、使用可能にする

CREATE TABLE 文または ALTER TABLE 文で整合性制約を定義するときに、制約の定義に ENABLE 句、DISABLE 句、ENABLE NOVALIDATE 句を入れることにより、その制約を使用可能、使用禁止、妥当性検査なしの使用可能にすることができます。制約定義にこれらの句のどれも指定しない場合、その制約は自動的に使用可能にされ、妥当性検査されます。

定義時に制約を使用禁止にする

次の CREATE TABLE 文と ALTER TABLE 文は、整合性制約を定義し、使用禁止にします。

```
CREATE TABLE emp (  
    empno NUMBER(5) PRIMARY KEY DISABLE,    . . . ;
```

```
ALTER TABLE emp  
    ADD PRIMARY KEY (empno) DISABLE;
```

整合性制約を定義して使用禁止にする ALTER TABLE 文は、表の行がその整合性制約に違反しているために、異常終了することはありません。制約の定義は、そのルールが施行されていないので許可されます。

関連項目： 制約例外の詳細は、17-20 ページの「制約例外をレポートする」を参照してください。

定義時に制約を使用可能にする

次の CREATE TABLE 文と ALTER TABLE 文は、整合性制約を定義し、使用可能にします。

```
CREATE TABLE emp (
    empno NUMBER(5) CONSTRAINT emp.pk PRIMARY KEY,    . . . ;
ALTER TABLE emp
    ADD CONSTRAINT emp.pk PRIMARY KEY (empno);
```

整合性制約を定義し、使用可能にしようとする ALTER TABLE 文は、表の行が整合性制約に違反している場合、異常終了する可能性があります。その場合、その文はロールバックされ、制約定義は格納されず使用可能にもされません。

対応する索引を作成する UNIQUE キーまたは PRIMARY KEY を使用可能にするには、表の所有者は索引を収録する表領域の割当て制限、または UNLIMITED TABLESPACE システム権限が必要です。

既存の整合性制約を使用可能、使用禁止にする

ALTER TABLE コマンドで、ENABLE 句を指定して使用禁止の制約を使用可能にしたり、DISABLE 句を指定して使用可能の制約を使用禁止にしたりできます。

使用可能状態の制約を使用禁止にする

次の文は、使用可能状態の整合性制約を使用禁止にします。

```
ALTER TABLE dept
    DISABLE CONSTRAINT dname_ukey;
ALTER TABLE dept
    DISABLE PRIMARY KEY,
    DISABLE UNIQUE (dname, loc);
```

UNIQUE キー制約または PRIMARY KEY 制約、およびすべての依存する FOREIGN KEY 制約を一度で使用禁止または削除するには、DISABLE 句または DROP 句の CASCADE オプションを使用してください。たとえば、次の文は PRIMARY KEY 制約とこれに依存する FOREIGN KEY 制約を使用禁止にします。

```
ALTER TABLE dept
    DISABLE PRIMARY KEY CASCADE;
```

使用禁止にした制約を妥当性検査なしで使用可能にする

妥当性検査なしで制約を使用可能すると、新しい文についてだけ制約に従っているかどうかをチェックされます。したがって、妥当性検査なしで制約を使用可能にするほうが古いデータがチェックされないので制約を使用可能にするよりはるかに高速です。また、すでに施行された制約を使用可能にする場合、妥当性検査中の DML ロックは不要です (すでに使用禁止にした制約を妥当性検査する場合とは違う)。制約を使用可能にすると、妥当性チェック中の違反が持ち込まれないことが保証されます。

次の文は、妥当性検査なしで使用禁止にした整合性制約を使用可能にします。

```
ALTER TABLE dept
    ENABLE NOVALIDATE CONSTRAINT dname_ukey;
ALTER TABLE dept
    ENABLE NOVALIDATE PRIMARY KEY,
    ENABLE NOVALIDATE UNIQUE (dname, loc);
```

対応付けられる索引を作成する UNIQUE キーまたは PRIMARY キーを使用可能にするか、妥当性検査なしで使用可能にするには、表の所有者に、索引を収録する表領域に対する割当て制限、または UNLIMITED TABLESPACE システム権限が必要です。UNIQUE キーまたは PRIMARY キーが既存の索引を使用しているときは、索引が作成されず、割当て制限は不要です。

制約を妥当性検査なしで使用可能にするときは、UNIQUE キーと PRIMARY キーに一意でない索引を使用し、それらを作成しないで済むようにしてください。

注意： DML を保持せずに妥当性検査なしで使用可能にされた制約を使用可能にするには、各 ALTER TABLE ENABLE 文が制約を 1 つだけ有効にする必要があります。

使用禁止状態の制約を使用可能にする

次の文は、使用禁止状態の整合性制約を使用可能にします。

```
ALTER TABLE dept
    ENABLE CONSTRAINT dname_ukey;
ALTER TABLE dept
    ENABLE PRIMARY KEY,
    ENABLE UNIQUE (dname, loc);
```

整合性制約を使用可能にしようとする ALTER TABLE 文は、表の行がその整合性制約に違反している場合、異常終了する可能性があります。この場合には、文はロールバックされ、制約は使用可能になりません。

対応する索引を作成する UNIQUE キーまたは PRIMARY KEY を使用可能にするには、表の所有者は索引を収録する表領域の割当て制限、または UNLIMITED TABLESPACE システム権限が必要です。

注意： 制約を使用可能にする前に制約を ENABLE NOVALIDATE で施行する場合、ENABLE の間に保持されるロックはありません。このため、次のことが許されます。

- すべての制約を同時に使用可能にする
 - 各制約を内部でパラレル化する
 - 表での同時アクティビティを許可する
-

整合性制約を削除する

整合性制約は、施行するルールが真でなくなった場合、またはその制約が必要でなくなった場合に削除できます。整合性制約は、ALTER TABLE コマンドに DROP 句を指定して削除します。次の 2 つの文は、整合性制約を削除します。

```
ALTER TABLE dept
  DROP UNIQUE (dname, loc);
ALTER TABLE emp
  DROP PRIMARY KEY,
  DROP CONSTRAINT dept_fkey;
```

UNIQUE キー制約と PRIMARY KEY 制約を削除すると、対応する索引が削除されます。また、FOREIGN KEY が UNIQUE キーまたは PRIMARY KEY を参照している場合、DROP 文に CASCADE CONSTRAINTS 句を指定しなければ、制約を削除できません。

制約例外をレポートする

CREATE TABLE...ENABLE... 文または ALTER TABLE...ENABLE... 文を発行する際にまったく例外がなければ、整合性制約は使用可能となり、続くすべての DML 文は、その使用可能となった整合性制約に従います。

制約が使用可能となる際に例外が存在する場合は、エラーが戻され、整合性制約は使用禁止の状態のままです。整合性制約の例外が存在しているため文が正常に実行されない場合、文はロールバックされます。例外が存在している場合には、制約に対するすべての例外を更新または削除するまで、制約を使用可能にできません。

CREATE TABLE 文を使用して、どの行が違反しているかを判断することはできません。整合性制約に違反している行を判断するためには、ENABLE 句に EXCEPTIONS オプションを指定して ALTER TABLE 文を発行します。EXCEPTIONS オプションにより、すべての例外となる行の ROWID、表所有者、表名、制約名はある特定の表に設定されます。

注意： 制約を使用可能にする前に、ENABLE 句の EXCEPTION オプションから情報を受け取る、適切な例外レポート表を作成しなければなりません。例外の表は、スクリプト UTLEXCPT.SQL を送ることによって作成でき、それにより EXCEPTIONS という名前の表が作成されます。また、スクリプトを修正し、再実行することによって、新たに別の名前で例外表を作成できます。

次の文は DEPT 表の PRIMARY KEY を使用可能にしようとします。例外が存在した場合は、EXCEPTIONS という名前の表に情報が挿入されます。

```
ALTER TABLE dept ENABLE PRIMARY KEY EXCEPTIONS INTO exceptions;
```

重複する主キー値が DEPT 表に存在し、DEPT の PRIMARY KEY 制約の名前が SYS_C00610 である場合には、前の文によって次の行が EXCEPTIONS 表の中に入れられることがあります。

```
SELECT * FROM exceptions;
```

ROWID	OWNER	TABLE_NAME	CONSTRAINT
AAAAZ9AABAAABvqAAB	SCOTT	DEPT	SYS_C00610
AAAAZ9AABAAABvqAAG	SCOTT	DEPT	SYS_C00610

問合せ情報により、例外レポート表とマスター表の行を結合し、特定の制約に違反している実際の行を記述します。

```
SELECT deptno, dname, loc FROM dept, exceptions
WHERE exceptions.constraint = 'SYS_C00610'
AND dept.rowid = exceptions.row_id;
```

DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK
10	RESEARCH	DALLAS

制約が指定された表から、制約に違反しているすべての行を更新または削除しなければなりません。例外を更新する場合には、制約に違反する値を、制約または NULL と一致する値に変更しなければなりません。マスター表の行を更新または削除した後で、例外レポート表の例外に対応する行は、他の例外レポートとの混同を避けるために削除しておかなければなりません。また、マスター表と例外レポート表を更新する文は、必ず同一トランザクション上で指定し、トランザクション一貫性を保証しなければなりません。

前の例の例外を訂正するために、次のトランザクションを発行できます。

```
UPDATE dept SET deptno = 20 WHERE dname = 'RESEARCH';
DELETE FROM exceptions WHERE constraint = 'SYS_C00610';
COMMIT;
```

例外を管理する上での最終的な目的は、例外レポート表の例外をすべて排除することにあります。

注意： 制約が使用禁止になっている表の現在の例外を訂正している場合は、他のユーザーが新しい例外を作成する文を発行する可能性があります。これを避けるには、例外の排除を行う前に制約を妥当性検査なしで使用可能にします。

関連項目： UTLEXCPT.SQL スクリプトの正確な名前と位置は、オペレーティング・システムによって異なります。詳細は、オペレーティング・システム固有の Oracle マニュアルを参照してください。

オブジェクトの依存性の管理

ここでは、様々なオブジェクトの依存性について説明します。トピックは次のとおりです。

- ビューを手動で再コンパイルする
- プロシージャとファンクションを手動で再コンパイルする
- パッケージを手動で再コンパイルする

まず、表 17-1 を見てください。この表は、オブジェクトが依存している他のオブジェクト内の変更によってどのような影響を受けるかを示したものです。

表 17-1 オブジェクトの状態に影響を与える操作

操作	結果としてのオブジェクトの状態	結果としての依存オブジェクトの状態
CREATE 表、順序、シノニム	VALID(エラーがない場合)	変化なし ¹
ALTER 表(ADD 列 MODIFY 列) RENAME 表、順序、シノニム、ビュー	VALID(エラーがない場合)	INVALID
DROP 表、順序、シノニム、ビュー、 プロシージャ、ファンクション、パッケージ	変化なし。オブジェクトは削除される	INVALID
CREATE ビューまたはプロシージャ ²	VALID (エラーがない場合) INVALID (構文エラー、認可エラーの場合)	変化なし ¹
CREATE OR REPLACE ビューまたは プロシージャ ²	VALID (エラーがない場合) INVALID (構文エラー、認可エラーの場合)	INVALID
REVOKE オブジェクト権限 ³ ON オブジェクト TO/FROM ユーザー	変化なし	INVALID (オブジェクトに依存するユーザーのすべてのオブジェクト) ³
REVOKE オブジェクト権限 ³ ON オブジェクト TO/FROM PUBLIC	変化なし	INVALID (オブジェクトに依存するデータベース内のすべてのオブジェクト) ³

表 17-1 オブジェクトの状態に影響を与える操作

操作	結果としてのオブジェクトの状態	結果としての依存オブジェクトの状態
REVOKE システム権限 ⁴ TO/FROM ユーザー	変化なし	INVALID (ユーザーのすべてのオブジェクト) ⁴
REVOKE システム権限 ⁴ TO/FROM PUBLIC	変化なし	INVALID (データベース内のすべてのオブジェクト) ⁴
<p>1. 以前にオブジェクトが存在しなかった場合は、依存オブジェクトを INVALID にすることがあります。</p> <p>2. スタンドアロンのプロシージャ、ファンクション、パッケージ、トリガー</p> <p>3.SELECT、INSERT、UPDATE、DELETE、EXECUTE などの DML オブジェクト権限だけ。妥当性の再検査での再コンパイルは必要ありません。</p> <p>4.SELECT、INSERT、UPDATE、DELETE ANY TABLE、EXECUTE ANY PROCEDURE などの DML システム権限だけ。妥当性の再検査での再コンパイルは必要ありません。</p>		

Oracle は、無効なビューまたは PL/SQL プログラム・ユニットを、それが次に使われるときに自動的に再コンパイルします。さらに、SQL コマンドに COMPILE パラメータを指定して、無効なビューまたはプログラム・ユニットを強制的にコンパイルすることもできます。強制コンパイルは、依存ビューまたはプログラム・ユニットが無効であることがわかっていて、現在使用していない場合に、エラーを検査する目的で指定します。この場合、ビューまたはプログラム・ユニットが実行されない限り、自動再コンパイルは実行されません。無効な依存オブジェクトを識別するには、ビュー USER_/ALL_/DBA_OBJECTS を問い合せてください。

ビューを手動で再コンパイルする

ビューを手動で再コンパイルするには、そのビューが自分のスキーマ内にあるか、または ALTER ANY TABLE システム権限を持っていないければなりません。ビューを再コンパイルするには、COMPILE パラメータを指定した ALTER VIEW コマンドを使用します。次の文は、自分のスキーマ内の EMP_DEPT ビューを再コンパイルします。

```
ALTER VIEW emp_dept COMPILE;
```

プロシージャとファンクションを手動で再コンパイルする

プロシージャを手動で再コンパイルするには、そのプロシージャが自分のスキーマ内にあるか、または ALTER ANY PROCEDURE システム権限を持っていないければなりません。スタンドアロンのプロシージャまたはファンクションを再コンパイルするには、COMPILE パラ

メータを指定した ALTER PROCEDURE/FUNCTION コマンドを使います。次の文は、自分のスキーマ内のストアド・プロシージャ UPDATE_SALARY を再コンパイルします。

```
ALTER PROCEDURE update_salary COMPILE;
```

パッケージを手動で再コンパイルする

パッケージを手動で再コンパイルするには、そのパッケージが自分のスキーマ内にあるか、または ALTER ANY PROCEDURE システム権限を持っていなければなりません。パッケージ本体、またはパッケージ本体とパッケージ仕様部の両方を再コンパイルするには、COMPILE パラメータを指定した ALTER PACKAGE コマンドを使用します。次の文はそれぞれ、パッケージ ACCT_MGMT の本体だけ、および本体と仕様部の両方を再コンパイルします。

```
ALTER PACKAGE acct_mgmt COMPILE BODY;  
ALTER PACKAGE acct_mgmt COMPILE PACKAGE;
```

オブジェクト名の名前の変換の管理

ここでは、Oracle でオブジェクト名を分解する方法を説明します。

1. Oracle は、SQL 文で参照される名前の最初の断片を識別しようとします。たとえば、SCOTT.EMP では SCOTT が最初の断片です。1 つの断片だけが存在する場合、1 つの断片は、最初の断片と見なされます。
 - a. Oracle は、現行のスキーマで、オブジェクトの最初の断片に一致するオブジェクトを検索します。そのようなオブジェクトが見つからない場合には、ステップ b を続けます。
 - b. 現行のスキーマでスキーマ・オブジェクトが見つからない場合、Oracle は、オブジェクトの最初の断片に一致するパブリック・シノニムを検索します。そのようなシノニムが見つからない場合には、ステップ c を続けます。
 - c. パブリック・シノニムも見つからない場合、Oracle は、オブジェクトの最初の断片に一致するスキーマを検索します。スキーマが見つかったら、ステップ a へ戻り、識別されたスキーマ内で見つけるオブジェクトとして名前の 2 番目の断片を使用します。2 番目の断片が前に識別されたスキーマ内のオブジェクトに一致しない場合、または 2 番目の断片がない場合には、エラーが戻されます。

ステップ c でスキーマが見つからない場合、そのオブジェクトは識別できず、エラーが戻されます。
2. スキーマ・オブジェクトが識別されました。SQL 文で与えられた名前の残りの断片（ある場合）は、見付けられたオブジェクトの妥当な部分に一致しなければなりません。たとえば、文で SCOTT.EMP.DEPTNO が与えられ、SCOTT がスキーマとして識別され、EMP が表として識別される場合には、（EMP が表であるため）DEPTNO は列に対応しなければなりません。また、EMP がパッケージとして識別された場合、DEPTNO はその

パッケージのパブリック定数、または変数、プロシージャ、ファンクションに対応しなければなりません。

明示的、またはシノニム内で間接的に分散データベースにおいてグローバル・オブジェクト名が使われるとき、ローカル Oracle はその参照をローカルに分解します。たとえば、シノニムをリモート表のグローバル・オブジェクト名に分解します。部分的に分解された文はリモート・データベースに転送され、リモート Oracle が、ここで説明したオブジェクトの分解を完了します。

データ・ディクショナリの記憶領域パラメータの変更

ここでは、データ・ディクショナリの記憶領域パラメータについて説明します。トピックは次のとおりです。

- データ・ディクショナリの構造
- データ・ディクショナリの記憶領域の変更が必要なエラー

データベースが非常に大規模であったり、または著しい数のオブジェクト、表の列、制約定義、ユーザー、その他の定義を含んでいる場合、データ・ディクショナリを構成する表が、どこかの時点で追加のエクステントを取得できなくなる可能性があります。たとえば、データ・ディクショナリ表が追加のエクステントを必要としていても、SYSTEM 表領域内に十分な連続領域がない場合です。このような場合、オブジェクトを保持する表領域に十分な領域があるように見えても、新しいオブジェクトを作成できません。このような状況を改善するために、ユーザーの作成したセグメントの記憶設定を変更する場合と同じ方法で、基礎となるデータ・ディクショナリ表の記憶領域パラメータを変更し、より多くのエクステントが割り当てられるようにできます。たとえば、データ・ディクショナリ表の NEXT または PCTINCREASE の値を調整できます。

警告： データ・ディクショナリのオブジェクトの記憶領域の設定を変更するときは、注意してください。不適切な設定を選択すると、データ・ディクショナリの構造が損傷し、データベース全体を作成しなおさなければならぬおそれがあります。たとえば、データ・ディクショナリ表 USER\$ の PCTINCREASE を 0、NEXT を 2K に設定すると、その表はすぐにセグメントの最大エクステントに達し、データベース全体をエクスポートし、作成しなおし、インポートしなければ、ユーザーまたはロールをそれ以上作成できなくなります。

データ・ディクショナリの構造

次の表とクラスタは、ユーザーがデータベース内に作成したすべてのオブジェクトの定義を含みます。

SEG\$	データベース内に定義されているセグメント (一時セグメントを含む)。
-------	---------------------------------------

OBJ\$	データベース内のユーザー定義オブジェクト（クラスタ化した表を含む）。I_OBJ1 と I_OBJ2 による索引付き。
UNDO\$	データベース内に定義されているロールバック・セグメント。I_UNDO1 による索引付き。
FET\$	どのセグメントにも割り当てられていない使用可能な空きエクステント。
UET\$	セグメントに割り当てられているエクステント。
TS\$	データベース内に定義されている表領域。
FILE\$	データベースを構成するファイル。I_FILE1 による索引付き。
FILEXT\$	AUTOEXTEND オプションがオンに設定されているデータ・ファイル。
TAB\$	データベース内に定義されている表（クラスタ化した表を含む）。I_TAB1 による索引付き。
CLU\$	データベース内に定義されているクラスタ。
IND\$	データベース内に定義されている索引。 I_IND1 による索引付き。
ICOL\$	索引を定義されている列（複合索引内の各列の個々のエントリを含む）。I_COL1 による索引付き。
COL\$	データベース内の表に定義されている列。 I_COL1 と I_COL2 による索引付き。
CON\$	データベース内に定義されている制約（制約の所有者に関する情報を含む）。I_CON1 と I_CON2 による索引付き。
CDEF\$	CON\$ 内の制約の定義。I_CDEF1、I_CDEF2、I_CDEF3 による索引付き。
CCOL\$	制約を定義されている列（複合キー内の各列の個々のエントリを含む）。I_CCOL1 による索引付き。

USER\$	データベース内に定義されているユーザーとロール。I_USER1 による索引付き。
TSQ\$	ユーザーに対する表領域の割当て制限（各ユーザーに定義されている各表領域の割当て制限ごとに 1 エントリを含む）。
C_OBJ#	TAB\$、CLU\$、ICOL\$、IND\$、COL\$ を含むクラスタ。I_OBJ# による索引付き。
C_TS#	FET\$、TS\$、FILE\$ を含むクラスタ。I_TS# による索引付き。
C_USER#	USER\$ と TSQ\$ を含むクラスタ。I_USER# による索引付き。
C_COBJ#	CDEF\$ と CCOL\$ を含むクラスタ。I_COBJ# による索引付き。

すべてのデータ・ディクショナリ・セグメントの中で、変更する可能性があるセグメントは次のとおりです。

C_TS#	データベース内の空き領域がかなり断片化している場合
C_OBJ#	表に多くの索引または多くの列がある場合
CON\$, C_COBJ#	整合性制約を頻繁に使用する場合
C_USER#	データベースに多数のユーザーが定義されている場合

クラスタ化した表については、その表の記憶設定ではなく、クラスタの記憶設定を変更しなければなりません。

データ・ディクショナリの記憶領域の変更が必要なエラー

Oracle がデータ・ディクショナリに追加のエクステントを割り当てなければならないような新しいオブジェクトをユーザーが作成しようとして、Oracle がエクステントを割り当てることができないと、エラーが戻されます。エラー・メッセージ ORA-1653「表領域～内でサイズ～のエクステントの割当てエラーが発生しました。」は、このような問題を示します。

このエラー・メッセージを受け取り、変更しようとしていたセグメントがその定義に指定された制限に到達していない場合、その定義を含むオブジェクトの記憶設定をチェックしてください。

たとえば、表に対して新しい PRIMARY KEY 制約を定義しようとして、Oracle が作成しなければならないキーの索引のために十分な領域が存在しているにもかかわらず、ORA-1547 を受け取った場合、CON\$ または C_COBJ# を別のエクステンツに割り当てることができるかどうか、チェックしてください。このために、DBA_SEGMENTS を問い合わせ、CON\$ または C_COBJ# の記憶領域パラメータを変更することを検討してください。

関連項目：詳細は、17-32 ページの「例 7: 追加のエクステンツを割り当てることのできないセグメントを表示する」を参照してください。

スキーマ・オブジェクトについての情報を表示

データ・ディクショナリには、第 10 章～第 16 章までで説明したスキーマ・オブジェクトに関する多くのビューがあります。次に、スキーマ・オブジェクトに関するビューをまとめます。

- ALL_OBJECTS、USER_OBJECTS、DBA_OBJECTS
- ALL_CATALOG、USER_CATALOG、DBA_CATALOG
- ALL_TABLES、USER_TABLES、DBA_TABLES
- ALL_TAB_COLUMNS、USER_TAB_COLUMNS、DBA_TAB_COLUMNS
- ALL_TAB_COMMENTS、USER_TAB_COMMENTS
- ALL_COL_COMMENTS、USER_COL_COMMENTS、DBA_COL_COMMENTS
- ALL_VIEWS、USER_VIEWS、DBA_VIEWS
- ALL_INDEXES、USER_INDEXES、DBA_INDEXES
- ALL_IND_COLUMNS、USER_IND_COLUMNS、DBA_IND_COLUMNS
- USER_CLUSTERS、DBA_CLUSTERS
- USER_CLU_COLUMNS、DBA_CLU_COLUMNS
- ALL_SEQUENCES、USER_SEQUENCES、DBA_SEQUENCES
- ALL_SYNONYMS、USER_SYNONYMS、DBA_SYNONYMS
- ALL_DEPENDENCIES、USER_DEPENDENCIES、DBA_DEPENDENCIES

データベースのセグメントに関する情報は、次のデータ・ディクショナリ・ビューに含まれています。

- USER_SEGMENTS
- DBA_SEGMENTS

データベースのエクステンツに関する情報は、次のデータ・ディクショナリ・ビューに含まれています。

- USER_EXTENTS

- DBA_EXTENTS
- USER_FREE_SPACE
- DBA_FREE_SPACE

ディクショナリの記憶領域の Oracle パッケージ

表 17-2 は、PL/SQL で一部の SQL 機能にアクセスできるようにするか、またはデータベースの機能を拡張するために、Oracle で提供されるパッケージについて説明します。

表 17-2 提供されるパッケージ：追加機能

プロシージャ	説明
dbms_space.unused_space	オブジェクト（表または索引、クラスタ）の中の使われていない領域に関する情報を戻す。
dbms_space.free_blocks	オブジェクト（表または索引、クラスタ）の中の空きブロックに関する情報を戻す。
dbms_session.free_unused_user_memory	大容量の（100KB より大きい）メモリーを必要とする操作を実行したあと、未使用のメモリーを再生させるためのプロシージャ。このプロシージャは、メモリーに余裕のない場合にだけ使用する。
dbms_system.set_sql_trace_in_session	シリアル番号と SID（これらの値は v\$session に入っている）によって識別されるセッションの sql_trace を使用可能にする。

次の例では、様々なスキーマ・オブジェクトを表示する方法を示します。

例 1: スキーマ・オブジェクトをタイプ別に表示する

次の問合せは、問合せを発行しているユーザーによって所有されるオブジェクトのすべてをリストします。

```
SELECT object_name, object_type FROM user_objects;
```

OBJECT_NAME	OBJECT_TYPE
EMP_DEPT	CLUSTER
EMP	TABLE
DEPT	TABLE
EMP_DEPT_INDEX	INDEX
PUBLIC_EMP	SYNONYM
EMP_MGR	VIEW

例 2: 例情報を表示する

_COLUMNS 接尾辞で終わるビューの 1 つを使用して、名前、データ型、長さ、精度、位取り、デフォルト・データ値などの列情報を記述できます。たとえば、次の問合せは、EMP 表と DEPT 表のデフォルト列値のすべてをリストします。

```
SELECT table_name, column_name, data_default
FROM user_tab_columns
WHERE table_name = 'DEPT' OR table_name = 'EMP';
```

TABLE_NAME	COLUMN_NAME	DATA_DEFAULT
DEPT	DEPTNO	
DEPT	DNAME	
DEPT	LOC	'NEW YORK'
EMP	EMPNO	
EMP	ENAME	
EMP	JOB	
EMP	MGR	
EMP	HIREDATE	SYSDATE
EMP	SAL	
EMP	COMM	
EMP	DEPTNO	

列はすべてユーザー指定デフォルトを持っていません。これらの列は、デフォルトとして NULL を想定します。

例 3: ビューとシノニムの依存性を表示する

ビューまたはシノニムを作成するとき、ビューやシノニムはその基礎になるベース・オブジェクトに基づきます。ALL/USER/DBA_DEPENDENCIES データ・ディクショナリ・ビューはビューの依存性を知るのに使用することができ、ALL/USER/DBA_SYNONYMS データ・ディクショナリ・ビューは、シノニムのベース・オブジェクトをリストするために使用できます。たとえば、次の問合せは、ユーザー JWARD によって作成されたシノニムのベース・オブジェクトをリストします。

```
SELECT table_owner, table_name, synonym_name
FROM sys.dba_synonyms
WHERE owner = 'JWARD';
```

TABLE_OWNER	TABLE_NAME	SYNONYM_NAME
SCOTT	DEPT	DEPT
SCOTT	EMP	EMP

例 4: 一般的なセグメント情報を表示する

次の問合せは、各ロールバック・セグメントの名前、各ロールバック・セグメントを含む表領域、各ロールバック・セグメントのサイズを戻します。

```
SELECT segment_name, tablespace_name, bytes, blocks, extents
FROM sys.dba_segments
WHERE segment_type = 'ROLLBACK';
```

SEGMENT_NAME	TABLESPACE_NAME	BYTES	BLOCKS	EXTENTS
-----	-----	-----	-----	-----
RS1	SYSTEM	20480	10	2
RS2	TS1	40960	20	3
SYSTEM	SYSTEM	184320	90	3

例 5: 一般的なエクステント情報を表示する

データベース内に現在割り当てられているエクステントに関する一般的な情報は、DBA_EXTENTS データ・ディクショナリ・ビューに格納されます。たとえば、次の問合せによって、ロールバック・セグメントに対応付けられたエクステントとそれらのエクステントのサイズがそれぞれ識別されます。

```
SELECT segment_name, bytes, blocks
FROM sys.dba_extents
WHERE segment_type = 'ROLLBACK';
```

SEGMENT_NAME	BYTES	BLOCKS
-----	-----	-----
RS1	10240	5
RS1	10240	5
SYSTEM	51200	25
SYSTEM	51200	25
SYSTEM	51200	25

SYSTEM ロールバック・セグメントは、50KB の 3 つの等しいサイズのエクステントから構成され、RS1 ロールバック・セグメントは、それぞれ 10KB の 2 つのエクステントから構成されていることがわかります。

例 6: データベースの空き領域（エクステント）を表示する

データベース内の使用可能エクステント（どのセグメントにも割り当てられていない）に関する情報は、DBA_FREE_SPACE データ・ディクショナリ・ビューに格納されています。たとえば、次の問合せは、各表領域内の使用可能エクステントによって利用できる空き領域を示します。

```
SELECT tablespace_name, file_id, bytes, blocks
FROM sys.dba_free_space;
```

TABSPACE_NAME	FILE_ID	BYTES	BLOCKS
-----	-----	-----	-----
SYSTEM	1	8120320	3965
SYSTEM	1	10240	5
TS1	2	10432512	5094

例 7: 追加のエクステントを割り当てることのできないセグメントを表示する

また、DBA_FREE_SPACE を、ビュー DBA_SEGMENTS、DBA_TABLES、DBA_CLUSTERS、DBA_INDEXES、DBA_ROLLBACK_SEGS と組み合わせて使用して、他のセグメントに追加のエクステントを割り当てることのできないかどうか、判断できます。

セグメントは、次のどちらかの理由でエクステントを割り当てることのできない可能性があります。

- セグメントを含む表領域に次のエクステントのための十分な領域がない。
- セグメントに、データ・ディクショナリに記録されている、最大数のエクステントが存在する。
- セグメントに、データ・ブロック・サイズによって許可される、最大数のエクステントが存在する。これは、オペレーティング・システム固有の値です。

注意： MAXEXTENTS の STORAGE 句の値を UNLIMITED にできますが、データ・ディクショナリ表は許容されるブロックの最大値より大きな MAXEXTENTS を持つことはできません。したがって、データ・ディクショナリ表を制限なしのフォーマットに変換できません。

次の問合せは、上記の基準のどちらかに適合するすべてのセグメントの名前および所有者、表領域を戻します。

```
SELECT seg.owner, seg.segment_name,
       seg.segment_type, seg.tablespace_name,
       DECODE(seg.segment_type,
              'TABLE', t.next_extent,
              'CLUSTER', c.next_extent,
              'INDEX', i.next_extent,
              'ROLLBACK', r.next_extent)
FROM sys.dba_segments seg,
     sys.dba_tables t,
     sys.dba_clusters c,
     sys.dba_indexes i,
     sys.dba_rollback_segs r

WHERE ((seg.segment_type = 'TABLE'
       AND seg.segment_name = t.table_name
       AND seg.owner = t.owner
```

```

AND NOT EXISTS (SELECT tablespace_name
                  FROM dba_free_space free
                  WHERE free.tablespace_name = t.tablespace_name
                  AND free.bytes >= t.next_extent))
OR (seg.segment_type = 'CLUSTER'
    AND seg.segment_name = c.cluster_name
    AND seg.owner = c.owner
    AND NOT EXISTS (SELECT tablespace_name
                      FROM dba_free_space free
                      WHERE free.tablespace_name = c.tablespace_name
                      AND free.bytes >= c.next_extent))
OR (seg.segment_type = 'INDEX'
    AND seg.segment_name = i.index_name
    AND seg.owner = i.owner
    AND NOT EXISTS (SELECT tablespace_name
                      FROM dba_free_space free
                      WHERE free.tablespace_name = i.tablespace_name
                      AND free.bytes >= i.next_extent))
OR (seg.segment_type = 'ROLLBACK'
    AND seg.segment_name = r.segment_name
    AND seg.owner = r.owner
    AND NOT EXISTS (SELECT tablespace_name
                      FROM dba_free_space free
                      WHERE free.tablespace_name = r.tablespace_name
                      AND free.bytes >= r.next_extent)))
OR seg.extents = seg.max_extents OR seg.extents = data_block_size;

```

注意： この問合せを使用するときには、data_block_size をシステムのデータ・ブロック・サイズで置き換えてください。

追加のエクステントを割り当てることのできないセグメントを識別すれば、その原因に応じて、次のどちらかの方法で問題を解決できます。

- 表領域が満杯であれば、表領域にデータ・ファイルを追加する。
- セグメントに多くのエクステントが存在し、セグメントの MAXEXTENTS を大きくできない場合は、次の手順を実行する。最初に、セグメント内のデータをエクスポートします。次に、多くのエクステントを割り当てる必要がないように INITIAL の設定を大きくして、セグメントを削除して作成しなおします。最後に、データをセグメントにインポートします。

第4部

データベース・セキュリティ

ロールバック・セグメントの管理

この章では、ロールバック・セグメントを管理する方法について説明します。トピックは次のとおりです。

- ロールバック・セグメントを管理するためのガイドライン
- ロールバック・セグメントの作成
- ロールバック・セグメントの記憶領域パラメータを指定
- ロールバック・セグメントのオンライン、オフライン
- ロールバック・セグメントにトランザクションを明示的に割り当て
- ロールバック・セグメントの削除
- ロールバック・セグメント情報を監視

関連項目：Parallel Server オプションを使用する場合、『Oracle8 Parallel Server 概要および管理』を参照してください。

この章には、Oracle Enterprise Manager について述べている箇所がいくつかあります。Enterprise Manager/GUI または Server Manager/LineMode を使って特定のタスクを実行する方法の詳細は、『Oracle Server Manager ユーザーズ・ガイド』または『Oracle Enterprise Manager 管理者ガイド』を参照してください。

ロールバック・セグメントを管理するためのガイドライン

ここでは、データベースのロールバック・セグメントを作ったり管理したりする前に考慮すべきガイドラインについて説明します。トピックは次のとおりです。

- 複数ロールバック・セグメントを使用する
- パブリックとプライベートのロールバック・セグメントの選択
- 自動的に獲得するロールバック・セグメントの指定
- ロールバック・セグメント・サイズの設定
- 等しいサイズのエクステントを数多く持つロールバック・セグメントを作る
- 各ロールバック・セグメントに対するエクステントの最適数の設定
- ロールバック・セグメントに対する記憶位置の設定

すべてのデータベースには、1 つまたは複数のロールバック・セグメントが含まれています。ロールバック・セグメントとは、トランザクションがロールバックされるイベントでのトランザクションのアクションが記録されるデータベースの部分です。ロールバック・セグメントを使って、読み込みの一貫性を実現し、トランザクションのロールバック、データベースの回復ができます。

関連項目：ロールバック・セグメントの詳細は、『Oracle8 Server 概要』を参照してください。

複数ロールバック・セグメントを使用する

複数のロールバック・セグメントを使うと、多くのセグメントにわたるロールバック・セグメントの競合が分散され、システム・パフォーマンスが改善されます。複数のロールバック・セグメントは、次の状況で必要とされます。

- データベースが作られると、SYSTEM という名前の単一ロールバック・セグメントが SYSTEM 表領域に作られます。SYSTEM 表領域以外にオブジェクトを作成できますが、SYSTEM 表領域に少なくとも 1 つの追加のロールバック・セグメントを作ってオンラインにするまでは、それらに書き込むことはできません。
- 多くのトランザクションが同時に処理されているときには、より多くのロールバック情報が同時に生成されます。インスタンスに対して用意する同時実行のトランザクション数はパラメータ TRANSACTIONS で指定し、各ロールバック・セグメントが処理すべきトランザクション数はパラメータ TRANSACTIONS_PER_ROLLBACK_SEGMENT で指定します。インスタンスは、データベースのオープン時に、少なくとも $\text{TRANSACTIONS} / \text{TRANSACTIONS_PER_ROLLBACK_SEGMENT}$ で算出される数のロールバック・セグメントを獲得して、最大量のトランザクションを処理しようとします。したがって、パラメータを設定した後で、TRANSACTIONS / TRANSACTIONS_PER_ROLLBACK_SEGMENT ロールバック・セグメントを作ってください。

関連項目：Oracle Parallel Server 環境では、インスタンスを起動するためには、各インスタンスは SYSTEM ロールバック・セグメントの他にそれ自身のロールバック・セグメントにア

クセスできなければなりません。詳細は、『Oracle8 Parallel Server 概要および管理』を参照してください。

SYSTEM 表領域にロールバック・セグメントを追加する

SYSTEM という名前の初期ロールバック・セグメントが、データベースを作るときに作られます。SYSTEM ロールバック・セグメントは、対応するデフォルトの記憶領域パラメータを使って、SYSTEM 表領域内に作られます。このロールバック・セグメントは削除できません。

インスタンスは、必要な他のロールバック・セグメントに加えて、常に SYSTEM ロールバック・セグメントを獲得します。しかし、ロールバック・セグメントが複数ある場合には、Oracle は SYSTEM ロールバック・セグメントを特別なシステム・トランザクションだけに使って、ユーザー・トランザクションを他のロールバック・セグメントに分散させようとしています。SYSTEM 以外のロールバック・セグメントに対するトランザクションが多すぎる場合には、Oracle は SYSTEM ロールバック・セグメントを使います。したがって、データベースを作った後、SYSTEM 表領域内に追加のロールバック・セグメントを少なくとも 1 つ作ってください。

パブリックとプライベートのロールバック・セグメントの選択

プライベート・ロールバック・セグメントは、インスタンスがデータベースをオープンするときに、インスタンスによって明示的に獲得されます。パブリック・ロールバック・セグメントは、ロールバック・セグメントを必要とする任意のインスタンスが使えるロールバック・セグメントのプールを形成します。

データベースに Parallel Server オプションがない場合には、パブリック・ロールバック・セグメントとプライベート・ロールバック・セグメントは同一です。したがって、すべてをパブリック・ロールバック・セグメントとして作れます。また、セグメント数が十分で、データベースをオープンする各インスタンスが、SYSTEM ロールバック・セグメントに加えて、ロールバック・セグメントを少なくとも 1 つ獲得できる限り、Parallel Server オプション付きのデータベースでも、パブリック・セグメントだけにできます。Oracle Parallel Server を使っている場合には、プライベート・ロールバック・セグメントも使えます。

関連項目：Parallel Server オプションとロールバック・セグメントの詳細は、『Oracle8 Parallel Server 概要および管理』を参照してください。

パブリック・ロールバック・セグメントとプライベート・ロールバック・セグメントの詳細は、『Oracle8 Server 概要』を参照してください。

自動的に獲得するロールバック・セグメントの指定

インスタンスが起動すると、デフォルトではインスタンスは TRANSACTIONS/TRANSACTIONS_PER_ROLLBACK_SEGMENT で算出される数のロールバック・セグメントを獲得します。特定のサイズまたは特定の表領域を持つ特定のロールバック・セグメントをインスタンスが獲得するようにしたい場合は、インスタンスのパラメータ・ファイル内の

パラメータ `ROLLBACK_SEGMENTS` に、そのロールバック・セグメントの名前を指定してください。

`TRANSACTIONS/TRANSACTIONS_PER_ROLL-BACK_SEGMENT` で算出される数よりも多くのロールバック・セグメントが指定されていても、インスタンスはこのパラメータ内にリストされたセグメントをすべて獲得します。ロールバック・セグメントは、プライベートでもパブリックでもかまいません。

ロールバック・セグメント・サイズの設定

ロールバック・セグメントの全体のサイズは、データベースに対して発行される最も一般的なトランザクションのサイズを基準にして設定すべきです。一般に、バッチ・ジョブなどの長時間実行のトランザクションでは、ロールバック・セグメントが大きいほどパフォーマンスは向上しますが、短いトランザクションでは、データベースに小さなロールバック・セグメントが数多く存在するときにパフォーマンスが向上します。多くの場合、ロールバック・セグメントはどのようなサイズのトランザクションであっても処理できます。ただし、トランザクションが非常に短かったり、長かったりするような極端な場合には、適切なサイズのロールバック・セグメントも使えます。

システムが短いトランザクションだけを実行する場合、ロールバック・セグメントは、主メモリ内に常時キャッシュされるように小さくすべきです。LRU アルゴリズムでは、ロールバック・セグメントが十分小さい場合に SGA 内にキャッシュされる可能性が大きくなり、必要なディスク I/O が少なくなるため、データベース・パフォーマンスが改善されます。小さいロールバック・セグメントの主な欠点は、他のトランザクションが頻繁に更新するレコードを必要とする、長い問合せを実行しているときに、「スナップショットが古すぎる」というエラーが起こる可能性が増大することです。このエラーは、他の更新エントリがロールバック・セグメントを折り返すと、読み込み一貫性を実現するのに必要なロールバック・エントリが上書きされるために発生します。アプリケーションのトランザクションを設計する際にはこのことを考慮して、トランザクションを短い最小作業単位にして、この問題が発生しないようにしてください。

対照的に、長時間実行のトランザクションのロールバック・エントリは、大きなロールバック・セグメントの事前に割り当てられたエクステントに適合できるので、長時間実行のトランザクションは、より大きなロールバック・セグメントとともに効率よく作動します。

データベース・システムのアプリケーションが、非常に短いトランザクションと非常に長いトランザクションを混合して同時に発行するとき、トランザクションが、トランザクション / ロールバック・セグメント・サイズを基準にしたロールバック・セグメントに明示的に割り当てられた場合に、パフォーマンスを最適化できます。つまり、ロールバック・セグメントに対する動的なエクステント割当てと切捨てを最小限に抑えることができます。これは、多くのシステムに対して必要なわけではなく、極端に大きい、または小さいトランザクションだけを対象にします。

非常に小さいトランザクションと非常に大きいトランザクションを混合して発行する際のパフォーマンスを最適化するには、各トランザクションのタイプ（小型、中型、大型など）に対するロールバック・セグメントを適切なサイズにしてください。ロールバック・セグメントのほとんどは、一般トランザクションに対応し、非一般トランザクション用のロールバック

ク・セグメントの数は少ないはずです。このように少ない方のロールバック・セグメントに OPTIMAL を設定することにより、セグメント・サイズは必要に応じて目的のレベルに戻ります。

ロールバック・セグメントは、トランザクションの種類によって使い分けなければならないことをユーザーに徹底させてください。多くの場合、トランザクションを特定のロールバック・セグメントに明示的に割り当てる利点はありません。ただし、不規則なトランザクションに対して作られた適切なロールバック・セグメントにそのトランザクションを割り当てることができます。たとえば、大きなバッチ・ジョブを含むトランザクションを、大きなロールバック・セグメントに割り当てることができます。

トランザクションの混合がそれほど多くなければ、ほとんどの SQL 文が表の 10% 以下にしか影響を及ぼさないため、それぞれのロールバック・セグメントは、データベースの最大の表のサイズの 10% にすべきです。したがって、このサイズのロールバック・セグメントは、多くの SQL 文によって実行されるアクションを格納するのに十分な大きさになるはずで

一般的には、ロールバック・セグメントに対して、大きな MAXEXTENTS を設定するとよいでしょう。これによって、ロールバック・セグメントは、必要なだけ、次のエクステントを割り当てることができます。

等しいサイズのエクステントを数多く持つロールバック・セグメントを作る

それぞれのロールバック・セグメントに割り当てられた全体の領域は、同じサイズの複数のエクステントに分ける必要があります。一般に、インスタンスに対する各ロールバック・セグメントに、等しいサイズのエクステントが 10 から 20 ある場合に、ロールバック I/O パフォーマンスは最適になります。

初期ロールバック・セグメントの合計のサイズとセグメントに対する初期エクステントの数を決定した後に、ロールバック・セグメントの各エクステントのサイズを算出するために、次の公式を使ってください。

$$T / n = s$$

ここで

T = 初期ロールバック・セグメントの合計のサイズ (単位はバイト)

n = 最初に割り当てられる初期エクステントの数

s = 最初に割り当てられる各エクステントの算出サイズ (単位はバイト)

s が計算された後、ロールバック・セグメントを作り、記憶領域パラメータ INITIAL と NEXT を s、MINEXTENTS を n に指定してください。PCTINCREASE はロールバック・セグメントに対しては指定できないので、0 にデフォルト設定されます。また、エクステントのサイズ s がデータ・ブロック・サイズの正確な倍数でない場合、次の倍数に切り上げられます。

各ロールバック・セグメントに対するエクステントの最適数の設定

それぞれロールバック・セグメントの OPTIMAL パラメータを設定するときには、システムが実行するトランザクションの種類を慎重に評価してください。長時間実行のトランザクションを頻繁に実行するシステムの場合、Oracle が頻繁に、エクステントの縮小と割当てを実行する必要がないように、OPTIMAL を大きくすべきです。アクティブ・データに対して長い問合せを実行するシステムでも、「スナップショットが古すぎる」という問題を避けるために OPTIMAL を大きくする必要があります。主に短いトランザクションと問合せを実行するシステムの場合、OPTIMAL をより小さく設定して、メモリー内にキャッシュされるようにロールバック・セグメントを十分小さくしてください。これにより、システムのパフォーマンスが向上します。

ロールバック・セグメントに対する OPTIMAL の値を推測し、その有効性を監視するには、Enterprise Manager/GUI の MONITOR ROLLBACK 機能を使います。このモニターでは、次のような統計情報がロールバック・セグメントごとに与えられます。

Size, High Water	ロールバック・セグメントに割り当てられた領域の最大値 (単位はバイト)
Sizes, Optimal	ロールバック・セグメントの OPTIMAL サイズ (単位はバイト)
Occurrences, Wraps	トランザクションがロールバック・セグメント内のあるエクステントから別の既存エクステントに書き込みを継続した累積回数
Occurrences, Extends	新しいエクステントがロールバック・セグメントに割り当てられた累積回数
Shrinks	Oracle がロールバック・セグメントからエクステントを切り捨てた累積回数
Average Sizes, Shrunk	Oracle がロールバック・セグメントから切り捨てた領域の平均サイズ (単位はバイト)
Average Sizes, Active	ロールバック・セグメント内のアクティブ・エクステントの平均バイト数 (時間計測)

インスタンスが、同等のサイズのエクステントを持つ同じサイズのロールバック・セグメントを持っていると想定すると、ロールバック・セグメントの OPTIMAL パラメータは、

Average Sizes, Active より少し大きく設定すべきです。表 18-1 に、このモニターに表示される統計表示の解釈の方法の詳細を示します。

表 18-1 現在の OPTIMAL 設定の有効性を分析する

縮小する回数	縮小される平均サイズ	分析と推奨事項
少ない	少ない	Average Sizes, active が Sizes、Optimal に近い場合、OPTIMAL の設定は正しい。そうでない場合は OPTIMAL が大きすぎる（縮小はあまり実行されていない）。
少ない	大きい	OPTIMAL は理想的に設定されている。
大きい	少ない	OPTIMAL が小さすぎる。縮小が必要以上に多く実行されている。
大きい	大きい	おそらく定期的な大規模トランザクションがこの統計の原因である。縮小の回数が少なくなるまで OPTIMAL パラメータの設定を大きくする。

ロールバック・セグメントに対する記憶位置の設定

SYSTEM 表領域内に必要とされる 2 つのロールバック・セグメントの他にロールバック・セグメントをすべて保持するように、表領域を 1 つ作ることをお勧めします。このようにすれば、すべてのロールバック・セグメント・データは、他のタイプのデータから分離して格納されます。このようなロールバック・セグメント表領域を作ることには、次の利点があります。

- ロールバック・セグメントを保持している表領域は常にオンライン状態にしておくことができるため、ロールバック・セグメントの合計記憶容量を常に最大限利用できます。ただし、ロールバック・セグメントの中に利用できないものがある場合には、データベース操作全体に影響が及ぶ可能性があります。
- アクティブ・ロールバック・セグメントを持つ表領域は、オフラインにできません。したがって、ある表領域がデータベースのすべてのロールバック・セグメントを保持するように設計することによって、他の表領域内に格納されたデータをデータベースのロールバック・セグメントとは無関係に、オフラインにできます。
- エクステンツの割当てと割当て解除が頻繁になされるロールバック・セグメントがある表領域では、使用可能エクステンツが断片化する可能性は高くなります。

ロールバック・セグメントの作成

ロールバック・セグメントを作るには、CREATE ROLLBACK SEGMENT システム権限を持っていなければなりません。データベースに追加のロールバック・セグメントを作るには、Enterprise Manager の「ロールバック・セグメント作成」プロパティ・シート、または SQL コマンド CREATE ROLLBACK SEGMENT を使います。新しいロールバック・セグメントが入る表領域はオンラインでなければなりません。

次の文は、USERS 表領域のデフォルトの記憶領域パラメータを使って、USERS 表領域内にパブリック・ロールバック・セグメント USERS_RS を作ります。

```
CREATE PUBLIC ROLLBACK SEGMENT users_rs TABLESPACE users;
```

新しいロールバック・セグメントをオンラインにする

新しいプライベート・ロールバック・セグメントを作る場合、そのロールバック・セグメントの名前をデータベースのパラメータ・ファイル内の ROLLBACK_SEGMENTS パラメータに追加してください。これにより、そのプライベート・ロールバック・セグメントはインスタンス起動時にインスタンスによって獲得されるようになります。たとえば、2 つの新しいプライベート・ロールバック・セグメント RS1 と RS2 が作られる場合、パラメータ・ファイルの ROLLBACK_SEGMENTS パラメータは、次のようになります。

```
ROLLBACK_SEGMENTS= (RS1, RS2)
```

関連項目：作られたロールバック・セグメントは、オンラインになるまで、どのインスタンスのトランザクションにも使えません。詳細は、18-10 ページの「ロールバック・セグメントのオンライン、オフライン」を参照してください。

ロールバック・セグメントの記憶領域パラメータを指定

ここでは、ロールバック・セグメントの記憶領域パラメータの設定について説明します。トピックは次のとおりです。

- ロールバック・セグメントを作成するときに記憶領域パラメータを設定する
- ロールバック・セグメントの記憶領域パラメータを変更する
- ロールバック・セグメントのフォーマットを変更する
- ロールバック・セグメントを手動で縮小する

ロールバック・セグメントを作成するときに記憶領域パラメータを設定する

次のような記憶領域パラメータと最適サイズを持つパブリック・ロールバック・セグメント DATA1_RS を作る場合を想定します。

- ロールバック・セグメントには、50KB の初期エクステントが割り当てられる。
- ロールバック・セグメントには、50KB の第 2 エクステントが割り当てられる。

- ロールバック・セグメントの最適サイズは 750KB である。
- 最小エクステント数、およびセグメントが作られるときに最初に割り当てられるエクステント数は 15 である。
- 初期エクステントを含めて、ロールバック・セグメントの割当て可能な最大エクステント数は 100 である。

次の文は、このような特性を持つロールバック・セグメントを作ります。

```
CREATE PUBLIC ROLLBACK SEGMENT data1_rs
    TABLESPACE users
    STORAGE (
        INITIAL 50K
        NEXT 50K
        OPTIMAL 750K
        MINEXTENTS 15
        MAXEXTENTS 100);
```

また、Enterprise Manager の「ロールバック・セグメント作成」プロパティ・シートを使ってロールバック・セグメントの記憶領域パラメータも設定できます。

ロールバック・セグメントの記憶領域パラメータを変更する

ロールバック・セグメントの記憶領域パラメータは、作成後に変更することもできます。ただし、ロールバック・セグメントに対して現在割り当てられているエクステントのサイズは変更できません。つまり、将来のエクステントだけに影響します。

Enterprise Manager の「ロールバック・セグメント変更」プロパティ・シート、または SQL コマンド ALTER ROLLBACK SEGMENT を使って、ロールバック・セグメントの記憶領域パラメータを変更します。

次の文を使って、DATA1_RS ロールバック・セグメントが割り当てることのできるエクステントの最大数を変更します。

```
ALTER PUBLIC ROLLBACK SEGMENT data1_rs
    STORAGE (MAXEXTENTS 120);
```

他のロールバック・セグメントの設定変更と同じように、OPTIMAL パラメータを含む、SYSTEM ロールバック・セグメントの設定を変更できます。

注意： パブリック・ロールバック・セグメントを変更する場合には、ALTER ROLLBACK SEGMENT コマンドに、キーワード PUBLIC を指定しなければなりません。

関連項目： ロールバック・セグメントのサイズと記憶領域パラメータ (OPTIMAL を含む) の設定に関するガイドラインについては、18-2 ページの「ロールバック・セグメントを管理するためのガイドライン」を参照してください。

ロールバック・セグメントのフォーマットを変更する

ロールバック・セグメントを変更するには、ALTER ROLLBACK SEGMENT システム権限が必要です。

ロールバック・セグメントについては、制限付きフォーマットまたは制限なしフォーマットを定義できます。制限付きまたは制限なしのフォーマットに変換するときは、そのロールバック・セグメントをオフラインにしなければなりません。ロールバック・セグメントのフォーマットが制限なしの場合は、そのセグメントのエクステントには最小限 4 つのデータ・ブロックがなければなりません。このため、データブロックが 4 つ未満のエクステントがある場合には、制限付きフォーマットのロールバック・セグメントを制限なしフォーマットに変換できません。制限付きフォーマットから制限なしフォーマットに変換し、エクステント内のデータ・ブロックを 4 未満にする場合は、そのロールバック・セグメントを削除して作りなおすしかありません。

ロールバック・セグメントを手動で縮小する

ロールバック・セグメントを縮小するには、ALTER ROLLBACK SEGMENT システム権限が必要です。

SQL コマンド ALTER ROLLBACK SEGMENT を使って、ロールバック・セグメントのサイズを手動で小さくできます。縮小させるロールバック・セグメントはオンラインになっていなければなりません。

次の文は、ロールバック・セグメント RBS1 を 100KB まで縮小します。

```
ALTER ROLLBACK SEGMENT rbs1 SHRINK TO 100K;
```

関連項目：ALTER ROLLBACK SEGMENT コマンドの詳細は、『Oracle8 Server SQL リファレンス』を参照してください。

ロールバック・セグメントのオンライン、オフライン

ここでは、ロールバック・セグメントをオンラインおよびオフラインにする方法について説明します。トピックは次のとおりです。

- ロールバック・セグメントをオンラインにする
- ロールバック・セグメントをオフラインにする

ロールバック・セグメントは、オンラインでトランザクションに利用できるか、またはオフラインでトランザクションに利用できないかのどちらかの状態にあります。ほとんどの場合、ロールバック・セグメントは、オンラインで、トランザクションによる使用に対して利用可能な状態にあります。

次のような状況では、オンライン・ロールバック・セグメントをオフラインにすると効果的です。

- 表領域をオフラインにしたいが、その表領域にロールバック・セグメントが入っている場合。トランザクションによって使われているロールバック・セグメントを、表領域が

含んでいる場合には、その表領域をオフラインにできない。対応するロールバック・セグメントが使われないようにするには、表領域をオフラインにする前にロールバック・セグメントをオフラインにします。

- トランザクションによって、現在ロールバック・セグメントが使われていると、そのロールバック・セグメントを削除できない。ロールバック・セグメントが使われないように、削除する前にロールバック・セグメントをオフラインにします。

注意： SYSTEM ロールバック・セグメントはオフラインにできません。

オフライン・ロールバック・セグメントを後からオンラインにして、トランザクションで使えます。作成直後、ロールバック・セグメントはオフラインになっています。新しく作ったロールバック・セグメントは、インスタンスのトランザクションで使う前に、明示的にオンラインにしなければなりません。そのロールバック・セグメントを含むデータベースにアクセスする任意のインスタンスを介して、オフラインのロールバック・セグメントをオンラインにできます。

ロールバック・セグメントをオンラインにする

現在の状態（DBA_ROLLBACK_SEGS データ・ディクショナリ・ビューに示される）が OFFLINE または PARTLY AVAILABLE であるロールバック・セグメントだけをオンラインにできます。オフラインのロールバック・セグメントをオンラインにするには、Enterprise Manager の「オンラインにする」メニュー項目または SQL コマンド ALTER ROLLBACK SEGMENT に ONLINE オプションを指定したものを使います。

PARTLY AVAILABLE ロールバック・セグメントをオンラインにする

PARTLY AVAILABLE 状態のロールバック・セグメントには、インダウトまたは回復した分散トランザクションのデータが入っています。その状態は、データ・ディクショナリ・ビュー DBA_ROLLBACK_SEGS では PARTLY AVAILABLE となっています。ロールバック・セグメントは、そのトランザクションが RECO によって自動的に、または DBA によって手動で解決されるまで、通常この状態のままです。しかし、すべてのロールバック・セグメントが PARTLY AVAILABLE になっている場合もあります。そのような場合には、前述の方法で PARTLY AVAILABLE セグメントをオンラインにできます。

ロールバック・セグメントがインダウト・トランザクションのために使ういくつかのリソースは、そのトランザクションが解決されるまで、アクセスできない状態になります。その結果、他のトランザクションが追加の領域を必要とする場合、ロールバック・セグメントは大きくする必要があるかもしれません。

PARTLY AVAILABLE ロールバック・セグメントをオンラインにするかわりに、インダウト・トランザクションが解決されるまで、一時的に新しいロールバック・セグメントを作る方が簡単なこともあります。

ロールバック・セグメントを自動的にオンラインにする

データベースを起動するたびにロールバック・セグメントを自動的にオンラインにする場合は、データベースのパラメータ・ファイルの ROLLBACK_SEGMENTS パラメータにそのセグメントの名前を追加します。

ロールバック・セグメントをオンラインにする例

次の例は、ロールバック・セグメント USER_RS_2 をオンラインにします。

```
ALTER ROLLBACK SEGMENT user_rs_2 ONLINE;
```

オンラインにした後、データ・ディクショナリ・ビュー DBA_ROLLBACK_SEGS におけるロールバック・セグメントの状態は ONLINE です。

関連項目：ROLLBACK_SEGMENTS パラメータと DBA_ROLLBACK_SEGS パラメータの詳細は、『Oracle8 Server リファレンス・マニュアル』を参照してください。

ロールバック・セグメントの状態をチェックする問合せについては、18-15 ページの「ロールバック・セグメント情報を表示する」を参照してください。

ロールバック・セグメントをオフラインにする

オンライン・ロールバック・セグメントをオフラインにするには、Enterprise Manager の「オフラインにする」メニュー項目または OFFLINE オプションを指定した ALTER ROLLBACK SEGMENT コマンドを使います。BA_ROLLBACK_SEGS のロールバック・セグメントの状態は "ONLINE" でなければならず、そのロールバック・セグメントは現行インスタンスが獲得しなければなりません。

たとえば、ロールバック・セグメント USER_RS_2 がオフラインになります。

```
ALTER ROLLBACK SEGMENT user_rs_2 OFFLINE;
```

アクティブ・ロールバック・セグメントを含まないロールバック・セグメントをオフラインにしようとすると、Oracle はただちにそのセグメントをオフラインにし、その状態を "OFFLINE" に変更します。

対照的に、アクティブ・トランザクション（ローカルまたはリモート、分散）のロールバック・データを含むロールバック・セグメントを使う場合、Oracle はロールバック・セグメントが将来のトランザクションで使われないようにし、そのロールバック・セグメントを使っているトランザクションがすべて完了すると、オフラインにします。トランザクションが完了するまで、オフラインにしようとしているインスタンス以外の、どのインスタンスもそのロールバック・セグメントをオンラインにできません。この間、ビュー DBA_ROLLBACK_SEGS 内のロールバック・セグメントの状態は、"ONLINE" になっています。しかし、ビュー V\$ROLLSTAT 内のロールバック・セグメントの状態は、PENDING OFFLINE になっています。

ロールバック・セグメントをオフラインにしようとして、PENDING OFFLINE を引き起こしたインスタンスは、いつでもそのロールバック・セグメントをオンラインに戻せます。ロールバック・セグメントは、オンラインに戻されても、通常どおり機能します。

パブリックとプライベートのロールバック・セグメントをオフラインにする

パブリックまたはプライベートのロールバック・セグメントをオフラインにすると、それを明示的にオンラインに戻すか、インスタンスを再起動するまで、オフライン状態のままです。

関連項目：ロールバック・セグメントの状態の表示については、18-15 ページの「ロールバック・セグメント情報を表示する」を参照してください。

ビュー DBA_ROLLBACK_SEGS と V\$ROLLSTAT の詳細は、『Oracle8 Server リファレンス・マニュアル』を参照してください。

ロールバック・セグメントにトランザクションを明示的に割り当て

トランザクションは、USE ROLLBACK SEGMENT パラメータ付きの SET TRANSACTION コマンドを使って、特定のロールバック・セグメントに対して明示的に割り当てられます。次のような目的で、トランザクションはロールバック・セグメントに明示的に割り当てられます。

- トランザクションによって生成されるロールバック情報の予想量を、割り当てられたロールバック・セグメントの現在のエクステントに合わせるできます。
- 付加的なエクステントをロールバック・セグメントに動的に割り当てる（そして切り捨てる）必要はありません。もしそのようにすると、システム全体のパフォーマンスが低下します。

トランザクションをロールバック・セグメントに明示的に割り当てるには、ロールバック・セグメントが現行のインスタンスに対してオンラインでなければなりません。さらに、SET TRANSACTION USE ROLLBACK SEGMENT 文はトランザクションの最初の文でなければなりません。指定したロールバック・セグメントがオンラインでない場合、または SET TRANSACTION USE ROLLBACK SEGMENT 文がトランザクション内の最初の文でない場合には、エラーが戻されます。

たとえば、大量（多くのトランザクションよりもさらに多く）の作業を含むトランザクションを開始する場合には、次の文で、そのトランザクションを大きなロールバック・セグメントに割り当てることができます。

```
SET TRANSACTION USE ROLLBACK SEGMENT large_rsl;
```

トランザクションがコミットされた後、ユーザーが新しいトランザクションを特定のロールバック・セグメントに明示的に割り当てない限り、Oracle は次のトランザクションを利用可能なロールバック・セグメントに自動的に割り当てます。

ロールバック・セグメントの削除

セグメントのエクステントがディスク上でかなり断片化した場合や、セグメントを異なる表領域に再割当てする必要があるときには、ロールバック・セグメントを削除できます。

ロールバック・セグメントを削除する前に、その状態が OFFLINE であることを確認してください。ロールバック・セグメントが、ONLINE、PARTLY AVAILABLE、NEEDS RECOVERY、INVALID になっている場合、そのロールバック・セグメントを削除できません。状態が INVALID の場合、そのセグメントはすでに削除されています。削除する前に、ロールバック・セグメントをオフラインにしなければなりません。

ロールバック・セグメントを削除するには、DROP ROLLBACK SEGMENT システム権限を持っていなければなりません。

ロールバック・セグメントがオフラインであれば、Enterprise Manager の「削除」メニュー項目、または SQL コマンド DROP ROLLBACK SEGMENT を使ってそのロールバック・セグメントを削除できます。

次の文は、DATA1_RS ロールバック・セグメントを削除します。

```
DROP PUBLIC ROLLBACK SEGMENT data1_rs;
```

DROP ROLLBACK SEGMENT コマンドを使う場合、削除するロールバック・セグメントのタイプ（パブリックまたはプライベート）を、PUBLIC キーワードの有無によって正しく指定しなければなりません。

注意： ROLLBACK_SEGMENTS に指定されたロールバック・セグメントを削除する場合、データベースのパラメータ・ファイルを編集して、ROLLBACK_SEGMENTS パラメータのリストから削除したロールバック・セグメントの名前を必ず削除してください。このステップが次のインスタンス起動の前に実行されていないと、削除されたロールバック・セグメントを獲得できないため、起動は失敗に終わります。

ロールバック・セグメントが削除されると、その状態は INVALID になります。ただし、次回、ロールバック・セグメントが作られると、利用できるようであれば、そのロールバック・セグメントは削除されたロールバック・セグメントによって空き状態になっていた行を使います。これによって、削除されたロールバック・セグメントの行が、DBA_ROLLBACK_SEGS ビューに示されることはありません。

関連項目： ビュー DBA_ROLLBACK_SEGS パラメータの詳細は、『Oracle8 Server リファレンス・マニュアル』を参照してください。

ロールバック・セグメント情報を監視

Enterprise Manager/GUI の MONITOR ROLLBACK 機能を使って、ロールバック・セグメントのサイズおよびエクステント数、最適なエクステント数、エクステントの動的な割当解除に関するアクティビティ、アクティブ・トランザクションによる現在の使用状況を監視できます。

関連項目：対応する操作で MONITOR を使う方法の詳細は、18-6 ページの「各ロールバック・セグメントに対するエクステントの最適数の設定」を参照してください。

ロールバック・セグメント情報を表示する

DBA_ROLLBACK_SEGS データ・ディクショナリ・ビューは、データベースのロールバック・セグメントに関する情報を格納します。たとえば、次の問合せは、データベース内の各ロールバック・セグメントの名前、および対応する表領域、状態をリストします。

```
SELECT segment_name, tablespace_name, status
       FROM sys.dba_rollback_segs;
```

SEGMENT_NAME	TABLESPACE_NAME	STATUS
SYSTEM	SYSTEM	ONLINE
PUBLIC_RS	SYSTEM	ONLINE
USERS_RS	USERS	ONLINE

さらに、次のデータ・ディクショナリ・ビューには、データベースのセグメントに関する情報が含まれています。

- USER_SEGMENTS
- DBA_SEGMENTS

すべてのロールバック・セグメントを表示する

次の問合せは、各ロールバック・セグメントの名前、各ロールバック・セグメントを含む表領域、各ロールバック・セグメントのサイズを戻します。

```
SELECT segment_name, tablespace_name, bytes, blocks, extents
       FROM sys.dba_segments
      WHERE segment_type = 'ROLLBACK';
```

SEGMENT_NAME	TABLESPACE_NAME	BYTES	BLOCKS	EXTENTS
RS1	SYSTEM	20480	10	2
RS2	TS1	40960	20	3
SYSTEM	SYSTEM	184320	90	3

ロールバック・セグメントがオフラインになったかどうかを表示する

ロールバック・セグメントをオフラインにしようとするとき、アクティブ・トランザクションがすべて完了するまで、実際にはオフラインになりません。オフラインにしようとしてから実際にオフラインになるまでの間、DBA_ROLLBACK_SEGS のその状態が ONLINE になっていても、新しいトランザクションでは使われません。インスタンスに対するロールバック・セグメントがこの状態になっているかどうかを判断するには、次の問合せを使ってください。

```
SELECT name, xacts 'ACTIVE TRANSACTIONS'
      FROM v$rollname, v$rollstat
 WHERE status = 'PENDING OFFLINE'
      AND v$rollname.usn = v$rollstat.usn;
```

NAME	ACTIVE TRANSACTIONS
-----	-----
RS2	3

インスタンスが Parallel Server 構成の一部である場合、この問合せは現行インスタンスだけのロールバック・セグメント情報を表示し、その他のインスタンスの情報は表示しません。

遅延ロールバック・セグメントを表示する

次の問合せは、どのロールバック・セグメントがプライベートで、どのセグメントがパブリックかを表示します。なお、現行インスタンスに対して現在オンラインになっているロールバック・セグメントに関する情報だけが表示されることに注意してください。

```
SELECT segment_name, tablespace_name, owner
      FROM sys.dba_rollback_segs;
```

SEGMENT_NAME	TABLESPACE_NAME	OWNER
-----	-----	-----
SYSTEM	SYSTEM	SYS
PUBLIC_RS	SYSTEM	PUBLIC
USERS_RS	USERS	SYS

遅延ロールバック・セグメントをすべて表示する

次の問合せは、すべての遅延ロールバック・セグメント（表領域がオンラインに戻されるまでオフラインにされた表領域のロールバック・エントリを保持するために作られたロールバック・セグメント）を表示します。

```
SELECT segment_name, segment_type, tablespace_name
      FROM sys.dba_segments
 WHERE segment_type = 'DEFERRED ROLLBACK';
```

SEGMENT_NAME	SEGMENT_TYPE	TABLESPACE_NAME
-----	-----	-----
USERS_RS	DEFERRED ROLLBACK	USERS<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2//EN">

セキュリティ方針の設定

この章では、セキュリティ方針を作るためのガイドラインについて、次の分野のデータベース操作ごとに解説します。

- システム・セキュリティ方針
- データ・セキュリティ方針
- ユーザー・セキュリティ方針
- パスワード管理方針
- 監査方針

システム・セキュリティ方針

ここでは、システム・セキュリティ方針について説明します。トピックは次のとおりです。

- データベース・ユーザー管理
- ユーザー認証
- オペレーティング・システムのセキュリティ

各データベースには、多岐にわたるセキュリティ方針のメンテナンスに責任を負う 1 人以上の管理者、つまりセキュリティ管理者が必要です。小規模なデータベース・システムの場合、データベース管理者がセキュリティ管理も兼務できます。しかし、大規模なデータベース・システムの場合、特別な人物またはグループがセキュリティ管理者に制限される責務を担当するとよいでしょう。

システムのセキュリティ管理担当者の決定後、あらゆるデータベースに対してセキュリティ方針を設定しなければなりません。データベースのセキュリティ方針には、次の部分で説明する、いくつかの副方針を設定しなければなりません。

データベース・ユーザー管理

データベース・ユーザーとは、Oracle データベースの情報へのアクセス・パスです。このため、データベース・ユーザーの管理に関しては厳密なセキュリティを保守しなければなりません。データベース・システムのサイズやデータベース・ユーザーの管理に必要な作業量に応じて、データベース管理者は、データベース・ユーザーの作成、変更、削除に必要な権限を持つ唯一のユーザーの場合があります。また、データベース・ユーザーを管理する権限を持つ管理者が多数存在する可能性もあります。どちらにしても、信頼のおける担当者の一人に、データベース・ユーザーの管理に必要な強力な権限を付与しなければなりません。

ユーザー認証

Oracle では、オペレーティング・システム、ネットワーク・サービス、データベースを使って、データベース・ユーザーを認証する（正しい人物であることを確認する）ことができます。多くの場合、ホスト・オペレーティング・システムを使ったユーザー認証の方が、次の理由により望ましいと言えるでしょう。

- ユーザー名やパスワードの指定なしに、ユーザーは Oracle に迅速かつ簡便に接続できる。
- オペレーティング・システムにおいてユーザー認証全体を集中的に管理する。このため、Oracle はオペレーティング・システムやデータベースに対応するユーザーのパスワードやユーザー名を格納または管理する必要はありません。
- データベースの監査証跡とオペレーティング・システムの監査証跡のユーザー・エンタリは一致する。

通常、データベースによるユーザー認証は、ホスト・オペレーティング・システムでユーザー認証がサポートできない場合に使われます。

関連項目：ネットワーク認証の詳細は、『Oracle8 Server 分散システム』を参照してください。

ユーザー認証の詳細は、20-10 ページの「ユーザーを作成する」を参照してください。

オペレーティング・システムのセキュリティ

状況に応じて、Oracle とその他のデータベース・アプリケーションが稼働するオペレーティング・システム環境では、次のようなセキュリティを考慮する必要があります。

- データベース管理者は、ファイルを作成および削除するためのオペレーティング・システム権限を必ず持っていなければならない。
- 通常のデータベース・ユーザーは、データベースに関連付けられたファイルを作成および削除するためのオペレーティング・システム権限を有してはならない。
- オペレーティング・システムを使ってユーザーのデータベース・ロールを識別する場合、セキュリティ管理者は、オペレーティング・システム・アカウントのセキュリティ定義域を修正するためのオペレーティング・システム権限を持っていなければならない。

関連項目：Oracle データベースに関するオペレーティング・システムのセキュリティの問題点の詳細は、オペレーティング・システムに固有の Oracle のマニュアルを参照してください。

データ・セキュリティ方針

データ・セキュリティでは、オブジェクト・レベルでのデータベースへのアクセスおよびデータベースの使用を制御するメカニズムを採用しています。データ・セキュリティ方針によって、特定のスキーマ・オブジェクトへのアクセス権限を所有するユーザーと、そのオブジェクトに対して許可される、ユーザーごとの特定のアクションのタイプが決まります。たとえば、ユーザー SCOTT は、EMP 表を使って SELECT 文および INSERT 文を発行できますが、DELETE 文は発行できません。また、データ・セキュリティ方針では、スキーマ・オブジェクトごとに監査が必要なアクション（ある場合）を定義しなければなりません。

データ・セキュリティ方針は、主にデータベースのデータに対して設定するセキュリティのレベルによって決まります。たとえば、任意のユーザーが任意のスキーマ・オブジェクトを作れるようにする場合、またはユーザーにシステム内の他のユーザーのオブジェクトへのアクセス権限を付与する場合は、データベースのデータ・セキュリティは厳密に設定しなくてもかまいません。一方、データベース管理者またはセキュリティ管理者だけがオブジェクトを作ったり、オブジェクトのアクセス権限をロールおよびユーザーへ付与できるようにしたりするには、データ・セキュリティの管理を強化する必要があります。

データ・セキュリティ全般はデータの機密性に基づいています。機密性の低い情報の場合には、データ・セキュリティ方針は厳しく設定してなくてもかまいません。しかし、機密性の高いデータの場合には、セキュリティ方針を慎重に設定し、オブジェクトに対するアクセスを厳しく制御しなければなりません。

ユーザー・セキュリティ方針

ここでは、ユーザー・セキュリティ方針について説明します。トピックは次のとおりです。

- 一般的なユーザー・セキュリティ
- エンド・ユーザーのセキュリティ
- 管理者のセキュリティ
- アプリケーションの開発者のセキュリティ
- アプリケーション管理者のセキュリティ

一般的なユーザー・セキュリティ

すべてのタイプのデータベース・ユーザーについて、次の一般的なユーザー・セキュリティ事項を考えてみます。

- パスワード・セキュリティ
- 権限管理

パスワード・セキュリティ

ユーザー認証をデータベースで管理している場合、データベース・アクセス・セキュリティを保守するために、パスワード・セキュリティ方針を設定しなければなりません。たとえば、各データベース・ユーザーは、一定の間隔を置いて、およびパスワードが別のユーザーに漏れた場合にパスワードを変更する必要があります。このような場合に、ユーザーに強制的にパスワードを修正するように要求すれば、許可されていないデータベース・アクセスの可能性を減らせます。

暗号化されたパスワードによる安全な接続

パスワードの機密性の保護を強化するため、クライアント / サーバーまたはサーバー / サーバーの接続に暗号化されたパスワードを使うよう Oracle を構成できます。

次の値を設定すると、接続の確認に使うパスワードを常に暗号化できます。

- クライアント・マシンの `ORA_ENCRYPT_LOGIN` 環境変数を `TRUE` に設定する。
- サーバーの `DBLINK_ENCRYPT_LOGIN` 初期化パラメータを `TRUE` に設定する。

パスワードはクライアントとサーバーの両方で使用可能になっても、ネットワーク内を「そのままの形」で送信されるのではなく、修正済み DES（データ暗号化規格）アルゴリズムを使って暗号化されます。

`DBLINK_ENCRYPT_LOGIN` パラメータは 2 つの Oracle Server を接続するときに使われます（たとえば、分散問合せを実行する場合など）。クライアントから接続する場合は、`ORA_ENCRYPT_LOGIN` 環境変数がチェックされます。

パスワードを使ってサーバーに接続しようとする、そのパスワードはサーバーに送信される前に必ず暗号化されます。接続が失敗して監査が使用可能になっていた場合、監査ログにその失敗が記録されます。次に、適切な DBLINK_ENCRYPT_LOGIN 値または ORA_ENCRYPT_LOGIN 値がチェックされます。この値が FALSE に設定されていると、Oracle は暗号化されていないパスワードを使って再度接続を試みます。接続が成功すると、監査ログに以前記録された失敗が接続にかわり、接続が維持されます。不正なユーザーが暗号化されていないパスワードを使って、Oracle に接続を強制的に再試行させることがないように、適切な（パラメータの）値を TRUE に設定しておいてください。

権限管理

セキュリティ管理者は、すべてのタイプのユーザーについて権限管理に関することを考慮する必要があります。たとえば、多数のユーザー名を扱うデータベースで、ユーザーが使える権限を管理するには、ロール（ユーザーまたは他のロールに付与する、関連する権限のグループ）を使うと便利な場合があります。ただし、少数のユーザー名を扱うデータベースにおいては、ユーザー名に権限を明示的に付与し、ロールは使わない方が簡単です。

セキュリティ管理者は、多数のユーザーまたはアプリケーション、オブジェクトを扱うデータベースを管理する場合、ロールの特長を利用する必要があります。ロールは非常に複雑な環境における権限管理のタスクを簡易にします。

エンド・ユーザーのセキュリティ

セキュリティ管理者は、エンド・ユーザーのセキュリティの方針も定義する必要があります。多数のユーザーがアクセスする大規模なデータベースの場合、セキュリティ管理者は、ユーザー・グループのカテゴリの決定およびそのユーザー・グループに対するユーザー・ロールの作成、各ユーザー・ロールに必要な権限またはアプリケーション・ロールの付与、ユーザーへのユーザー・ロールの割当てができます。例外を考慮して、セキュリティ管理者は必ず個々のユーザーに対して明示的に付与しなければならない権限もあわせて決定しなければなりません。

エンド・ユーザー権限管理のためにロールを使う

ロールは、データベース・ユーザーの異なるグループで必要となる共通の権限をまとめて付与および管理する最も簡単な方法です。

ある会社の経理部門のユーザー全員がデータベース・アプリケーション ACCTS_RECEIVABLE および ACCTS_PAYABLE を実行するための権限を必要としている状況を考えてみます。ロールは上記の両アプリケーションに対応付けられ、それらのアプリケーションを実行するために必要なオブジェクト権限を含んでいます。

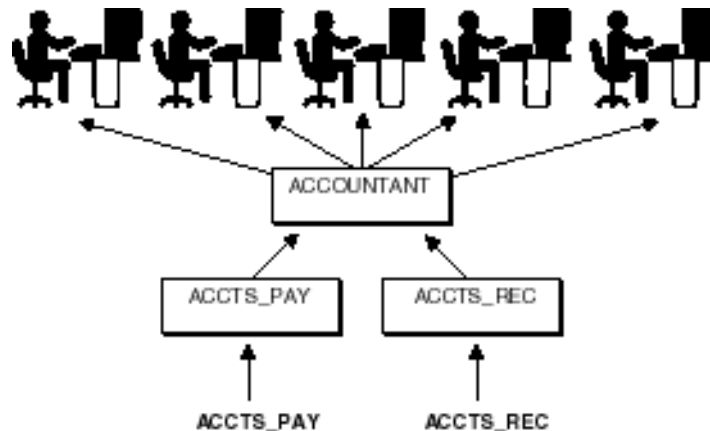
このような単純なセキュリティを必要とする状況に対処するには、データベース管理者またはセキュリティ管理者は次のアクションを実施します。

1. ACCOUNTANT という名前のロールを作る。
2. データベース・アプリケーション ACCTS_RECEIVABLE および ACCTS_PAYABLE のロールを ACCOUNTANT ロールに付与する。

3. ACCOUNTANT ロールを経理部門の各ユーザーに付与する。

このセキュリティ・モデルを図 19-1 に示します。

図 19-1 ユーザーのロール



このモデルは、次のような状況に対処します。

- その後に経理部門が新しいデータベース・アプリケーションに対してロールを必要とする場合には、そのアプリケーションのロールが ACCOUNTANT ロールに付与され、その結果、経理部門のすべてのユーザーに新しいデータベース・アプリケーションの権限が付与される。毎回、アプリケーションを使う必要がある個々のユーザーに、アプリケーションのロールを付与する必要はありません。
- 同様に、その経理部門が特定のアプリケーションを必要としなくなった場合も、そのアプリケーションのロールを ACCOUNTANT ロールから削除できる。
- ACCTS_RECEIVABLE または ACCTS_PAYABLE の各アプリケーションで必要な権限が変更される場合、新しい権限をそのアプリケーションのロールに付与またはロールから削除できる。ACCOUNTANT ロールのセキュリティ定義域や ACCOUNTANT ロールを付与されたすべてのユーザーにその権限の修正が自動的に反映されます。
- このように、ユーザーに直接割り当てるロールは 1 つだけで済みます。

できれば、すべての状況でロールを使って、エンド・ユーザーの権限管理を効率化し、かつ簡素化してください。

管理者のセキュリティ

セキュリティ管理者には、管理者のセキュリティに対処する方針が必要です。たとえば、データベースが大規模で各種のデータベース管理者が存在している場合、セキュリティ管理者は、関連のある管理権限をいくつかの管理ロールに分類することに決定するとよいでしょう。これにより、管理ロールを適切な管理者ユーザーに付与できます。ただし、少量のデータベースでさらに管理者が数人しかいない場合、1つの管理ロールを作り、その権限を管理者全員に付与するのが便利です。

SYS や SYSTEM としての接続に対する保護

データベースの作成後、ただちに管理用の SYS ユーザー名と SYSTEM ユーザー名のパスワードを変更して、データベースに対する無許可のアクセスを防止してください。SYS および SYSTEM として接続すると、多数の方法でデータベースを変更する強力な権限がユーザーに与えられます。したがって、これらのユーザー名の権限は非常に機密性があり、データベース管理者を選定するときだけ使用すべきです。

関連項目：これらのアカウントに対応するパスワードは、20-14 ページの「ユーザーを変更する」で説明されている手順に従って変更できます。

管理者の接続に対する保護

管理者権限を使ってデータベースに接続できるのは、データベース管理者だけでなければなりません。SYSDBA として接続すると、データベースに関する処理（起動、停止、回復など）またはデータベース内のオブジェクトに関する処理（作成、削除など）を実行するための無制限な権限がユーザーに付与されます。

管理者権限管理のためにロールを使う

ロールは、データベースの管理者に必要な、強力なシステム権限やロールに制限を与える最も簡単な方法です。

大規模なインストール作業におけるデータベース管理者の責任を、何人かのデータベース管理者で分担し、各管理者がそれぞれ次のような特定のデータベース管理業務を担当するシナリオを考えてみます。

- オブジェクトを作成することと保守を担当する管理者
- データベースの調整とパフォーマンスを担当する管理者
- 新規ユーザーを作成すること、データベース・ユーザーに対するロールや権限の付与を担当するシステム管理者
- ルーチン・データベース操作（たとえば、起動、停止、バックアップ）を担当するデータベース管理者
- データベース回復などの緊急事態を担当する管理者
- データベース管理の経験がない新任のデータベース管理者で権限を制限する必要がある者

このシナリオでは、セキュリティ管理者は管理者に対して次のようなセキュリティを組織的に構成する必要があります。

1. 6つのロールを定義して、それぞれのタイプの業務を遂行するために必要な権限をその中に明確に盛り込まなければなりません（たとえば、DBA_OBJECT、DBA_TUNE、DBA_SECURITY、DBA_MAINTAIN、DBA_RECOV、DBA_NEW）。
2. 各ロールには適切な権限を付与します。
3. 各データベース管理者ごとに対応するロールを付与できます。

この計画では、次のような方法で今後問題が発生する可能性が抑えられます。

- データベース管理者のジョブの記述を変更して、さらに責任担当部分を付加する場合、そのデータベース管理者には新しい担当に対応する他の管理ロールを付与できる。
- データベース管理者のジョブの記述からその責任担当部分を減らすような変更をする場合、そのデータベース管理者は適切な管理ロールを取り消せる。
- データ・ディクショナリは必ず各ロールと各ユーザーに関する情報を格納しているので、こうした情報によって各管理者のタスクを明らかにする目的で適用できる。

アプリケーションの開発者のセキュリティ

セキュリティ管理者は、データベースを使うアプリケーション開発者に対して特殊なセキュリティ方針を定義しなければなりません。セキュリティ管理者は、アプリケーション開発者に、必要なオブジェクトを作るための権限を付与することがあります。そのかわりに、オブジェクトを作るための権限をデータベース管理者だけに付与することもできます。

アプリケーション開発者とその権限

データベース・アプリケーション開発者は、開発作業を遂行するために特殊な権限グループを要求する独特なデータベース・ユーザーです。エンド・ユーザーとは異なり、開発者はCREATE TABLE、CREATE PROCEDUREなどの各種システム権限を必要とします。ただし、データベースにおける機能全般を制限するために、開発者には唯一の特定システム権限を付与しなければなりません。

アプリケーション開発者の環境：テストと本番データベース

多くの場合、アプリケーション開発者は、データベースをテストすることだけに制限されており、本番データベースに関して許可されていません。この制限によって、アプリケーション開発者はエンド・ユーザーと競合せずにデータベース・リソースを獲得できるため、本番データベースに悪影響を及ぼすことはありません。

アプリケーションを開発し徹底的にテストをした時に、そのアプリケーションは本番データベースにアクセスすることが許可され、その本番データベースの適切なエンド・ユーザーに適用します。

無制限および制限付きのアプリケーション開発

データベース管理者はアプリケーション開発者に対して付与すべき権限を決定する際に、次のオプションを定義できます。

無制限の開発	アプリケーション開発者は、表、索引、プロシージャ、パッケージなど、新しいスキーマ・オブジェクトを作ることが許可されています。このオプションを使えば、このアプリケーションが他のオブジェクトとは独立したアプリケーションを開発できます。
制限付きの開発	アプリケーション開発者は、新しいスキーマ・オブジェクトを作ることが許可されていません。必要な表、索引、プロシージャなどすべて、アプリケーション開発者の要求どおりに、データベース管理者が作ります。このオプションを使えば、データベース管理者がデータベースの領域の用途やデータベース内の情報のアクセス経路を完全に制御できます。

データベース・システムによっては、この2つのオプションのうち1つだけしか使わないものもありますが、2つのオプションを混用できるシステムもあります。たとえば、アプリケーション開発者は新しいストアド・プロシージャやパッケージの作成を許可して、表や索引の作成を許可しないようにもできます。セキュリティ管理者は、次の状況に応じてこの事項に関して決定します。

- データベースの領域の使用全般に必要な制御
- スキーマ・オブジェクトに対するアクセス経路全般に必要な制御
- アプリケーションを開発するために適用するデータベース。テスト・データベースをアプリケーション開発に適用する場合、より自由な開発方針が望ましいと言えます。

アプリケーション開発者のためのロールと権限

セキュリティ管理者は、通常のアプリケーション開発者が必要とする権限を管理するためのロールを作れます。たとえば、APPLICATION_DEVELOPER という名前の付いた通常のロールには、CREATE TABLE、CREATE VIEW、CREATE PROCEDURE などの各システム権限を指定できます。アプリケーション開発者のロールを定義する場合は、次の点に考慮してください。

- CREATE システム権限は、通常、アプリケーション開発者に権限付与されるので、開発者独自のオブジェクトを作れる。ただし、CREATE ANY システム権限は、任意のユーザーの定義域でのオブジェクトの作成を許可するためのものなので、通常、開発者には付与されません。この制限によって、その開発者のユーザー・アカウントだけが、新しいオブジェクトを作れます。
- オブジェクト権限はアプリケーション開発者が使うロールに付与されることはほとんどない。ただし、ロールを介してオブジェクト権限を付与すると、他のオブジェクト（主

として、ビュー、ストアド・プロシージャ）を作るときの有用性が制限されるので、これはあまり実用的ではありません。むしろ、アプリケーション開発者が開発の目的で独自のオブジェクトを作れるように許可することの方が実用的となります。

アプリケーション開発者に課せられる領域の制限

一般的に、アプリケーション開発者は開発プロセスの一部としてオブジェクトを作る権限が付与されているのに対して、セキュリティ管理者は必ず各アプリケーション開発者で使用可能なデータベースの領域の限定とサイズの制限を保守しなければなりません。たとえば、セキュリティ管理者として、個々のアプリケーション開発者ごとに次の制限を設定してください。

- 開発者が表または索引を作れる表領域
- 開発者にとってアクセス可能な各表領域としての制限割当て

関連項目：どちらの制限も開発者のセキュリティ定義域を変更することで設定できます。詳細は、20-14 ページの「ユーザーを変更する」を参照してください。

アプリケーション管理者のセキュリティ

大量のデータベース・アプリケーション（たとえば、プリコンパイラ、Forms アプリケーションなど）を使う大規模データベース・システムにおいては、アプリケーション管理者が必要となります。アプリケーション管理者は次のタイプの業務に責任があります。

- アプリケーションに対するロールの作成と各アプリケーション・ロールの権限の管理
- データベース・アプリケーションによって使われるオブジェクトの作成と管理
- 必要に応じて、アプリケーション・コードや Oracle プロシージャの保守と更新

多くの場合、アプリケーション管理者は、アプリケーションを設計したアプリケーション開発者でもあります。ただし、これらのジョブを必ずしも開発担当者に任せる必要はありません。データベース・アプリケーションに詳しい別の担当者に指名することもできます。

パスワード管理方針

データベース・セキュリティ・システムは、どんな時にもパスワードが機密になっていることに依存しています。それでもパスワードは、盗難、偽造、悪用などには弱い場合があります。データベース・セキュリティの制御をさらに強化するために、Oracle のパスワード管理方針が DBA によって制御されます。

ここでは、Oracle のパスワード管理の次の点を説明します。

- アカウント・ロック
- パスワード・エイジングおよび時間切れ
- パスワード履歴
- パスワードの複雑さの検証

アカウント・ロック

特定のユーザーが、指定された回数以上ログインを失敗した場合、サーバーはそのユーザーのアカウントを自動的にロックします。DBA は、CREATE PROFILE 文を使って、ログインの失敗が許可される回数を指定します。また、DBA はアカウントがロックされる時間の長さも指定します。

次の例では、ユーザー ASHWINI に許可されているログイン失敗の最大回数は 4 で、アカウントがロックされる時間の長さは 30 日です。アカウントは、30 日がすぎると自動的にロック解除されます。

```
CREATE PROFILE prof LIMIT
  FAILED_LOGIN_ATTEMPTS 4
  ACCOUNT_LOCK_TIME 1/24;
ALTER USER ashwini PROFILE prof;
```

DBA がアカウントのロック解除に関して時間間隔を指定しないなら、ACCOUNT_LOCK_TIME はデフォルト値になります。DBA が ACCOUNT_LOCK_TIME を UNLIMITED と指定した場合、システム・セキュリティ管理担当者は明示的にそのアカウントのロックを解除する必要があります。したがって、アカウントがロックされている時間の長さは、ユーザーに割り当てられているリソース・プロファイルが DBA が構成する方法によって違います。

ユーザーが正常にアカウントにログインできると、そのユーザーが失敗したログインのカウントは、ゼロにリセットされます。

セキュリティ管理担当者は、明示的にユーザー・アカウントをロックすることもできます。そのようにした場合、アカウントは自動的にロック解除されません。セキュリティ管理担当者がアカウントのロックを解除することが必要です。

関連項目：CREATE PROFILE 文の詳細は、『Oracle8 Server SQL リファレンス』を参照してください。

パスワード・エイジングおよび時間切れ

DBA は、CREATE PROFILE 文を使ってパスワードの最大存続期間を指定します。指定された時間が経過してパスワードの期限が切れると、ユーザーまたは DBA はパスワードを変更しなければなりません。次の文の場合、ASHWINI は、同じパスワードを 90 日間使うことができ、90 日間経過すると期限が切れます。

```
CREATE PROFILE prof LIMIT
  FAILED_LOGIN_ATTEMPTS 4
  ACCOUNT_LOCK_TIME 30
  PASSWORD_LIFE_TIME 90;
ALTER USER ashwini PROFILE prof;
```

DBA は、CREATE PROFILE 文を使って、猶予期間を指定することもできます。ユーザーは、パスワードの期限が切れた後、データベース・アカウントに最初にログインしようとするとき、猶予期間に入ります。猶予期間中は、ユーザーがアカウントにログインしようとするたびに警告メッセージが表示され、猶予期間が終わるまで表示され続けます。ユーザーは、猶予期間内にパスワードを変更しなければなりません。パスワードが猶予期間内に変更されないなら、アカウントの期限が切れ、パスワードが変更されるまでそのアカウントにはログインできません。図 19-2 は、パスワードの存続期間と猶予期間の記録を示したものです。

図 19-2 パスワードの存続期間と猶予期間の記録



たとえば、パスワードの存続期間が 60 日間で、猶予期間が 3 日間だとします。ユーザーが 60 日目 (70 日目、100 日目などでもよく、要はそれがパスワードの存続期間以降の試行の最初のログインであるということである) 以降の任意の日にログインを試行する場合、パスワードが 3 日以内に期限切れになることを示す警告メッセージがそのユーザーに表示されます。ユーザーが猶予期間の 1 ~ 3 日の間にパスワードを変更しないなら、そのユーザーのアカウントの期限が切れます。次の文は、ユーザーがパスワードを期限切れ 3 日以内に変更する必要があることを指定しています。

```
CREATE PROFILE prof LIMIT
  FAILED_LOGIN_ATTEMPTS 4
  ACCOUNT_LOCK_TIME 30
  PASSWORD_GRACE_TIME 3;
ALTER USER ashwini PROFILE prof;
```

セキュリティ管理担当者は、明示的にアカウントを期限切れにすることができます。これは、特に新しいアカウントの場合に役立ちます。

関連項目 : CREATE PROFILE 文の詳細は、『Oracle8 Server SQL リファレンス』を参照してください。

パスワード履歴

DBA は、CREATE PROFILE 文を使って、ユーザーがパスワードを再使用できない時間間隔を指定します。

次の文では、DBA はユーザーが現在の自分のパスワードを 60 日間再使用できないことを指定しています。

```
CREATE PROFILE prof LIMIT
PASSWORD_REUSE_TIME 60
PASSWORD_REUSE_MAX UNLIMITED;
```

次の文は、ユーザーが現在のパスワードを再使用できるようになるまでに、パスワードを最低 3 回変更しなければならないことを指定しています。

```
CREATE PROFILE prof LIMIT
PASSWORD_REUSE_MAX 3
PASSWORD_REUSE_TIME UNLIMITED;
```

注意 : PASSWORD_REUSE_TIME または PASSWORD_REUSE_MAX のどちらか一方を指定し、両方同時には使わないでください。

パスワードの複雑さの検証

Oracle のパスワードの複雑さの検証ルーチンは、デフォルトのプロファイル・パラメータを設定する PL/SQL スクリプト (utlpwdmg.sql) を使って指定できます。

パスワードの複雑さの検証ルーチンは、次のことを調べます。

- パスワードの長さが 4 以上である。
- パスワードがユーザー ID と同じではない。
- パスワードに少なくとも 1 つのアルファベット文字、1 つの数字、1 つの句読点が含まれている。
- パスワードが、welcome、account、database、user などの簡単な単語ではない。
- 以前のパスワードとの違いが 3 文字以上ある。

注意 : Oracle では、ALTER USER 文ではパスワード検証機能を十分サポートしないので、その文でパスワードを変更しないように勧めています。そのかわり、SQL*Plus や Enterprise Manager のような Oracle が提供しているツールを使用してパスワードを変更してください。

パスワード検証ルーチンのフォーマットに関するガイドライン

DBA は、パスワードの既存の複雑さ検証ルーチンを拡張したり、PL/SQL またはサード・パーティ製のツールを使って独自のパスワード検証ルーチンを作れます。

DBA が作った PL/SQL コールは、次のフォーマットになっていなければなりません。

```
routine_name (  
  userid_parameter IN VARCHAR(30),  
  password_parameter IN VARCHAR (30),  
  old_password_parameter IN VARCHAR (30)  
)  
RETURN BOOLEAN
```

新しいルーチンを作ったなら、ユーザーのプロファイルまたはシステムのデフォルト・プロファイルを使って、それをパスワード検証ルーチンとして割り当てる必要があります。

```
CREATE/ALTER PROFILE profile_name LIMIT  
PASSWORD_VERIFY_FUNCTION routine_name
```

パスワード検証ルーチンは、SYS が所有していなければなりません。

パスワード検証ルーチン：サンプル・スクリプト 次のサンプル・スクリプトは、デフォルトのパスワード・リソースの制限を設定し、パスワードの複雑さについて最小のチェックを行います。新しいパスワードに対する独自の複雑さのチェックを開発する際は、このサンプル・スクリプトをモデルとして使えます。

このスクリプトは、デフォルトのパスワード・リソース・パラメータを設定しますが、パスワード機能を使用可能にするときには必ず実行してください。ただし、必要に応じてデフォルトのリソース・パラメータを変更できます。

デフォルトのパスワード複雑さ機能は、次の最小複雑さチェックを実行します。

- このパスワードは、最小の長さの条件を満たします。
- パスワードがユーザー名でないことを確認します。必要に応じて、この機能を変更できます。

この機能は SYS スキーマ内で作成しますが、スクリプトを実行する前に sys/<password> を sysdba として接続する必要があります。

```
CREATE OR REPLACE FUNCTION verify_function  
(username varchar2,  
  password varchar2,  
  old_password varchar2)  
RETURN boolean IS  
  n boolean;  
  m integer;  
  differ integer;  
  isdigit boolean;  
  ischar boolean;
```

```

ispunct boolean;
digitarray varchar2(20);
punctarray varchar2(25);
chararray varchar2(52);

BEGIN
    digitarray:= '0123456789';
    chararray:= 'abcdefghijklmnopqrstuvmxyzABCDEFGHJKLMNOPQRSTUVWXYZ';
    punctarray:= '!"#$%&'()*' '+, - / : ; < = > ? _ ' ;

    --Check if the password is same as the username
    IF password = username THEN
        raise_application_error(-20001, 'Password same as user');
    END IF;

    --Check for the minimum length of the password
    IF length(password) < 4 THEN
        raise_application_error(-20002, 'Password length less than 4');
    END IF;

    --Check if the password is too simple. A dictionary of words may be
    --maintained and a check may be made so as not to allow the words
    --that are too simple for the password.
    IF NLS_LOWER(password) IN ('welcome', 'database', 'account', 'user', 'password', 'oracle',
    'computer', 'abcd') THEN raise_application_error(-20002, 'Password too simple');
    END IF;

    --Check if the password contains at least one letter, one digit and one
    --punctuation mark.
    --1. Check for the digit
    --You may delete 1. and replace with 2. or 3.
    isdigit:=FALSE;
    m := length(password);
    FOR i IN 1..10 LOOP
        FOR j IN 1..m LOOP
            IF substr(password,j,1) = substr(digitarray,i,1) THEN
                isdigit:=TRUE;
                GOTO findchar;
            END IF;
        END LOOP;
    END LOOP;
    IF isdigit = FALSE THEN
        raise_application_error(-20003, 'Password should contain at least one
    digit, one character and one punctuation');
    END IF;

    --2. Check for the character
    <<findchar>>
    ischar:=FALSE;
    FOR i IN 1..length(chararray) LOOP

```

```
FOR j IN 1..m LOOP
    IF substr(password,j,1) = substr(chararray,i,1) THEN
        ischar:=TRUE;
        GOTO findpunct;
    END IF;
END LOOP;
END LOOP;
IF ischar = FALSE THEN
    raise_application_error(-20003, 'Password should contain at least one digit, one
character and one punctuation');
END IF;
--3. Check for the punctuation
<<findpunct>>
ispunct:=FALSE;
FOR i IN 1..length(punctarray) LOOP
    FOR j IN 1..m LOOP
        IF substr(password,j,1) = substr(punctarray,i,1) THEN
            ispunct:=TRUE;
            GOTO endsearch;
        END IF;
    END LOOP;
END LOOP;
IF ispunct = FALSE THEN raise_application_error(-20003, 'Password should contain at least
one ¥ digit, one character and one punctuation');
END IF;

<<endsearch>>

--Check if the password differs from the previous password by at least 3 letters
IF old_password = '' THEN
    raise_application_error(-20004, 'Old password is null');
END IF;
--Everything is fine; return TRUE ;
differ := length(old_password) - length(password);

IF abs(differ) < 3 THEN
    IF length(password) < length(old_password) THEN
        m := length(password);
    ELSE
        m:= length(old_password);
    END IF;
    differ := abs(differ);
    FOR i IN 1..m LOOP
        IF substr(password,i,1) != substr(old_password,i,1) THEN
            differ := differ + 1;
        END IF;
    END LOOP;
    IF differ < 3 THEN
        raise_application_error(-20004, 'Password should differ by at ¥
least 3 characters');
```

```
        END IF;  
    END IF;  
    --Everything is fine; return TRUE ;  
    RETURN(TRUE);  
END;
```

監査方針

セキュリティ管理者は、各データベースの監査プロシージャの方針を定義する必要があります。たとえば、なんらかの疑いのあるアクティビティが存在しているとは考えられない場合は、データベース監査の使用禁止を決定してもかまいません。監査が必要な場合、セキュリティ管理者はデータベースの監査のレベルを詳細に決定する必要があります。通常、一般的なシステム監査では、疑いのあるアクティビティを特定した後、もっと細かい監査を実行します。

ユーザーとリソースの管理

この章では、Oracle データベースへのアクセスの制御方法を説明します。トピックは次のとおりです。

- セッションとユーザーのライセンス
- ユーザー認証
- Oracle ユーザー
- プロファイルでリソースを管理
- データベース・ユーザーとプロファイルに関する情報のリスト

関連項目：ユーザーおよびプロファイルのセキュリティ方針の設定に関するガイドラインは、第 19 章「セキュリティ方針の設定」を参照してください。

権限およびロールは、ユーザーのデータベースに対するアクセスやデータベース中のスキーマ・オブジェクトに対するアクセスを制御します。権限とロールの詳細は、第 21 章「ユーザー権限とロールの管理」を参照してください。

この章には、Oracle Enterprise Manager について述べている箇所がいくつかあります。Enterprise Manager/GUI または Server Manager/LineMode を使って特定のタスクを実行する方法の詳細は、『Oracle Server Manager ユーザーズ・ガイド』または『Oracle Enterprise Manager 管理者ガイド』を参照してください。

セッションとユーザーのライセンス

ここでは、セッションおよびユーザー・ライセンスについて説明します。トピックは次のとおりです。

- 同時使用ライセンス
- 接続権限
- セッションの最大数を設定する
- セッション警告制限を設定する
- データベースの稼働中に同時使用制限を変更する
- 名前付きユーザー制限
- ライセンス制限と現行値を参照する

Oracle は、サイトが Oracle Server のライセンス契約に従っていることを保証する手助けをします。サイトが同時ユーザーのライセンスを受けている場合、データベースに同時に接続するセッション数を追跡し、制限できます。サイトが端末ユーザーによってライセンスされている場合、データベース内に作成できる端末ユーザー数を制限できます。どちらの場合も、ライセンス機能を制御し、その機能を使用可能にし、適切な制限を設定しなければなりません。

ライセンス機能を使うには、サイトのライセンス契約のタイプ、およびセッションまたは名前付きユーザーの最大数を確認する必要があります。サイトはどちらかのタイプのライセンス（同時使用または名前付きユーザー）を使い、両方を使うことはありません。

注意： 場合によっては、同時使用ライセンスまたは名前付きユーザー・ライセンスではなく、無制限のライセンスを保持しているサイトがあります。この場合に限り、ライセンス・メカニズムを使用禁止にしておきます。つまり、パラメータ・ファイル内の LICENSE_MAX_SESSIONS および LICENSE_SESSIONS_WARNING、LICENSE_MAX_USERS を指定しないか、またはこれら 3 つのパラメータをすべて 0 に設定します。

同時使用ライセンス

同時使用ライセンスは、指定したコンピュータ上のデータベースへ同時に接続できるセッションを制限します。インスタンスを起動する前に、同時実行のセッション数に対して制限を設定できます。実際には、初期インストール手順の一部として、この制限を設定しておいてください。また、データベースの稼働中に同時実行セッションの最大数も変更できます。

関連項目： 初期インストール手順の詳細は、第 2 章「Oracle データベースの作成」を参照してください。

接続権限

インスタンスのセッションの制限に達すると、その後データベースに接続できるのは、RESTRICTED SESSION 権限を持つユーザー（通常、DBA）だけになります。RESTRICTED SESSION 権限を持つユーザーがデータベースに接続すると、最大制限数に達していることを示すメッセージがこのユーザーに送信され、このメッセージが ALERT ファイルに書き込まれます。最大数に到達しているときには、不要なプロセスを停止 (KILL) する場合だけ、接続してください。Oracle のライセンス契約をアップグレードしていない限り、ライセンス制限を大きくしないでください。

最大同時実行セッションの設定に加えて、同時実行セッションの数に警告制限を設定できます。この制限に到達すると、追加のユーザーは引き続き接続できますが（最大制限まで）、Oracle は各接続で ALERT ファイルに適切なメッセージを書き込み、RESTRICTED SESSION 権限を持っている接続中の各ユーザーに、最大値に到達しそうであることを示す警告を送信します。

ユーザーが管理者権限で接続している場合も、この制限は適用されます。ただし、Oracle が制限を施行するのは、ユーザーが実行する最初の文の後になります。

同時使用制限の施行に加えて、Oracle はインスタンスごとに同時実行セッションの最大数を追跡記録します。この「高水位標」を使えます。

関連項目：セッションの停止の詳細は、4-18 ページの「セッションの停止」を参照してください。

Oracle ライセンス制限のアップグレードの詳細は、20-6 ページの「ライセンス制限と現行値を参照する」を参照してください。

Parallel Server の同時制限

Parallel Server で実行しているインスタンスの場合、各インスタンスには、固有の同時使用制限と警告制限を設定できます。ただし、それらのインスタンスの制限の合計は、サイトの同時使用ライセンスを超えてはなりません。

警告： 多重化ソフトウェアやハードウェア（TP モニターなど）を通じて Oracle に接続するセッションは、それぞれ個別に同時使用制限に寄与します。しかし、Oracle のライセンス・メカニズムは、この方法で接続されるセッション数を区別できません。サイトが多重化ソフトウェアやハードウェアを使う場合、そのことを考慮し、多重化されたセッションを計算に入れるために、最大同時使用制限をより小さく設定しなければなりません。

関連項目：パラレル・サーバー環境での制限の設定と変更の詳細は、『Oracle8 Parallel Server 概要および管理』を参照してください。

セッションの最大数を設定する

インスタンスに対して同時実行セッションの最大数を設定するには、パラメータ `LICENSE_MAX_SESSIONS` を設定してください。

```
LICENSE_MAX_SESSIONS = 80
```

この制限を設定する場合、警告制限の設定（`LICENSE_SESSIONS_WARNING`）が必要とされるわけではありません。ただし、警告制限を使うと、サイトが最大使用に近づいていることがわかりますから、最大制限の管理が容易になります。

セッション警告制限を設定する

インスタンスに警告制限を設定するには、インスタンスの起動に使うパラメータ・ファイルにパラメータ `LICENSE_SESSIONS_WARNING` を設定します。

セッション警告制限には、現行の同時実行最大制限（`LICENSE_MAX_SESSIONS`）よりも小さな値を設定してください。

データベースの稼働中に同時使用制限を変更する

データベースの稼働中に同時使用制限の最大数または警告制限を変更するには、適切なオプションを指定した `ALTER SYSTEM` コマンドを使います。次の文は、同時実行セッションの最大制限数を 100 に変更します。

```
ALTER SYSTEM SET LICENSE_MAX_SESSIONS = 100;
```

次の文は、警告制限と最大制限数の両方を変更します。

```
ALTER SYSTEM  
  SET LICENSE_MAX_SESSIONS = 64  
  LICENSE_SESSIONS_WARNING = 54;
```

どちらかの制限を現行のセッション数よりも小さい値に変更すると、現行セッションはそのままです。ただし、新しい設定は、インスタンスが停止されるまで、その後のすべての接続に施行されます。制限を永久的に変更するには、パラメータ・ファイル内の適切なパラメータの値を変更してください。

データベースの稼働中に同時使用制限を変更するには、`ALTER SYSTEM` 権限を持っていないければなりません。また、最大制限に到達しているインスタンスに接続するには、`RESTRICTED SESSION` 権限を持っていないければなりません。

警告： Oracle Server ライセンスを適切にアップグレードしていない限り、同時使用制限を大きくしないでください。詳細は、オラクル社カスタマー・サポートまで連絡してください。

名前付きユーザー制限

名前付きユーザー・ライセンスでは、指定したコンピュータ上で Oracle を使うことが許可されるユーザー数を制限します。このライセンスを施行するため、インスタンス起動前に、データベースに作るユーザー数に対して制限を設定できます。また、インスタンスの稼働中に最大ユーザー数を変更したり、制限を使用禁止にしたりできます。この制限に達すると、それ以上ユーザーを作れません。制限を超えてユーザーを作ろうとすると、すでに最大数のユーザーが作られているというエラーが戻され、このメッセージが ALERT ファイルに書き込まれます。

このメカニズムは、データベースにアクセスしているユーザーがそれぞれ一意のユーザー名を持っており、共有されているユーザー名がないことを前提として機能します。複数のユーザーが同一ユーザー名を使ってデータベースにアクセスしないようにしてください。

関連項目：Parallel Server で稼働しているインスタンスの場合、同一のデータベースに接続しているインスタンスはすべて同一の名前付きユーザー制限を持っていなければなりません。詳細は、『Oracle8 Parallel Server 概要および管理』を参照してください。

ユーザー制限を設定する

データベースに対して作られるユーザー数を制限するには、そのデータベースのパラメータ・ファイル内の LICENSE_MAX_USERS パラメータを設定します。次の例では、最大ユーザー数が 200 に設定されます。

```
LICENSE_MAX_USERS = 200
```

データベースを起動するときに、LICENSE_MAX_USERS よりも多くのユーザーがデータベース内に存在すると、Oracle は警告を戻し、ALERT ファイルに適切なメッセージを書き込みます。ユーザー数が制限を下回るまで、またはユーザーを削除するか、Oracle ライセンスをアップグレードするまで、追加のユーザーを作成することはできません。

ユーザー制限を変更する

名前付きユーザーの最大制限数を変更するには、LICENSE_MAX_USERS オプションを指定した ALTER SYSTEM コマンドを使います。次の文は、定義されるユーザーの最大数を 300 に変更します。

```
ALTER SYSTEM SET LICENSE_MAX_USERS = 300;
```

制限を現行のユーザー数よりも小さな値に変更しようとする、Oracle はエラーを戻し、引き続き古い制限を使います。制限の変更に成功すると、新しい制限はインスタンスを停止するまで効果があります。制限を半永久的に変更するには、パラメータ・ファイル内の LICENSE_MAX_USERS の値を変更してください。

名前付きユーザー制限の最大値を変更するには、ALTER SYSTEM 権限が必要です。

警告： Oracle ライセンスを適切にアップグレードしない限り、名前付きユーザー制限を大きくしないでください。詳細は、オラクル社カスタマー・サポートまで連絡してください。

ライセンス制限と現行値を参照する

V\$LICENSE データ・ディクショナリ・ビューを問い合わせることによって、すべてのライセンス設定の現行設定、およびセッションの現行数、インスタンスに対する現行セッションの最大数を参照できます。この情報を使って、より多くの現行セッションまたは名前付きユーザーを許可するために、Oracle ライセンスをアップグレードする必要があるかどうかを判断できます。

```
SELECT sessions_max s_max,
       sessions_warning s_warning,
       sessions_current s_current,
       sessions_highwater s_high,
       users_max
FROM v$llicense;
```

S_MAX	S_WARNING	S_CURRENT	S_HIGH	USERS_MAX
100	80	65	82	50

さらに、データベースを停止するときにセッションの高水位標がデータベースの ALERT ファイルに書き込まれるため、このファイルで高水位標を確認できます。

データベースに定義されている現行の端末ユーザー数を確認するには、次の問合せを使ってください。

```
SELECT COUNT(*) FROM dba_users;

COUNT(*)
-----
174
```

ユーザー認証

ここでは、ユーザーの認証について説明します。トピックは次のとおりです。

- データベース認証
- 外部認証
- 企業認証

ユーザーを定義する方法には、データベース・セッションを作る許可を与える前にユーザー ID を認証するための方法に応じて、次の 3 つの方法があります。

1. Oracle がユーザーの識別と認証の両方をするよう、Oracle を構成できます。これをデータベース認証と呼びます。
2. Oracle がユーザーの識別だけをする（認証はオペレーティング・システムまたはネットワーク・サービスでする）ように、Oracle を構成できます。これを外部認証と呼びます。
3. Oracle がユーザーの識別だけをする（認証は Oracle Security Service でする）ように、Oracle を構成できます。これを企業認証と呼びます。

データベース認証

ユーザーのデータベース認証を選択すると、そのユーザーのユーザー・アカウント、パスワード、認証の管理全体を Oracle が行います。Oracle にユーザーを認証させるには、ユーザーの登録または変更のときに、そのユーザーのパスワードを指定します。ユーザーはいつでも自分のパスワードを変更できます。パスワードは暗号形式で格納されます。データベースがマルチバイト・キャラクタ・セットを使っている場合も、各パスワードはシングルバイト・キャラクタによって構成されなければなりません。

データベース認証を使うときのセキュリティを高めるために、オラクル社では、アカウントのロック、パスワードのエージングと期限切れ、パスワードの履歴、パスワードの複雑さの検証など、パスワード管理の使用を薦めています。

次の文は、Oracle によって識別および認証されるユーザーを作成します。

```
CREATE USER scott IDENTIFIED BY tiger;
```

関連項目：CREATE USER コマンドと ALTER USER コマンドの詳細は、『Oracle8 Server SQL リファレンス』を参照してください。

有効なパスワードの詳細は、『Oracle8 Server SQL リファレンス』を参照してください。

Oracle パスワード管理の詳細は、第 19 章「セキュリティ方針の設定」を参照してください。

データベース認証の利点

次に、データベース認証の利点を示します。

- ユーザー・アカウントとすべての認証は、データベースで制御されます。データベースの外部のものに依存しません。

- Oracle は、データベース認証を使うときにセキュリティを高めるために強力なパスワード管理機能を提供しています。
- 小さなユーザー・コミュニティの管理が容易になります。

外部認証

ユーザーの外部認証を選択すると、ユーザー・アカウントは Oracle でメンテナンスされますが、パスワード管理とユーザー認証は外部サービスにより行われます。この外部サービスは、オペレーティング・システム、または Oracle Advanced Networking オプション (ANO) などのネットワーク・サービスとすることができます。

外部認証を使うと、データベースはデータベース・アカウントへのアクセスの制限をその基礎となるオペレーティング・システムまたはネットワーク認証サービスに依存します。データベース・パスワードはこのタイプのログインには使われません。オペレーティング・システムまたはネットワーク・サービスで許可されていれば、そのオペレーティング・システムにユーザーを認証させることができます。そのようにする場合、パラメータ

OS_AUTHENT_PREFIX を設定し、Oracle ユーザー名にこの接頭辞を使ってください。このパラメータは、あらゆるユーザーのオペレーティング・システム・アカウント名の先頭に追加する接頭辞を定義します。Oracle は、ユーザーが接続しようとする、その接頭辞付きのユーザー名をデータベース内の Oracle ユーザー名と比較します。

たとえば、OS_AUTHENT_PREFIX が次のように設定されている場合を想定します。

```
OS_AUTHENT_PREFIX=OPS$
```

オペレーティング・システム・アカウント名 "TSMITH" を持つユーザーが、Oracle データベースに接続しようとし、オペレーティング・システムによって認証される場合、Oracle は対応するデータベース・ユーザー "OPS\$TSMITH" の存在を調べ、存在していれば接続を許可します。オペレーティング・システムによって認証されるユーザーへのすべての参照には、"OPS\$TSMITH" のように、接頭辞が含まれていなければなりません。

このパラメータのデフォルト値は "OPS\$" であり、以前のバージョンの Oracle と下位互換です。ただし、接頭辞の値は他の文字列や NULL 文字列（2 つの二重引用符 " " で指定します）も設定できます。NULL 文字列を使うと、オペレーティング・システム・アカウント名に接頭辞は追加されませんから、Oracle ユーザー名とオペレーティング・システム・ユーザー名は完全に一致します。

OS_AUTHENT_PREFIX を設定すると、データベースの存続中は同じ設定のまま維持されます。接頭辞を変更する場合、古い接頭辞を含むデータベース・ユーザー名は、パスワード認証を使うように変更しない限り、そのユーザー名を使っては接続を確立できません。

Oracle によって識別され、オペレーティング・システムまたはネットワーク・サービスによって認証されるユーザーの作成には、次のコマンドを使います。

```
CREATE USER scott IDENTIFIED EXTERNALLY;
```

CREATE USER IDENTIFIED EXTERNALLY を使うと、データベース管理者は、オペレーティング・システムまたはネットワーク・サービスによる認証が必要なデータベース・アカウントを作れます。ただし、パスワードを使っても認証できません。

関連項目：OS_AUTHENT_PREFIX パラメータのテキストは、オペレーティング・システムにより大 / 小文字の区別が必要になります。初期化パラメータの詳細は、オペレーティング・システム固有の Oracle のマニュアルを参照してください。

オペレーティング・システム認証

デフォルトでは、オペレーティング・システムで認証されており、安全な接続を介したログインだけが許可されます。したがって、オペレーティング・システムにユーザーを認証させる場合、デフォルトでは、そのユーザーは Net8 を介してデータベースに接続できません。つまり、この接続では Net8 を使うため、マルチスレッド・サーバーでは接続できません。このデフォルトの制限により、リモート・ユーザーはネットワーク接続を介して別のオペレーティング・システムのユーザーのふりをすることができません。

ネットワーク接続を介してリモート・ユーザーが別のオペレーティング・システムのユーザーのふりをするについて心配がなく、ネットワーク・クライアントでオペレーティング・システム・ユーザー認証を使う場合は、データベースのパラメータ・ファイルで REMOTE_OS_AUTHENT(デフォルト設定は FALSE) を TRUE に設定してください。初期化パラメータ REMOTE_OS_AUTHENT を TRUE に設定すると、RDBMS は保護のない接続を介して受信したクライアントのオペレーティング・システムのユーザー名を受け入れてアカウント・アクセスに使用できます。この変更は、次回インスタンスを起動し、データベースをマウントするときに有効となります。

ネットワーク認証

ネットワーク認証は、Oracle Advanced Networking オプション (ANO) で行いますが、このオプションは Kerberos などのサードパーティのサービスを利用するように設定できます。ANO を唯一の外部認証サービスとして利用する場合、ANO では安全な接続しか行えないので、パラメータ REMOTE_OS_AUTHENT の設定は無関係です。

関連項目：ネットワーク認証の詳細は、『Oracle8 Server 分散システム』と『Oracle Security Server ガイド』を参照してください。

外部認証の利点

次に、外部認証の利点を示します。

- スマート・カード、フィンガープリント、Kerberos、オペレーティング・システムなどの認証メカニズムの選択肢も使用できます。
- すでに上記のようなある種の外部認証メカニズムを使用している場合、データベースでそのメカニズムを使用すると管理費用も低減できます。

企業認証

ユーザーの企業認証を選択すると、ユーザー・アカウントは Oracle でメンテナンスされますが、パスワード管理とユーザー認証は Oracle Security Service により行われます。この認証サービスは複数の Oracle データベース・サーバーの間で共有でき、このサービスを使用するとユーザーの認証と許可の情報を中央で管理できます。

Oracle によって識別され、Oracle Security Service によって 認証されるユーザー（グローバルなユーザーと呼ぶ）の作成には、次のコマンドを使います。

```
CREATE USER scott IDENTIFIED GLOBALLY as '<external name>';
```

関連項目：<EXTERNAL NAME> 文字列の詳細は、『Oracle8 Server 分散システム』と『Oracle Security Server ガイド』を参照してください。

企業認証の利点

次に、企業認証の利点を示します。

- 多数のデータベースを持つ大きなユーザー・コミュニティの管理が容易になります。
- 業界標準の公用鍵の証明書を使用すると、相互操作の機会が増えます。

関連項目：企業認証の詳細は、『Oracle8 Server 分散システム』と『Oracle Security Server ガイド』を参照してください。

Oracle ユーザー

各 Oracle データベースには、有効なデータベース・ユーザーのリストがあります。データベースに対してアクセスするには、ユーザーは必ずデータベース・アプリケーションを稼働させた上で、データベース中で定義した有効なユーザー名を使って、データベース・インスタンスに接続しなければなりません。ここでは、データベースに対するユーザーの作成法、管理法、削除法について解説します。

- ユーザーを作成する
- ユーザーを変更する
- ユーザーを削除する

ユーザーを作成する

データベース・ユーザーを作るには、CREATE USER システム権限が必要です。新しいユーザーを作るときには、たとえその作成者が指定する表領域の割当て制限を有していなくても、データベース上の任意の表領域を表領域割当て制限として指定できます。こうした権限があるため、通常セキュリティ管理者が CREATE USER システム権限を有する唯一のユーザーです。

Enterprise Manager/GUI の「ユーザー作成」プロパティ・シートまたは SQL コマンド CREATE USER のどちらかを使って、ユーザーを作成します。このどちらかのオプションを

使って、新たにユーザーのデフォルト、一時セグメント表領域、表領域割当て制限、プロファイルも指定できます。

```
CREATE USER OPS$jward
  IDENTIFIED EXTERNALLY
  DEFAULT TABLESPACE data_ts
  TEMPORARY TABLESPACE temp_ts
  QUOTA 100M ON test_ts
  QUOTA 500K ON data_ts
  PROFILE clerk;
```

関連項目：新たに作成されたユーザーは、CREATE SESSION システム権限が付与されるまで、データベースに接続できません。21-16 ページの「システム権限とロールを付与する」を参照してください。

名前を指定する

各データベース内では、ユーザー名は他のユーザー名とロールに対して一意でなければなりません。ユーザーとロールは同じ名前を持つことができません。さらに、各ユーザーには対応するスキーマがあります。スキーマ内では、各スキーマ・オブジェクトに必ず一意の名前を指定しなければなりません。

マルチバイト・キャラクタ・セットのユーザー名 マルチバイト・キャラクタ・セットを使うデータベースでは、各ユーザー名には、シングルバイト・キャラクタを少なくとも 1 つ含める必要があります。ユーザー名にマルチバイト・キャラクタしか使われていないと、暗号化されたユーザー名 / パスワードの組合せの安全性がかなり低くなります。

ユーザー認証を設定する

前の CREATE USER 文では、新規ユーザーをオペレーティング・システムを使って認証しました。ユーザー名には、デフォルトの接頭辞 "OPS\$" が含まれています。

OS_AUTHENT_PREFIX パラメータを異なるものに設定する場合（接頭辞を指定しなかったり、別の接頭辞を指定したりした場合）必要に応じて接頭辞を省略または適切な接頭辞に置き換えることでユーザー名を修正できます。

また、次のようにデータベースとパスワードを使って認証されるユーザーも作成できます。

```
CREATE USER jward
  IDENTIFIED BY airplane
  . . . ;
```

この場合、接続に成功するために、接続ユーザーはデータベースに対して正しい接続パスワードを指定しなければなりません。

マルチバイト・キャラクタ・セットを使う場合のユーザー・パスワード マルチバイト・キャラクタ・セットを使うデータベースでは、パスワードには必ずシングルバイト・キャラクタだけを使ってください。パスワードでは、マルチバイト・キャラクタは受け入れられません。

関連項目：有効なパスワードの詳細は、『Oracle8 Server SQL リファレンス』を参照してください。

デフォルト表領域を割り当てる

各ユーザーはそれぞれデフォルト表領域を持っています。ユーザーがスキーマ・オブジェクトを作って、このオブジェクトを格納する表領域を割り当てないと、このオブジェクトはユーザーのデフォルト表領域に格納されます。

すべてのユーザーのデフォルトの表領域に対するデフォルトの設定値は、SYSTEM 表領域です。ユーザーがオブジェクトを作らない場合には、デフォルトの設定値が最適です。ただし、ユーザーが任意のタイプのオブジェクトを作成できる場合、そのユーザーのデフォルトの表領域を個別に設定するように考慮しなければなりません。ユーザー作成中にユーザーのデフォルト表領域を設定しておき、作成後に変更できます。ユーザーのデフォルト表領域を変更すると、設定値の変更後に作られたオブジェクトだけがこの変更の影響を受けます。

指定する表領域を決定する際、次の項目を考慮します。

- ユーザーが任意のオブジェクト（表、ビュー、クラスタなど）を作る権限を持っている場合に限り、ユーザーのデフォルト表領域を設定する。
- ユーザーが割当て制限を有している表領域には、ユーザーのデフォルトの表領域を設定する。
- 可能な場合には、データ・ディクショナリ・オブジェクトとユーザー・オブジェクトとの間の競合を削減するため、SYSTEM 表領域以外の表領域にユーザーのデフォルト表領域を設定する。

前の例の CREATE USER 文では、JWARD のデフォルトの表領域は DATA_TS になります。

一時表領域を割り当てる

各ユーザーは、一時表領域も持っています。ユーザーが一時セグメントを必要とする SQL 文を実行すると、このセグメントはユーザーの一時表領域に格納されます。

ユーザーの一時表領域を明示的に設定しなければ、デフォルトは SYSTEM 表領域です。ただし、各ユーザーが一時表領域を設定することによって、一時セグメントとそれ以外のタイプのセグメントとの間で発生するファイル競合が減少します。ユーザーの一時表領域は、ユーザーの作成中に設定もできますし、作成後に変更もできます。

前の例の CREATE USER 文では、JWARD の一時表領域は TEMP_TS です。これは一時セグメントだけを格納するために明示的に作られた表領域です。

表領域の割当て制限を割り当てる

任意の表領域に対する表領域割当て制限を、各ユーザーに割り当てられます。割当て制限による影響は次のとおりです。

- ユーザーになんらかのオブジェクトを作る権限がある場合に、ユーザーは指定した表領域中にオブジェクトを作成できる。

- 指定した表領域内にあるユーザーのオブジェクトの記憶域に対して割り当てられる領域が、割当て制限以内に設定される。

デフォルトでは、ユーザーにはデータベース中の表領域に関する割当て制限はありません。ユーザーにスキーマ・オブジェクトを作る権限がある場合には、必ずオブジェクトを作成できる割当て制限を割り当てなければなりません。最低限、デフォルト表領域に対する割当て制限をユーザーに割り当て、さらにオブジェクトを作る際に他の表領域に対する割当て制限を追加で割り当てます。

ユーザーに対して、各表領域の固有のディスク領域に対して割当て制限を個々に割り当てるか、またはすべての表領域中のディスク領域に対して無制限に割り当てるかのどちらかを実施できます。個々に割当て制限を設定することによって、ユーザーのオブジェクトがデータベースの領域を大幅に消費することを未然に防げます。

ユーザーの表領域の割当て制限は、ユーザーの作成中に設定もできますし、作成後に追加または変更もできます。新たな割当て制限が古い制限値よりも少なければ、次の条件の場合に真となります。

- ユーザーがすでに新しい表領域割当て制限を超過している場合、これらのオブジェクトを合わせた領域が新しい割当て制限よりも下回らない限り、その表領域のユーザー・オブジェクトにさらに領域を割り当てられない。
- ユーザーが新しい表領域割当て制限を超過していない場合、または表領域上のユーザーのオブジェクトで使われる領域が新しい表領域割当て制限よりも少ない場合、そのユーザーのオブジェクトに新しい割当て制限の領域を割り当てることができる。

表領域アクセスを取り消す ユーザーの表領域アクセスを取り消すには、そのユーザーの現行の割当て制限をゼロに変更します。いったん、割当て制限 0（ゼロ）を割り当てておけば、取り消された表領域中のユーザーのオブジェクトはそのまま残りますが、そのオブジェクトを新たな領域に割り当ててすることはできません。

UNLIMITED TABLESPACE システム権限 データベース内の表領域をユーザーが無制限に使うことを許可するには、ユーザーに UNLIMITED TABLESPACE システム権限を付与します。これにより、ユーザーに対して明示的に指定されている表領域割当て制限がすべて上書きされます。この権限を後で取り消すと、明示的に指定された割当て制限が再び有効になります。なお、この権限はロールに対してではなく、ユーザーに限り付与できます。

UNLIMITED TABLESPACE システム権限を付与する前に、その措置の利点と欠点を考慮しておかなければなりません。

利点

- 1 文で、データベースのすべての表領域を無制限にアクセスする権限をユーザーに付与できる。

欠点

- この権限によって、そのユーザーに対する明示的な表領域割当て制限がすべて置き換えられる。

- UNLIMITED TABLESPACE システム権限を使うことによって、表領域アクセスを選択的にユーザーから取り消すことが不可能となる。その権限を取り消した後に限り、選択的にアクセスを付与できる。

デフォルト・ロールを設定する

CREATE USER 文ではユーザーのデフォルト・ロールを設定できません。最初にユーザーを作るときに、そのユーザーのデフォルト・ロールは ALL に設定され、これによってユーザーにその後付与されるロールはすべてデフォルト・ロールになります。ユーザーのデフォルト・ロールを変更するには、ALTER USER コマンドを使ってください。

警告： ユーザー・ロール以外のロールを作成する場合、そのロールを暗黙に付与し、デフォルトのロールとして追加します。複数の MAX_ENABLED_ROLES がある場合、ログインでエラーになります。このエラーを回避するには、ユーザーのデフォルトのロールを MAX_ENABLED_ROLES より小さな値に変更します。したがって、ユーザー・ロールを作成する前に SYS と SYSTEM の DEFAULT ROLE の設定を変更する必要があります。

ユーザーを変更する

ユーザーは自分のパスワードを変更できます。ただし、ユーザーのセキュリティ定義域の他のオプションを変更するには、ALTER USER システム権限を持っていなければなりません。通常、セキュリティ管理者が唯一このシステム権限を有するユーザーであるのは、その権限によりユーザーのセキュリティ定義域を修正できるからです。なお、修正を実行するユーザーが表領域に対する割当て制限を有していなくても、データベースの任意の表領域上のユーザーのために表領域割当て制限を設定する機能が、この権限には含まれています。

ユーザーのセキュリティの設定値を変更するには、Enterprise Manager/GUI の「ユーザー変更」プロパティ・シート、または SQL コマンド ALTER USER を使います。ユーザー・セキュリティの設定値の変更は、現在のセッションではなく、それ以降のセッションに反映されます。

次の文は、ユーザー AVYRROS のセキュリティの設定値を変更します。

```
ALTER USER avyrros
  IDENTIFIED EXTERNALLY
  DEFAULT TABLESPACE data_ts
  TEMPORARY TABLESPACE temp_ts
  QUOTA 100M ON data_ts
  QUOTA 0 ON test_ts
  PROFILE clerk;
```

この ALTER USER 文によって、AVYRROS のセキュリティの設定値は次のように変更されます。

- AVYRROS のオペレーティング・システム・アカウントを使って認証を変更する。

- AVYRROS のデフォルトと一時表領域を明示的に設定する。
- AVYRROS には、DATA_TS 表領域として 100MB 分の割当て制限を指定する。
- TEST_TS に対する AVYRROS の割当て制限を取り消す。
- AVYRROS を CLERK プロファイルに割り当てる。

ユーザーの認証メカニズムを変更する

DBA 以外のユーザーの大半は Enterprise Manager を使いませんが、次のように ALTER USER コマンドを使って自分のパスワードを変更できます。

```
ALTER USER andy  
  IDENTIFIED BY swordfish;
```

特別な権限（データベースに接続するための権限以外の権限）がなくても、ユーザーは誰でもこの方法で自分のパスワードを変更できます。ユーザーは自分のパスワードを必要に応じて変更するとよいでしょう。

Oracle データベースの認可、オペレーティング・システムの認可、企業認証の間を変更するには、ALTER USER 権限を持っていなければなりません。通常、DBA だけがこの権限を持つべきです。

マルチバイト・キャラクタ・セットを使う場合のパスワード マルチバイト・キャラクタ・セットを使うデータベースでは、パスワードには必ずシングルバイト・キャラクタだけを使ってください。パスワードでは、マルチバイト・キャラクタは受け入れられません。

関連項目：有効なパスワードの詳細は、『Oracle8 Server SQL リファレンス』を参照してください。

ユーザーのデフォルト・ロールを変更する

デフォルト・ロールは、ユーザーがセッションを作ったときに、自動的にそのユーザーに対して使用可能となるように設定されたものです。ユーザーにはゼロ以上のデフォルト・ロールを割り当てることができます。

関連項目：ユーザーのデフォルト・ロールの変更に関する詳細は、第 21 章「ユーザー権限とロールの管理」を参照してください。

ユーザーを削除する

ユーザーが削除されると、そのユーザーと対応するスキーマがデータ・ディクショナリから削除され、さらにユーザーのスキーマ内にスキーマ・オブジェクトがある場合、そのすべてのオブジェクトがただちに削除されます。

注意： ユーザーのスキーマおよびそれに対応するオブジェクトは残したままで、ユーザーからデータベースへのアクセスを取り消さなければならぬ場合、そのユーザーから CREATE SESSION 権限を取り消してください。

現在データベースに接続されているユーザーは削除できません。接続中のユーザーを削除するには、まず Enterprise Manager/GUI、または KILL SESSION 句を指定した SQL コマンドを使って、そのユーザーのセッションを停止しなければなりません。

ユーザーとそのユーザーのスキーマ・オブジェクト（ある場合）をすべて削除するには、DROP USER 権限が必要です。DROP USER システム権限は非常に強力な権限であるため、通常セキュリティ管理者がこの権限を持つ唯一のユーザーになります。

Enterprise Manager/GUI の削除メニュー項目または SQL コマンド DROP USER を使って、データベースからユーザーを削除します。

ユーザーのスキーマにスキーマ・オブジェクトが含まれている場合は、ユーザー、対応付けられているすべてのオブジェクト、そのユーザーの表に依存している外部キーをすべて削除するには、CASCADE オプションを使います。CASCADE を指定していない場合、ユーザーのスキーマにオブジェクトが含まれていると、エラー・メッセージが戻され、ユーザーは削除されません。ユーザーのスキーマがオブジェクトを含んでいるユーザーを削除する場合にはあらかじめ、どのオブジェクトがユーザーのスキーマに含まれているかを徹底的に調査し、削除することによる影響を事前によく調査しておかなければなりません。カスケードによる影響のうち、直接現れないものに注意してください。たとえば、表を所有するユーザーを削除する場合は、ビューまたはプロシージャがその表に依存していないかどうかを確認してください。

```
DROP USER jones CASCADE;
```

関連項目： セッションの停止の詳細は、4-18 ページの「セッションの停止」を参照してください。

プロファイルでリソースを管理

プロファイルは一連の命名されたリソース制限です。リソース制限が起動されると、データベースおよびインスタンス・リソースの使用が、そのプロファイルに指定されたとおりに制限されます。各ユーザーごとにプロファイルを割り当てたり、固有のプロファイルを持たないすべてのユーザーに対してデフォルト・プロファイルを割り当てたりできます。プロファイルを実施するためには、リソース制限をデータベース全体に起動しなければなりません。

ここでは、プロファイル管理について説明します。トピックは次のとおりです。

- プロファイルを作成する
- プロファイルを割り当てる
- プロファイルを変更する
- 複合制限を使用する
- プロファイルを削除する
- リソース制限を使用可能、使用禁止にする

プロファイルを作成する

プロファイルを作るには、CREATE PROFILE システム権限が必要です。Enterprise Manager/GUI の「プロファイル作成」プロパティ・シートまたは SQL コマンド CREATE PROFILE のどちらかを使って、プロファイルを作成します。このときに、特定のリソース制限を明示的に設定できます。

次の文は、CLERK プロファイルを作ります。

```
CREATE PROFILE clerk LIMIT
  SESSIONS_PER_USER 2
  CPU_PER_SESSION unlimited
  CPU_PER_CALL 6000
  LOGICAL_READS_PER_SESSION unlimited
  LOGICAL_READS_PER_CALL 100
  IDLE_TIME 30
  CONNECT_TIME 480;
```

新しいプロファイルに対してリソース制限をまったく指定していない場合には、DEFAULT プロファイルによって制限値が設定されます。また、DEFAULT プロファイルに対しても制限を指定できます。

DEFAULT プロファイルを使う

各データベースには DEFAULT というプロファイルがあります。DEFAULT プロファイルの制限は次の場合に使われます。

- ユーザーにプロファイルが明示的に割り当てられていない場合、ユーザーは DEFAULT プロファイルすべての制限に準拠する。

- プロファイルにまったく制限を指定していない場合、DEFAULT プロファイルに対応する制限を使う。

最初、DEFAULT プロファイルの制限はすべて UNLIMITED に設定されます。ただし、DEFAULT プロファイルのユーザーが無制限にリソースを消費することを未然に防ぐために、セキュリティ管理者は Enterprise Manager の「プロファイル変更」ダイアログ・ボックス、または次のような ALTER PROFILE 文を使ってデフォルトの制限を変更しなければなりません。

```
ALTER PROFILE default LIMIT
    . . . ;
```

ALTER PROFILE システム権限を有するユーザーは、DEFAULT プロファイル中でその制限を調整できます。なお、DEFAULT プロファイルは削除できません。

プロファイルを割り当てる

作ったプロファイルは、データベース・ユーザーに割り当てることができます。各ユーザーは任意に単一のプロファイルを割り当てられます。すでにプロファイルを有しているユーザーにプロファイルを割り当てると、最新のプロファイルの割当てが前回割り当てたプロファイルを置き換えます。プロファイルの割当ては現在のセッションにはまったく影響しません。なお、プロファイルは、ロールや他のプロファイルではなく、ユーザーに限り、割り当てられます。

ユーザーにプロファイルを割り当てするには、Enterprise Manager/GUI の「プロファイル割当て」ダイアログ・ボックス、または SQL コマンド CREATE USER、ALTER USER を使います。

関連項目：ユーザーのプロファイルの割当ての詳細は、20-10 ページの「ユーザーを作成する」および 20-14 ページの「ユーザーを変更する」を参照してください。

プロファイルを変更する

Enterprise Manager/GUI の「プロファイル変更」プロパティ・シート、または SQL コマンド ALTER PROFILE を使ってプロファイルのリソース制限の設定値を変更できます。プロファイルを変更するには、ALTER PROFILE システム権限が必要です。

プロファイルの制限を調整すると、そのプロファイルの調整前の設定値は上書きされます。DEFAULT の値で制限を調整することによって、リソース制限はデータベースのデフォルトの制限値に戻ります。プロファイルを変更する際に調整しなかったプロファイルは、すべて前回の設定値を維持します。プロファイルに対してなされた変更は、現行のセッションには影響しません。新しいプロファイルの設定値は、プロファイルの変更に作られたセッションに対してだけ使われます。

次の文は、CLERK プロファイルを変更します。

```
ALTER PROFILE clerk LIMIT
    CPU_PER_CALL default
    LOGICAL_READS_PER_SESSION 20000;
```

関連項目：デフォルトのプロファイルの詳細は、20-17 ページの「DEFAULT プロファイルを使う」を参照してください。

複合制限を使用する

複合制限によりセッションに対するリソース・コストの合計を制限できます。プロファイルに対する固有のリソース制限を明示的に設定するばかりでなく、単一複合制限は、プロファイル上のすべてのリソース制限に対するアカウントを設定できます。Enterprise Manager/GUI の「プロファイル作成」または「プロファイル変更」プロパティ・シートの「複合制限」チェックボックスか、または SQL コマンド CREATE PROFILE、ALTER PROFILE の COMPOSITE_LIMIT パラメータを使って、プロファイルの複合制限を設定できます。複合制限は、1 つのサービス単位で設定されます。サービス単位とは使われるリソースの合計量です。

次の CREATE PROFILE 文は、COMPOSITE_LIMIT パラメータを使って定義されています。

```
CREATE PROFILE clerk LIMIT
  COMPOSITE_LIMIT 20000
  SESSIONS_PER_USER 2
  CPU_PER_CALL 1000;
```

明示的に指定したリソース制限と複合制限は、プロファイルごとに共存できます。最初に適合した制限は、セッションのアクティビティを停止します。複合制限は、システム・リソースの使用を制限する際に、柔軟性のある制限を実現します。

複合制限の値を判断する

複合制限に対する適切なサービス単位の設定は、平均的なプロファイル・ユーザーによって使われるリソースの総合計に応じて異なります。各固有のリソース制限とともに履歴情報は、平均的なプロファイル・ユーザーのための複合リソースの通常範囲を決定するために、必ず収集しなければなりません。

リソース・コストを設定する

システムにはそれぞれ独自の特徴があります。中には他よりも価値があるシステム・リソースもあります。Oracle では、各システム・リソースにコストを設定できます。コストは、データベース・レベルで各システム・リソースに重みを付けます。コストが唯一プロファイルの複合制限に適用されます。ただし、個々のリソース制限を明示的に設定する場合には、コストは適用されません。

リソース・コストを設定するには、ALTER RESOURCE システム権限が必要です。

CPU_PER_SESSION、LOGICAL_READS_PER_SESSION、CONNECT_TIME、PRIVATE_SGA を含めて、コストは特定のリソースにだけ設定できます。データベースのコストは、SQL コマンド ALTER RESOURCE COST を使って設定します。

```
ALTER RESOURCE COST
  CPU_PER_SESSION 1
  LOGICAL_READS_PER_SESSION 50;
```

多額コストとはリソースが非常に高価であることを意味するのに対して、少額コストとはリソースが高価でないことを意味しています。デフォルトでは、最初に各リソースにコスト 0 が設定されます。コスト 0 は、リソースを複合制限で考慮してはならない（すなわち、このリソースを使うコストはかからない）ことを意味します。なお、リソースをコスト NULL に指定できません。

関連項目：リソース・コストの設定に関する詳細と推奨事項は、オペレーティング・システム固有の Oracle のマニュアルを参照してください。

プロファイルを削除する

プロファイルを削除するには、DROP PROFILE システム権限が必要です。Enterprise Manager/GUI または SQL コマンド DROP PROFILE を使って、プロファイルを削除します。現在ユーザーに割り当てられているプロファイルを正常に削除するには、CASCADE オプションを使います。

次の文では、CLERK プロファイルを削除します（このプロファイルがユーザーに割り当てられていても削除されます）。

```
DROP PROFILE clerk CASCADE;
```

削除されるプロファイルに現在割り当てられているユーザーは、自動的に DEFAULT のプロファイルに割り当てられます。なお、DEFAULT プロファイルは削除できません。プロファイルが削除されていても、その削除は現在のアクティブ・セッションには影響を及ぼしません。プロファイルが削除された後に作られたセッションに限り、修正済みのプロファイルの割当てに従います。

リソース制限を使用可能、使用禁止にする

プロファイルの作成、ユーザーへの割当て、変更、削除は、許可されている任意のデータベース・ユーザーによって随時実行されますが、プロファイルに設定されたリソース制限が施行されるのは、対応するデータベースのリソース制限が使用可能な場合だけです。リソース制限の強制の可否は、次の部分で説明する 2 種類の方法で設定できます。

データベースがオープンしている状態でリソース制限の強制を変更するには、ALTER SYSTEM システム権限が必要です。

起動前にリソース制限を使用可能、使用禁止にする

データベースを一時的に停止する場合には、データベースのパラメータ・ファイル中の RESOURCE_LIMIT の初期パラメータで、リソース制限の強制の可否を設定できます。パラメータとして有効な値は、TRUE（強制可）と FALSE です。特に何も指定しなければ、パラメータの値は FALSE に設定されます。一度、パラメータ・ファイルを編集したら、データベース・インスタンスを再起動して、その編集を反映しなければなりません。インスタンスが起動されるたびに、新たなパラメータ値によってリソース制限の強制の使用可能または使用禁止が設定されます。

データベースのオープン中にリソース制限を使用可能、使用禁止にする

データベースを一時的に停止できない場合やリソース制限機能を一時的に変更しなければならぬ場合には、SQL コマンド ALTER SYSTEM を使ってリソース制限の強制の可否のいずれかを選択しなければなりません。一度、インスタンスを起動すると、ALTER SYSTEM 文は RESOURCE_LIMIT パラメータで値の設定を置き換えます。たとえば、次の指定文はデータベースに対するリソース制限を強制的に設定します。

```
ALTER SYSTEM  
SET RESOURCE_LIMIT = TRUE;
```

注意： これは、パスワード・リソースに適用されません。

ALTER SYSTEM 文はリソース制限の強制を永続的に決定するわけではありません。データベースが停止または再起動されると、RESOURCE_LIMIT パラメータに設定されている値によってリソース制限の強制が決定します。

データベース・ユーザーとプロファイルに関する情報のリスト

データ・ディクショナリは、すべてのユーザーとプロファイルに関する情報を格納しています。次の情報が含まれています。

- データベース内のすべてのユーザー
- 表、クラスタ、索引に対する各ユーザーのデフォルト表領域
- 一時セグメントのための各ユーザーの表領域
- 各ユーザーの領域割当て制限（ある場合）
- 各ユーザーに割り当てられているプロファイルとリソース制限
- 適用可能な各システム・リソースに割り当てられているコスト
- 各現行セッションのメモリー使用

次のデータ・ディクショナリ・ビューは、データベース・ユーザーおよびプロファイルに関係のあるものです。

- ALL_USERS
- USER_USERS
- DBA_USERS
- USER_TS_QUOTAS
- DBA_TS_QUOTAS
- USER_PASSWORD_LIMITS

- USER_RESOURCE_LIMITS
- DBA_PROFILES
- RESOURCE_COST
- V\$SESSION
- V\$SESSTAT
- V\$STATNAME

関連項目：各ビューの詳細は、『Oracle8 Server リファレンス・マニュアル』を参照してください。

ユーザーとプロファイルに関する情報を記述する例

この部分の例は、次の文を実行したデータベースを想定しています。

```
CREATE PROFILE clerk LIMIT
  SESSIONS_PER_USER 1
  IDLE_TIME 30
  CONNECT_TIME 600;
```

```
CREATE USER jfee
  IDENTIFIED BY wilddcat
  DEFAULT TABLESPACE users
  TEMPORARY TABLESPACE temp_ts
  QUOTA 500K ON users
  PROFILE clerk;
```

```
CREATE USER dcranney
  IDENTIFIED BY bedrock
  DEFAULT TABLESPACE users
  TEMPORARY TABLESPACE temp_ts
  QUOTA unlimited ON users;
```

```
CREATE USER userscott
  IDENTIFIED BY "scott1"
  PASSWORD_LIFETIME 60
  PASSWORD_GRACE_TIME 10;
```

すべてのユーザーとその関連情報を記述する

次の問合せは、データベースに定義されているユーザーとユーザーに関連する情報を示したものです。

```
SELECT username, profile, account_status from dba_users;
```

USERNAME	PROFILE	ACCOUNT_STATUS
SYS	DEFAULT	OPEN
SYSTEM	DEFAULT	OPEN
BLAKE	DEFAULT	OPEN
SCOTT	DEFAULT	OPEN
ADAMS	DEFAULT	OPEN
JONES	DEFAULT	OPEN
CLARK	DEFAULT	OPEN
U	DEFAULT	LOCKED
USERSCOTT	PROF	EXPIRED

パスワードはすべてセキュリティ維持のために暗号化されています。

表領域割当て制限をすべて記述する

次の問合せは、各ユーザーごとに割り当てられている表領域割当て制限をすべて示すものです。

```
SELECT * FROM sys.dba_ts_quotas;
```

TABLESPACE	USERNAME	BYTES	MAX_BYTES	BLOCKS	MAX_BLOCKS
SYSTEM	SYSTEM	0	0	0	0
SYSTEM	JFEE	0	512000	0	250
SYSTEM	DCRANNEY	0	-1	0	-1

特定の割当て制限を設定すると、正確な数値が MAX_BYTES 列に示されます。制限のない割当ては "-1" で示されます。

すべてのプロファイルと割り当てられている制限をすべて記述する

次の問合せは、データベース内のすべてのプロファイルと各プロファイル内の各制限に対応する設定を記述します。

```
SELECT * FROM sys.dba_profiles
ORDER BY profile;
PROFILE                                RESOURCE_NAME                        RESOURCE      LIMIT
-----
DEFAULT                                COMPOSITE_LIMIT                      KERNEL        UNLIMITED
DEFAULT                                SESSIONS_PER_USER                    KERNEL        1
DEFAULT                                CPU_PER_CALL                         KERNEL        UNLIMITED
DEFAULT                                LOGICAL_READS_PER_CALL              KERNEL        UNLIMITED
DEFAULT                                CONNECT_TIME                         KERNEL        30
DEFAULT                                IDLE_TIME                           KERNEL        600
DEFAULT                                LOGICAL_READS_PER_SESSION           KERNEL        UNLIMITED
DEFAULT                                CPU_PER_SESSION                     KERNEL        UNLIMITED
DEFAULT                                PRIVATE_SGA                         KERNEL        UNLIMITED
DEFAULT                                FAILED_LOGIN_ATTEMPTS               PASSWORD      UNLIMITED
DEFAULT                                PASSWORD_LIFE_TIME                  PASSWORD      UNLIMITED
DEFAULT                                PASSWORD_REUSE_MAX                   PASSWORD      UNLIMITED
DEFAULT                                PASSWORD_LOCK_TIME                  PASSWORD      UNLIMITED
DEFAULT                                PASSWORD_GRACE_TIME                 PASSWORD      UNLIMITED
DEFAULT                                PASSWORD_VERIFY_FUNCTION             PASSWORD      UNLIMITED
DEFAULT                                PASSWORD_REUSE_TIME                  PASSWORD      UNLIMITED
PROF                                    COMPOSITE_LIMIT                      KERNEL        DEFAULT
PROF                                    PRIVATE_SGA                         KERNEL        DEFAULT
PROF                                    CONNECT_TIME                         KERNEL        DEFAULT
PROF                                    IDLE_TIME                           KERNEL        DEFAULT
PROF                                    LOGICAL_READS_PER_CALL              KERNEL        DEFAULT
PROF                                    LOGICAL_READS_PER_SESSION           KERNEL        DEFAULT
PROF                                    SESSIONS_PER_USER                    KERNEL        DEFAULT
PROF                                    CPU_PER_CALL                         KERNEL        DEFAULT
PROF                                    CPU_PER_SESSION                     KERNEL        DEFAULT
PROF                                    FAILED_LOGIN_ATTEMPTS               PASSWORD      5
PROF                                    PASSWORD_LIFE_TIME                  PASSWORD      60
PROF                                    PASSWORD_REUSE_MAX                   PASSWORD      UNLIMITED
PROF                                    PASSWORD_LOCK_TIME                  PASSWORD      1
PROF                                    PASSWORD_GRACE_TIME                 PASSWORD      10
PROF                                    PASSWORD_VERIFY_FUNCTION             PASSWORD      UNLIMITED
PROF                                    PASSWORD_REUSE_TIME                  PASSWORD      60
32 rows selected.
```

ユーザー・セッションあたりのメモリーの使用状況を確認する

次の問合せは、すべての現行セッションを記述し、セッションあたりの Oracle ユーザーと現在のメモリー使用を示します。

```
SELECT username, value || 'bytes' "Current session memory"
FROM v$session sess, v$sesstat stat, v$statname name
WHERE sess.sid = stat.sid
AND stat.statistic# = name.statistic#
AND name.name = 'session memory';
```

"Current session memory" に示される領域は、マルチスレッド・サーバーを通じて接続している各セッションの共有プール内に割り当てられています。PRIVATE_SGA リソース制限によって、ユーザーあたりに割り当てられるメモリー容量を制限できます。

インスタンス起動後に、各セッションに割り当てられた最大メモリーを確認するには、この問合せの 'session memory' を 'max session memory' に置き換えてください。

例

ここでは、この章全体で説明しているファンクションを使用する例を示します。

1. 次の文は、プロファイル prof を作ります。

```
CREATE PROFILE prof limit
  FAILED_LOGIN_ATTEMPTS 5
  PASSWORD_LIFE_TIME 60
  PASSWORD_REUSE_MAX 60
  PASSWORD_REUSE_MAX UNLIMITED
  PASSWORD_VERIFY_FUNCTION verify_function
  PASSWORD_LOCK_TIME 1
  PASSWORD_GRACE_TIME 10;
```

2. 次の文は、プロファイル prof でユーザー名と同じパスワードを持つユーザーを作ります。

```
CREATE USER userscott IDENTIFIED BY userscott PROFILE prof;
ORA-28003: Password verification for the specified password failed
ORA-20001: Password same as user
```

3. 次の文は、プロファイル prof により "scott1%" で識別されるユーザー userscott を作ります。

```
CREATE USER userscott IDENTIFIED BY "scott%" PROFILE prof;
```

4. 次の文は、ユーザーのパスワードを "scott%" にもう一度変更し、エラーを戻します。

```
ALTER USER userscott IDENTIFIED BY "scott%";
ORA-28007: The password cannot be reused
```

5. 次の文は、ユーザー・アカウントをロックします。

```
ALTER USER userscott ACCOUNT LOCK;
```

6. 次の文は、ユーザー・アカウントの状態をチェックします。

```
SELECT username, user_id, account_status, lock_date
FROM dba_users
WHERE username='USERSCOTT';
```

7. 次の文は、パスワードを期限切れにします。

```
ALTER USER userscott PASSWORD EXPIRE;
```

8. 次の文は、ユーザー・アカウントの状態をチェックします。

```
SELECT username, user_id, account_status, expiry_date
FROM dba_users
WHERE username='USERSCOTT';
```

9. 次の文は、ユーザーをアンロックします。

```
ALTER USER userscott ACCOUNT UNLOCK;
```

10. 次の文は、アカウントの状態をチェックします。

```
SELECT username, user_id, account_status, expiry_date
FROM dba_users
WHERE username='USERSCOTT';
```

ユーザー権限とロールの管理

この章では、権限およびロールを使って、システム操作を実行する機能と、スキーマ・オブジェクトへのアクセスを制御する方法について説明します。トピックは次のとおりです。

- ユーザー権限の識別
- ユーザー・ロールの管理
- ユーザー権限とロールの付与
- ユーザー権限とロールの取り消し
- オペレーティング・システムまたはネットワークを使ってロールを付与
- 権限とロールの情報を記述

関連項目：データベースへのアクセスの制御の詳細は、第 20 章を参照してください。

提案されている一般的なデータベース・セキュリティ方針の詳細は、第 19 章を参照してください。

この章には、Oracle Enterprise Manager について述べている箇所がいくつかあります。Enterprise Manager/GUI または Server Manager/LineMode を使って特定のタスクを実行する方法の詳細は、『Oracle Enterprise Manager 管理者ガイド』または『Oracle Server Manager ユーザーズ・ガイド』を参照してください。

ユーザー権限の識別

ここでは、Oracle ユーザー権限について説明します。トピックは次のとおりです。

- システム権限
- オブジェクト権限

ユーザー権限とは、特定のタイプの SQL 文を実行するための権利、または他のユーザーのオブジェクトへアクセスするための権利です。また、通常まとめて付与される権限または取り消される権限をグループ化するショート・カットも用意されています。

システム権限

システム権限は 80 種類以上あります。各システム権限を使うと、ユーザーは 1 つまたは複数の特定の データベース操作を実行できるようになります。表 21-1 に、すべてのシステム権限と各権限によって許可される操作をリストします。

セキュリティの理由により、システム権限ではユーザーからデータ・ディクショナリにアクセスできません。したがって、ANY 権限 (UPDATE ANY TABLE または SELECT ANY TABLE、CREATE ANY INDEX など) のユーザーは、PUBLIC に付与されていないディクショナリ表やビューにはアクセスできません。

警告： システム権限は非常に強力なので、この権限をデータベースのロールとトラステッド・ユーザーに付与する場合には十分に注意してください。ANY 権限を持つユーザーは、データ・ディクショナリにアクセスできません。

表 21-1 システム権限

システム権限	許可されている操作
ANALYZE	
ANALYZE ANY	データベース内の任意の表、クラスタ、索引を分析する。
AUDIT	
AUDIT ANY	データベース内のスキーマ・オブジェクトを監査する。
AUDIT SYSTEM	文監査オプションと権限監査オプションを使用可能または使用禁止にする。
CLUSTER	
CREATE CLUSTER	自分のスキーマ内にクラスタを作成する。
CREATE ANY CLUSTER	任意のスキーマ内にクラスタを作成する。これは CREATE ANY TABLE の実行結果と類似している。
ALTER ANY CLUSTER	データベース内の任意のクラスタを変更する。
DROP ANY CLUSTER	データベース内の任意のクラスタを削除する。

表 21-1 システム権限

システム権限	許可されている操作
DATABASE	
ALTER DATABASE	データベースを変更する。オペレーティング・システム権限には関係なく、Oracle を介してオペレーティング・システムにファイルを追加できる。
DATABASE LINK	
CREATE DATABASE LINK	自分のスキーマ内にプライベート・データベース・リンクを作成する。
INDEX	
CREATE ANY INDEX	任意の表に索引を任意のスキーマ内で作成する。
ALTER ANY INDEX	データベース内の任意の索引を変更する。
DROP ANY INDEX	データベース内の任意の索引を削除する。
LIBRARY	
CREATE LIBRARY	自分のスキーマ内にコールアウト・ライブラリを作成する。
CREATE ANY LIBRARY	任意のスキーマ内にコールアウト・ライブラリを作成する。
DROP LIBRARY	自分のスキーマ内のコールアウト・ライブラリを削除する。
DROP ANY LIBRARY	任意のスキーマ内のコールアウト・ライブラリを削除する。
PRIVILEGE	
GRANT ANY PRIVILEGE	任意のシステム権限を付与する（オブジェクト権限ではない）。
PROCEDURE	
CREATE PROCEDURE	自分のスキーマ内に格納するプロシージャ、ファンクション、パッケージを作成する。
CREATE ANY PROCEDURE	任意のスキーマ内に格納するプロシージャ、ファンクション、パッケージを作成する。（また、ユーザーは ALTER ANY TABLE、BACKUP ANY TABLE、DROP ANY TABLE、SELECT ANY TABLE、INSERT ANY TABLE、UPDATE ANY TABLE、DELETE ANY TABLE、GRANT ANY TABLE のいずれかの権限を持っている必要がある。）
ALTER ANY PROCEDURE	任意のスキーマ内に格納されている任意のプロシージャ、ファンクション、パッケージをコンパイルする。
DROP ANY PROCEDURE	任意のスキーマ内に格納されている任意のプロシージャ、ファンクション、パッケージを削除する。

表 21-1 システム権限

システム権限	許可されている操作
EXECUTE ANY PROCEDURE	任意のプロシージャやファンクション（スタンドアロン・プロシージャやパッケージ・ファンクション）を実行するか、または任意のスキーマ内の任意のパブリック・パッケージ変数を参照する。
PROFILE	
CREATE PROFILE	プロファイルを作成する。
ALTER PROFILE	データベース内の任意のプロファイルを変更する。
DROP PROFILE	データベース内の任意のプロファイルを削除する。
ALTER RESOURCE COST	すべてのユーザー・セッションで使われるリソースにコストを設定する。
PUBLIC DATABASE LINK	
CREATE PUBLIC DATABASE LINK	パブリック・データベース・リンクを作成する。
DROP PUBLIC DATABASE LINK	パブリック・データベース・リンクを削除する。
PUBLIC SYNONYM	
CREATE PUBLIC SYNONYM	パブリック・シノニムを作成する。
DROP PUBLIC SYNONYM	パブリック・シノニムを削除する。
ROLE	
CREATE ROLE	ロールを作成する。
ALTER ANY ROLE	データベース内の任意のロールを変更する。
DROP ANY ROLE	データベース内の任意のロールを削除する。
GRANT ANY ROLE	データベース内の任意のロールを付与する。
ROLLBACK SEGMENT	
CREATE ROLLBACK SEGMENT	ロールバック・セグメントを作成する。
ALTER ROLLBACK SEGMENT	ロールバック・セグメントを変更する。
DROP ROLLBACK SEGMENT	ロールバック・セグメントを削除する。
SESSION	
CREATE SESSION	データベースに接続する。
ALTER SESSION	ALTER SESSION 文を発行する。

表 21-1 システム権限

システム権限	許可されている操作
RESTRICTED SESSION	データベースが STARTUP RESTRICT を使って起動してから接続する。特殊な OSOPER と OSDBA の各ロールはこの権限を含んでいる。
SEQUENCE	
CREATE SEQUENCE	自分のスキーマ内に順序を作成する。
CREATE ANY SEQUENCE	任意のスキーマ内に順序を作成する。
ALTER ANY SEQUENCE	任意のスキーマ内の任意の順序を変更する。
DROP ANY SEQUENCE	任意のスキーマ内に任意の順序を削除する。
SELECT ANY SEQUENCE	任意のスキーマ内に任意の順序を参照する。
SNAPSHOT	
CREATE SNAPSHOT	自分のスキーマ内にスナップショットを作成する (CREATE TABLE 権限も持っていないといけない)。
CREATE SNAPSHOT	任意のスキーマ内にスナップショットを作成する (CREATE ANY TABLE 権限も持っていないといけない)。
ALTER SNAPSHOT	任意のスキーマ内に任意のスナップショットを変更する。
DROP ANY SNAPSHOT	任意のスキーマ内の任意のスナップショットを削除する。
SYNONYM	
CREATE SYNONYM	自分のスキーマ内にシノニムを作成する。
CREATE SYNONYM	任意のスキーマ内にシノニムを作成する。
DROP ANY SYNONYM	任意のスキーマ内の任意のシノニムを削除する。
SYSTEM	
ALTER SYSTEM	ALTER SYSTEM 文を発行する。
TABLE	
CREATE TABLE	自分のスキーマ内に表を作成する。また権限受領者は自分のスキーマ内に索引 (整合性制約のための索引も含める) も作れる。権限受領者はその表領域の割当て制限、また UNLIMITED TABLESPACE 権限を持っていないといけない。

表 21-1 システム権限

システム権限	許可されている操作
CREATE ANY TABLE	任意のスキーマ内に表を作成する。権限受領者が CREATE ANY TABLE 権限を持っていて別のユーザーのスキーマ内に表を作成する場合、その自分の割当て制限とデフォルト表領域が使われる。その表の所有者に CREATE [ANY] TABLE 権限は必要はない。
ALTER ANY TABLE	任意のスキーマ内に任意の表を変更した後、任意のスキーマ内に任意のビューをコンパイルする。
BACKUP ANY TABLE	任意のスキーマ内のエクスポート・ユーティリティを使って、増分エクスポートを実行する。
DROP ANY TABLE	任意のスキーマ内の任意の表を削除または切り捨てる。
LOCK ANY TABLE	任意のスキーマ内の任意の表またはビューをロックする。
COMMENT ANY TABLE	スキーマ内の任意の表、ビュー、列にコメントを作成する。
SELECT ANY TABLE	任意のスキーマ内の任意の表、ビュー、スナップショットを問い合わせる。
INSERT ANY TABLE	任意のスキーマ内の任意の表またはビューに行を挿入する。
UPDATE ANY TABLE	任意のスキーマ内の任意の表またはビューで行を更新する。
DELETE ANY TABLE	任意のスキーマの任意の表またはビューから行を削除する。
TABLESPACE	
CREATE TABLESPACE	表領域を作成する。つまり、ユーザーのオペレーティング・システム権限に関係なく、Oracle を介してオペレーティング・システムにファイルを追加する。
ALTER TABLESPACE	表領域を変更する。つまり、ユーザーのオペレーティング・システム権限に関係なく、Oracle を介してオペレーティング・システムにファイルを追加する。
MANAGE TABLESPACE	任意の表領域をオフライン、オンラインに切り替え、任意の表領域のバックアップの開始、終了する。
DROP TABLESPACE	表領域を削除する。

表 21-1 システム権限

システム権限	許可されている操作
UNLIMITED TABLESPACE	どの表領域でも無数に使用できます。この権限は、指定された割当て制限を上書きします。この権限を取り消すと、権限受領者のスキーマ・オブジェクトはそのまま残りますが、特定の表領域の割当て制限で許可されていないかぎり、それ以上の表領域の割当ては拒否されます。このシステム権限は、ロールに対してではなく、ユーザーだけに割り当てることができます。一般に、このシステム権限を付与せずに、特定の表領域の割当て制限が割り当てられます。
TRANSACTION	
FORCE TRANSACTION	ローカル・データベース内の自分のインダウト分散トランザクションを強制コミットまたはロールバックする。
FORCE ANY TRANSACTION	ローカル・データベース内の任意のインダウト分散トランザクションを強制コミットまたはロールバックする。
TRIGGER	
CREATE TRIGGER	自分のスキーマ内にトリガーを作成する。
CREATE ANY TRIGGER	任意のスキーマ内の任意の表に対応するトリガーを任意のスキーマ内に作成する。
ALTER ANY TRIGGER	任意のスキーマ内の任意のトリガーを使用可能、使用禁止、コンパイルする。
DROP ANY TRIGGER	任意のスキーマ内の任意のトリガーを削除する。
USER	
CREATE ANY USER	ユーザーを作成する。つまり、任意の表領域に割当て制限を割り当てる、デフォルト表領域や一時表領域を設定し、CREATE USER 文の一部としてプロファイルを割り当てる。
BECOME ANY USER	別のユーザーになる。(全データベース・インポートを実行する任意のユーザーに必要)。
ALTER USER	別のユーザーを変更する。つまり、ALTER USER 文で、ユーザーのパスワードまたは認証方法の変更、表領域の割当て制限の設定、デフォルト表領域および一時表領域の設定、プロファイルおよびデフォルト・ロールの割当てを実行する(自分のパスワードを変更する場合はこの機能は必要ない)。
DROP USER	別のユーザーを削除する。

表 21-1 システム権限

システム権限	許可されている操作
VIEW	
CREATE VIEW	自分のスキーマ内にビューを作成する。
CREATE ANY VIEW	任意のスキーマ内にビューを作成する。別のユーザーのスキーマにビューを作成するには、CREATE ANY VIEW 権限が必要であり、所有者はそのビューで参照されるオブジェクトに対する必要な権限がなければならない。
DROP ANY VIEW	任意のスキーマ内の任意のビューを削除する。

システム権限の制限

ディクショナリ保護のメカニズムによって、権限のないユーザーがディクショナリ・オブジェクトにアクセスできないようになっています。

ディクショナリ・オブジェクトへのアクセスは、ユーザー SYSDBA と SYSOPER に制限されています。他のスキーマのオブジェクトへのアクセスを提供するシステム権限があっても、ディクショナリ・オブジェクトにはアクセスできません。たとえば、SELECT ANY TABLE 権限によって他のスキーマのビューや表にはアクセスできますが、ディクショナリ・オブジェクト（実表、ビュー、パッケージ、シノニム）は選択できません。

また、SQL*Plus コマンド connect SYS/password で接続しようとしても失敗に終わります。しかし、次の 2 つの SQL*Plus コマンドは有効です。

```
connect SYS/password as SYSDBA
connect SYS/password as SYSOPER
```

Oracle7 の動作に戻すには、07_DICTIONARY_ACCESSIBILITY パラメータ（デフォルトは TRUE）を使います（そしてシステム権限の制限を削除します）。

関連項目：07_DICTIONARY_ACCESSIBILITY パラメータの詳細は、『Oracle8 Server リファレンス・マニュアル』を参照してください。

頻繁に使用するディクショナリにアクセスする

明示的なオブジェクト権限のあるユーザーと SYSDBA は、ディクショナリ・オブジェクトにアクセスできます。しかし、ディクショナリ・オブジェクトにアクセスする必要があるのに明示的なオブジェクト権限がない場合は、次のロールを付与されるようにできます。

- SELECT_CATALOG_ROLE

ユーザーは、このロールに付与されたエクスポート済みのすべてのカタログ・ビューと表を SELECT で選択できます。このロールを、データ・ディクショナリのすべてのエクスポート済みビューと表にアクセスする必要があるユーザーに付与してください。

- EXECUTE_CATALOG_ROLE

ディクショナリのエクスポート済みパッケージに対する EXECUTE 権限を提供します。

■ DELETE_CATALOG_ROLE

ユーザーは、AUD\$ 表からレコードを削除できます。

これらのロールによって、データベース管理者は、ディクショナリのセキュリティをメンテナンスするときにディクショナリ中の特定のオブジェクトにアクセスできます。

注意： SYSDBA は、ディクショナリ中のエクスポートされていないオブジェクトに関しては、どのユーザーにもオブジェクト権限を付与すべきではありません。そのようにすると、データベースの整合性が失われることがあります。

関連項目： エクスポート済みの表またはビューの詳細は、『Oracle8 Server リファレンス・マニュアル』を参照してください。

オブジェクト権限

オブジェクトの種類ごとに異なるオブジェクト権限が対応付けられます。表 21-2 に、各種オブジェクトに対して使用可能なオブジェクト権限を記述します。

表 21-2 オブジェクト権限

オブジェクト権限	表	ビュー	順序	プロシージャ (a)
ALTER	(1)		(1)	
DELETE	(1)	(1)		
EXECUTE				(1)
INDEX	✓(2)			
INSERT	(1)	(1)		
REFERENCES	✓ (2)			
SELECT	(1)	✓ (2)	(1)	
UPDATE	(1)	(1)		
1. スナップショットについても付与されます。 2. ロールに対する権限の付与はできません。 ✓ スタンドアロンのストアド・プロシージャとストアド・ファンクション、パブリック・パッケージ構造体も含まれます。				

なお、スキーマ・オブジェクトは、表 21-2 に示したもの以外にも存在しています。ここに示されていないスキーマ・オブジェクトの多く（たとえば、クラスタ、索引、トリガー、

データベース・リンク) は、システム権限を使って排他的に制御されます。たとえば、クラスタを変更するためには、ユーザーはそのクラスタを所有しているか、または ALTER ANY CLUSTER システム権限を持っていないければなりません。

表 21-3 は、表 21-2 に示したオブジェクト権限で許されている SQL 文をリストしたものです。

表 21-3 オブジェクト権限によって許可される SQL 文

オブジェクト権限	許可されている SQL 文
ALTER	ALTER オブジェクト (表または順序)
DELETE	DELETE FROM オブジェクト (表またはビュー)
EXECUTE	EXECUTE オブジェクト (プロシージャまたはファンクション)、パブリック・パッケージ変数への参照
INDEX	CREATE INDEX ON オブジェクト (表だけ)
INSERT	INSERT INTO オブジェクト (表またはビュー)
REFERENCES	オブジェクトに関する FOREIGN KEY 整合性制約を定義した CREATE 文または ALTER TABLE 文
SELECT	SELECT...FROM オブジェクト (表またはビュー、スナップショット)。順序を使った SQL 文
UPDATE	UPDATE オブジェクト (表またはビュー)

オブジェクト権限のショートカット

ALL および ALL PRIVILEGES ショートカットを使うと、あるオブジェクトに対して使用可能なオブジェクト権限がすべて付与されるか、またはすべて取り消されます。このショートカットは権限ではなく、GRANT 文および REVOKE 文の中の一語ですべてのオブジェクト権限を付与または取り消すための手段です。ALL を使ってすべてのオブジェクト権限を付与した場合にも、個々に権限を取り消せます。

同様に、ALL を使って、個々に付与した権限をすべて取り消すこともできます。ただし、REVOKE ALL を使い、その取消しによって整合性制約が削除される場合 (整合性制約は取り消そうとしている REFERENCES 権限に依存しているため) REVOKE 文に CASCADE CONSTRAINTS オプションを指定しなければなりません。

ユーザー・ロールの管理

ここでは、ロールの管理について説明します。トピックは次のとおりです。

- ロールを作成する
- 事前定義済みのロール

ロールは複数の権限やロールをまとめて指定しますので、ユーザーに対して同時に権限を付与したりその設定を取り消したりできます。ロールはユーザーごとに使用可能にしたり、使用禁止にしたりできます。

関連項目：ロールの詳細は、『Oracle8 Server 概要』を参照してください。

ロールを作成する

Enterprise Manager の「プロファイル作成」プロパティ・シートまたは SQL コマンド CREATE ROLE のどちらかを使って、プロファイルを作成できます。

ロールを作成するには、CREATE ROLE システム権限が必要になります。通常、セキュリティ管理者だけがこのシステム権限を持っています。

注意： ロールを作成した直後には、ロールに対応付けられている権限はありません。新しいロールに対応付けるには、新しいロールに権限や他のロールを付与しなければなりません。

次の文は、パスワード BICENTENNIAL を使うことでデータベースによって認可される CLERK ロールを作成します。

```
CREATE ROLE clerk  
IDENTIFIED BY bicentennial;
```

ロール名

作成する各ロールには、データベースの既存のユーザー名やロール名とは異なる、一意の名前を与えなければなりません。ロールはどのユーザーのスキーマ内にも指定できません。

マルチバイト・キャラクタ・セットのロール名

オラクル社は、マルチバイト・キャラクタ・セットを使うデータベースでは、各ロール名にシングルバイト・キャラクタを少なくとも 1 つ含めることをお勧めします。ロール名がマルチバイト・キャラクタしか含まない場合、暗号化されたロール名 / パスワードの組み合わせの安全性はかなり損なわれます。

事前定義済みのロール

表 21-4 にリストしたロールは、Oracle データベースに対して自動的に定義されます。これらのロールによって、以前のバージョンの Oracle との下位互換性が維持されます。これらの事前定義済みのロールに、ロールに定義することのできる権限とロールを付与し、取り消します。

表 21-4 事前定義済みロール

ロール名	ロールに付与されている権限
CONNECT ¹	ALTER SESSION、CREATE CLUSTER、CREATE DATABASE LINK、CREATE SEQUENCE、CREATE SESSION、CREATE SYNONYM、CREATE TABLE、CREATE VIEW
CREATE TYPE ⁷	CREATE TYPE、EXECUTE、EXECUTE ANY TYPE、ADMIN OPTION、GRANT OPTION
RESOURCE ^{1、2}	CREATE CLUSTER、CREATE PROCEDURE、CREATE SEQUENCE、CREATE TABLE、CREATE TRIGGER
DBA ^{1、3、4}	ADMIN OPTION 付きのすべてのシステム権限
EXP_FULL_DATABASE ⁵	表 SYS.INCVID、SYS.INCFIL、SYS.INCEXP に対する SELECT ANY TABLE、BACKUP ANY TABLE、INSERT、DELETE、UPDATE
IMP_FULL_DATABASE ⁵	BECOME USER
DELETE_CATALOG_ROLE ⁶	このロールに関するすべてのディクショナリ・パッケージに対する DELETE 権限
EXECUTE_CATALOG_ROLE ⁶	このロールに関するすべてのディクショナリ・パッケージに対する EXECUTE 権限
SELECT_CATALOG_ROLE ⁶	このロールに関するすべてのカタログ表およびビューに対する SELECT 権限

表 21-4 事前定義済みロール

ロール名	ロールに付与されている権限
¹ SQL.BSQ	によって作られます。
² RESOURCE	ロールの権限受領者は、UNLIMITED TABLESPACE システム権限を（RESOURCE ロールの一部としてではなく）明示的に付与されるものとして受領します。
³ DBA	ロールの権限受領者は、ADMIN OPTION 付きの UNLIMITED TABLESPACE システム権限を（DBA ロールの一部としてではなく）明示的に付与されるものとして受領します。したがって、DBA ロールが取り消されると、明示的に付与された UNLIMITED TABLESPACE もすべて取り消されます。
⁴ CATEXP.SQL	が実行されている場合は、EXP_FULL_DATABASE ロールと IMP_FULL_DATABASE ロールも含まれます。
⁵ CATEXP.SQL	によって作られます。
⁶ DBA	ロールはないが、データ・ディクショナリ中のビューと表へのアクセスが必要なユーザーには、このロールを付与しなければなりません。
⁷ Oracle	オブジェクトのオプションがユーザーのデータベース・サーバーにインストールされている場合だけ、CREATE TYPE コマンドを使用できます。

ロールの認可

ユーザーがデータベース・ロールを使用可能にすると、そのロールはオプションで認可も要求できます。ロールの認可は、データベース（パスワードを使う）またはオペレーティング・システム、ネットワーク・サービスによって保守されます。

ロールの認可方法を変更するには、ALTER ANY ROLE システム権限を持っているか、または ADMIN OPTION 付きのロールが付与されていなければなりません。

関連項目：ネットワーク・ロールの詳細は、『Oracle8 Server 分散システム』を参照してください。

データベースによるロールの認可

ロールの使用は、このロールに対応付けられているパスワードによって保護されます。パスワードによって保護付きのロールが付与される場合、そのロールの適切なパスワードを SET ROLE コマンドに指定した場合だけ、ロールを使用可能、または使用禁止にできます。

注意： マルチバイト・キャラクタ・セットを使うデータベースでは、ロールのパスワードにシングルバイト・キャラクタだけを使ってください。パスワードでは、マルチバイト・キャラクタは受け入れられません。

関連項目：有効なパスワードの詳細は、『Oracle8 Server リファレンス・マニュアル』を参照してください。

オペレーティング・システムによるロールの認可

次の文は、ACCTS_REC という名前のロールを作成します。このロールを使うための認可をオペレーティング・システムから受ける必要があります。

```
CREATE ROLE role IDENTIFIED EXTERNALLY;
```

オペレーティング・システムによるロールの認可は、オペレーティング・システムがオペレーティング・システムの権限をアプリケーションと動的にリンク可能なときだけ有効です。ユーザーがアプリケーションを開始すると、オペレーティング・システムはオペレーティング・システム権限をそのユーザーに付与します。付与されたオペレーティング・システム権限は、アプリケーションに対応付けられたロールと一致します。この時点で、アプリケーションはアプリケーション・ロールを使用可能にできます。アプリケーションが終了すると、先に付与されたオペレーティング・システム権限は、そのユーザーのオペレーティング・システム・アカウントから取り消されます。

ロールがオペレーティング・システムによって認可される場合には、オペレーティング・システム・レベルで各ユーザーの情報を構成しなければなりません。この操作は、オペレーティング・システムによって異なります。

ロールがオペレーティング・システムによって付与される場合、オペレーティング・システムにその認可を求める必要はありません。それは冗長な操作になります。

関連項目：オペレーティング・システムによって付与されるロールの詳細は、21-22 ページの「オペレーティング・システムまたはネットワークを使ってロールを付与」を参照してください。

ロールの認可とネットワーク・クライアント

ユーザーが SQL*Net を介してデータベースに接続する場合、デフォルトでは、ユーザーのロールはオペレーティング・システムによって認可できません。マルチスレッド・サーバーを介する接続には SQL*Net が必要であるためです。リモート・ユーザーがネットワーク接続を介して別のオペレーティング・システム・ユーザーのふりをする可能性があるため、この制限がデフォルトになっています。

このようなセキュリティの危険性の心配がなく、ネットワーク・クライアントに対してオペレーティング・システムによるロール認証を使う場合は、データベースのパラメータ・ファイルにあるパラメータ REMOTE_OS_ROLES を TRUE に設定してください。この変更は、次回インスタンスを起動し、データベースをマウントするときに有効となります。(パラメータのデフォルト設定は FALSE です。)

認可を保留する

認可を指定しないでロールを作成することもできます。どのような保護も設定しないでロールを作成すると、どの権限受領者でもそのロールを使用可能にしたり、使用禁止にしたりできます。

ロールの認可を変更する

Enterprise Manager/GUI の「ロール変更」プロパティ・シート、または SQL コマンド ALTER ROLE を使って、ロールの許可方法を設定、変更できます。

次の文は、CLERK ロールを外部的に認可されるように変更します。

```
ALTER ROLE clerk  
IDENTIFIED EXTERNALLY;
```

ユーザーのデフォルト・ロールを変更する

ユーザーのデフォルト・ロールのリストを設定および変更するには、Enterprise Manager の「ユーザー変更」ダイアログ・ボックス、または SQL コマンド ALTER USER を使えます。

関連項目：これらのオプションの詳細は、20-14 ページの「ユーザーを変更する」を参照してください。

ALL キーワードを使う ユーザーのデフォルト・ロールのリストを ALL に指定すると、ユーザーに付与されているロールがすべて自動的にユーザーのデフォルト・ロールのリストに追加されます。その後ユーザーのデフォルト・ロールを変更するだけで、新たに付与されたロールをデフォルトのロールのユーザーのリストから取り消すことができます。

MAX_ENABLED_ROLES パラメータを使う MAX_ENABLED_ROLES に指定したロール数だけロールを使えます。最初のロールが使用可能になった結果、使用可能となる間接的に付与されるロールもすべてその数に含まれます。データベース管理者はこのパラメータの値を修正することによって、この制限を変更できます。同時に使用可能にするロール数を増やすには、大きな値を各ユーザー・セッションに設定します。ただし、このパラメータの値が大きくなると、ユーザー・セッションごとに大量のメモリー領域が必要になります。これは、ユーザー・セッションごとに PGA サイズが影響を受け、ロールあたり 4 バイト必要となるためです。1 人のユーザーが同時に使用可能にできるロールの最大数を決定し、その値を MAX_ENABLED_ROLES パラメータに使用してください。

ロールを削除する

状況によっては、ロールをデータベースから削除した方が適切な場合があります。削除したロールを付与されていたユーザーとロールのセキュリティ定義域は、削除したロール権限がなくなったことを反映するためただちに更新されます。削除したロールによって間接的に付与されていたロールも、影響を受けていたセキュリティ定義域から削除されます。ロールを削除することによって、すべてのユーザーのデフォルト・ロール・リストからそのロールは自動的に削除されます。

オブジェクトを作成することはロールを介して受け取った権限に依存していませんので、ロールが削除されても、表や他のオブジェクトは削除されません。

ロールを削除するには、DROP ANY ROLE システム権限を持っているか、または ADMIN OPTION 付きのロールが付与されていなければなりません。

Enterprise Manager の削除メニュー項目または SQL コマンド DROP ROLE を使って、ロールを削除できます。

次の文では、CLERK が削除されます。

```
DROP ROLE clerk;
```

ユーザー権限とロールの付与

ここでは、権限とロールの付与について説明します。トピックは次のとおりです。

- システム権限とロールを付与する
- オブジェクト権限とロールを付与する
- 列に権限を付与する

システム権限とロールを付与する

システム権限とロールを他のロールおよびユーザーに付与するには、Enterprise Manager の「システム権限およびロールを付与する」ダイアログ・ボックスか、または SQL コマンド GRANT を使えます。

システム権限またはロールを付与するには、付与するすべてのシステム権限およびロールの ADMIN OPTION が必要です。また、GRANT ANY ROLE システム権限を持っているユーザーは、データベース内で任意のロールを付与できます。

次の文は、システム権限および ACCTS_PAY ロールをユーザー JWARD に付与します。

```
GRANT create session, accts_pay  
TO jward;
```

注意： オブジェクト権限は、同じ GRANT 文でシステム権限およびロールといっしょに付与できません。

ADMIN オプション

ユーザーがロールを作成すると、そのロールは自動的に ADMIN OPTION 付きでその作成ユーザーに付与されます。ADMIN オプションを持つ権限受領者は、権限の有効範囲が次のように拡大されます。

- 権限受領者は、データベース内の任意のユーザーまたは他のロールに対するシステム権限またはロールを付与したり取り消すことができます。（ただし、ユーザーは自分自身からロールを取り消すことはできない。）
- 権限受領者は、さらに ADMIN OPTION 付きのシステム権限やロールを付与できる。
- ロールの権限受領者は、そのロールを変更し、削除できる。

次の文では、セキュリティ管理者が NEW_DBA ロールを MICHAEL に付与します。

```
GRANT new_dba TO michael WITH ADMIN OPTION;
```

ユーザーの MICHAEL は、NEW_DBA ロール内の権限をすべて暗黙に使えるだけでなく、必要に応じて NEW_DBA ロールを付与し、取り消し、削除できます。このように ADMIN OPTION は非常に強力な機能であるため、これを使ってシステム権限やロールを付与する際には、十分に注意してください。通常、このような権限はセキュリティ管理者用に確保されているため、システム内の他の管理者やユーザーに付与されることはほとんどありません。

オブジェクト権限とロールを付与する

オブジェクト権限をロールとユーザーに付与するには、Enterprise Manager の「ロール / ユーザーに権限を追加する」ダイアログ・ボックスまたは SQL コマンド GRANT を使います。

オブジェクト権限を付与するには、次のどちらかの条件を満たしていなければなりません。

- 指定するオブジェクトを所有している。
- GRANT OPTION で付与しているオブジェクト権限を付与されている。

次の文では、EMP 表のすべての列に対する SELECT および INSERT、DELETE のオブジェクト権限がユーザー JFEE と TSMITH に付与されます。

```
GRANT select, insert, delete ON emp TO jfee, tsmith;
```

ユーザー JFEE と TSMITH に、EMP 表の ENAME 列と JOB 列の INSERT オブジェクト権限だけを付与するには、次のように入力します。

```
GRANT insert(ename, job) ON emp TO jfee, tsmith;
```

SALARY ビューのすべてのオブジェクト権限をユーザー JFEE に付与するには、次の例のように ALL ショートカットを使います。

```
GRANT ALL ON salary TO jfee;
```

注意： 同じ GRANT 文で、オブジェクト権限とともにシステム権限とロールを付与できません。

GRANT OPTION

スキーマにオブジェクトを含んでいるユーザーには、GRANT OPTION に対応するすべてのオブジェクト権限が付与されます。この特別な権限によって、権限受領者の権限が拡張されます。

- 権限受領者は、データベース内の任意のユーザーや任意のロールにオブジェクト権限を付与できる。

- 権限受領者は、GRANT OPTION を付ける、または付けなくて、他のユーザーにオブジェクト権限を付与することもできる。
- 権限受領者が GRANT OPTION 付きで表のオブジェクト権限を受け取り、さらにシステム権限 CREATE VIEW または CREATE ANY VIEW を持っている場合、この権限受領者はその表に対してビューを作成し、データベース内の任意のユーザーとロールにそのビューの対応する権限を付与できる。

オブジェクト権限をロールに付与する場合には、GRANT OPTION は無効です。Oracle はロールによるオブジェクト権限の波及を未然に防ぎますので、ロールの権限受領者はロールを介して受領したオブジェクト権限を反映できません。

列に権限を付与する

表の個々の列に INSERT、UPDATE、REFERENCES 権限を付与できます。

警告： 列固有の INSERT 権限を付与する前に、NOT NULL 制約が定義されている列が表に含まれているかどうかを判断します。NOT NULL 列を含めずに選択挿入機能を付与すると、ユーザーは行を表に挿入できません。このような状況を回避するために、各 NOT NULL 列が挿入可能であるか、NULL 以外のデフォルト値を持っているかを確認してください。それ以外は、権限受領者は行を表に挿入できず、エラーが発生します。

ACCOUNTS 表の ACCT_NO 列の INSERT 権限を SCOTT に付与します。

```
GRANT INSERT (acct_no)
ON accounts TO scott;
```

ユーザー権限とロールの取り消し

ここでは、ユーザー権限とロールの取消しについて説明します。トピックは次のとおりです。

- システム権限とロールを取り消す
- オブジェクト権限とロールを取り消す

システム権限とロールを取り消す

システム権限とロールを取り消すには、Enterprise Manager の「システム権限およびロールを取り消す」ダイアログ・ボックスまたは SQL コマンド REVOKE を使います。

システム権限またはロールの ADMIN OPTION を持っているユーザーは、他のデータベース・ユーザーやロールから権限またはロールを取り消せます。取消しユーザーは、その権限またはロールを最初に付与したユーザーである必要はありません。また、GRANT ANY ROLE を持っているユーザーは、任意のロールを取り消すことができます。

次の文は、TSMITH から CREATE TABLE システム権限と ACCTS_REC ロールを取り消します。

```
REVOKE create table, accts_rec FROM tsmith;
```

注意： システム権限またはロールの ADMIN OPTION は、選択的に取り消せません。権限またはロールを取り消してから、ADMIN OPTION を指定せずにその権限またはロールを再度付与してください。

オブジェクト権限とロールを取り消す

オブジェクト権限は、Enterprise Manager または SQL コマンド REVOKE を使って取り消せます。

オブジェクト権限を取り消すユーザーは、取消しの対象となるオブジェクト権限の権限付与者でなければなりません。

たとえば、ユーザー JFEE と TSMITH から EMP 表の INSERT 権限を取り消す場合は、この権限付与者自身が次の文を発行します。

```
REVOKE select, insert ON emp  
FROM jfee, tsmith;
```

次の文では、DEPT 表からすべての権限（HUMAN_RESOURCE ロールに最初に付与された権限すべて）が取り消されます。

```
REVOKE ALL ON dept FROM human_resources;
```

注意： この文は、他のユーザーが行った付与を取り消すのではなく、権限付与者が許可した権限だけを取り消します。オブジェクト権限の GRANT OPTION は、選択的に取り消せません。オブジェクト権限を取り消してから、GRANT OPTION を使わずに再度付与してください。ユーザーは、自分自身からオブジェクト権限を取り消せません。

列に選択的に付与されていたオブジェクト権限を取り消す

INSERT、UPDATE、REFERENCES 権限を表やビューの列に選択的に付与できますが、それに類似する REVOKE 文を使って、列固有の権限を選択的に取り消すことはできません。かわりに、権限付与者は表またはビューの全列に対するオブジェクト権限を最初に取り消し、さらに残しておきたい列固有の権限を選択的に再付与しなければなりません。

たとえば、HUMAN_RESOURCES ロールに、DEPT 表の DEPTNO 列および DNAME 列に対する UPDATE 権限が付与されているとします。その UPDATE 権限を DEPT 列だけから取り消すためには、次の 2 つの文を入力します。

```
REVOKE UPDATE ON dept FROM human_resources;
```

```
GRANT UPDATE (dname) ON dept TO human_resources;
```

REVOKE 文が HUMAN_RESOURCES ロールから DEPT 表の全列に対する UPDATE 権限を取り消します。GRANT 文で、DNAME 列の UPDATE 権限を HUMAN_RESOURCE ロールに再度付与します。

REFERENCES オブジェクト権限を取り消す

REFERENCES オブジェクト権限の権限受領者が外部キーの制約（現行の制約）を設定するための権限を使っている場合、権限付与者は REVOKE 文に CASCADE CONSTRAINTS オプションを指定することによって、唯一その権限を取り消せます。

```
REVOKE REFERENCES ON dept FROM jward CASCADE CONSTRAINTS;
```

CASCADE CONSTRAINTS オプションを指定すると、取消し REFERENCES 権限を使って、現在定義されている外部キーの制約が削除されます。

権限の取消しによる影響

権限のタイプによっては、権限を取り消す際に連鎖的な影響が生じる場合があります。

システム権限

DDL 操作に関連したシステム権限を取り消す際には、カスケードする影響はまったくありません。このとき、ADMIN OPTION の使用や、権限付与の有無にはまったく関係ありません。ここでは、次のような条件を想定します。

1. セキュリティ管理者は、ADMIN OPTION を使って、JFEE に対して CREATE TABLE システム権限を付与する。
2. JFEE は表を作成する。
3. JFEE は、TSMITH に CREATE TABLE システム権限を付与する。
4. TSMITH は表を作成する。
5. セキュリティ管理者は、JFEE から CREATE TABLE システム権限を取り消す。
6. JFEE の表は引き続き存在する。TSMITH は引き続き表と CREATE TABLE システム権限を持っています。

カスケードする影響は、DML 操作に関連するシステム権限の削除の際に発生しています。たとえば、SELECT ANY TABLE をユーザーに付与する際に、そのユーザーがプロシージャを事前に作ってある場合、ユーザーのスキーマに含まれているすべてのプロシージャは、再度使用可能となる前に必ず許可し直さなければなりません。

オブジェクト権限

オブジェクト権限を取り消すと、それに付随して連鎖的な影響が発生しますので、REVOKE 文を指定する前に必ず十分に検討してください。

- DML オブジェクト権限を取り消すと、その DML オブジェクト権限に依存するオブジェクト定義にまで影響を及ぼす可能性がある。たとえば、TEST プロシージャの中に、EMP 表のデータを問合せる SQL 文を指定したとします。EMP 表の SELECT 権限が TEST プロシージャの所有者から削除されると、それ以降そのプロシージャを正常に実行できません。
- ALTER または INDEX オブジェクト権限を取り消しても、ALTER と INDEX DDL の各オブジェクト権限が必要となるオブジェクト定義には影響しない。たとえば、別のユーザーの表に索引を作ったユーザーから、INDEX 権限を取り消しても、その索引は権限が取り消された後も存在します。
- 表に対する REFERENCES 権限をユーザーから取り消すと、その取り消された REFERENCES 権限を必要とするユーザーが定義した外部キーの整合性制約が自動的に削除される。たとえば、ユーザーの JWARD には DEPT 表の DEPTNO 列に対する REFERENCES 権限が付与されているときに、その DEPTNO 列を参照する EMP 表の DEPTNO 列上で外部キーを作成するとします。EMP 表の DEPTNO 列の REFERENCES 権限を取り消すと、その EMP 表の DEPTNO 列上の外部キーの制約まで、同一操作上で削除されます。
- GRANT OPTION を使ったことの影響としてのオブジェクト権限付与は、権限付与者のオブジェクト権限が取り消されると、取り消される。たとえば、GRANT OPTION を使って、SELECT オブジェクト権限が USER1 に付与されているときに、USER2 に対し EMP 表の SELECT 権限を付与するとします。次に SELECT 権限は USER1 から取り消されます。この取消しは USER2 にも同様に適用されます。USER1 と USER2 の取り消された SELECT 権限に従属するオブジェクトは、前述のような影響を受ける可能性もあります（前の説明箇所を参照）。

ユーザー・グループ PUBLIC に対する付与と取消し

ユーザー・グループ PUBLIC に対して、権限とロールの付与と取消しができます。PUBLIC はすべてのデータベース・ユーザーにアクセスできるため、PUBLIC に対して付与された権限やロールには、すべてのデータベース・ユーザーがアクセスできます。

セキュリティ管理者とデータベース・ユーザーは、すべてのデータベース・ユーザーが権限またはロールを必要としている場合だけ、権限またはロールを PUBLIC に付与してください。これにより、いつでも各データベース・ユーザーは、現行作業を正常に実行するために必要な権限だけを持つという一般的な規則が保証されます。

PUBLIC から権限を取り消すと、カスケードする深刻な影響が発生する可能性があります。DML 操作に関連する権限を PUBLIC から取り消す場合（たとえば、SELECT ANY TABLE、UPDATE ON emp など）、ファンクションとパッケージを含むデータベース内のすべてのプロシージャは、再度使う前に「再認可する」必要があります。したがって、DML に関連する権限を PUBLIC に付与するときには注意が必要です。

関連項目：オブジェクト依存性の詳細は、17-22 ページの「オブジェクトの依存性の管理」を参照してください。

権限付与とその取消しはいつ実施されるか

権限付与とその取消しの処理内容に応じて、それぞれの処理の実施時期は異なります。

- 任意の対象（ユーザー、ロール、PUBLIC）に対するシステム権限とオブジェクト権限のすべての付与 / 取消しは、即座に実施される。
- 任意の対象（ユーザー、ロール、PUBLIC）に対する、ロールのすべての付与 / 取消しは、現在のセッション・ユーザーがその付与 / 取消しを実施した後でロールを再度使えるように SET ROLE 文を発行するとき、またはその付与 / 取消しを実施した後で新規ユーザー・セッションが作られるときに限り、実施される。

オペレーティング・システムまたはネットワークを使ってロールを付与

ここでは、オペレーティング・システムまたはネットワークを使ったロールの付与について説明します。トピックは次のとおりです。

- オペレーティング・システムのロール識別機能を使用する
- オペレーティング・システムのロール管理機能を使用する
- OS_ROLES=TRUE の場合にロールを付与する、取り消す
- OS_ROLES=TRUE の場合にロールを使用可能、使用禁止にする
- オペレーティング・システムによるロール管理でネットワーク接続を使用する

セキュリティ管理者が GRANT 文と REVOKE 文を使って、ユーザーに対し明示的にデータベース・ロールを付与するかわりに、Oracle を操作するオペレーティング・システムを基にして、接続時にユーザーに対しロールを付与できます。このため、オペレーティング・システムを使ってロールを管理し、ユーザーのセッション作成時にそのロールを Oracle に対して渡せます。その一部として、各ユーザーのデフォルト・ロールや、ADMIN OPTION を使ってユーザーに対して付与したロールが識別できます。たとえオペレーティング・システムを使ってロールに対するユーザーを許可する場合でも、必ずすべてのロールをデータベース内に作るとともに、GRANT 文を使ってそのロールに対し権限を割り当てなければなりません。

ロールはネットワーク・サービスを介しても付与できます。ネットワーク・ロールの詳細は、『Oracle8 Server 分散システム』を参照してください。

オペレーティング・システムを使って、ユーザーのデータベース・ロールを識別する際に得られる利点は、Oracle データベースに対する権限管理を外部的に実施できることです。つまり、オペレーティング・システムによって提供されるセキュリティ機能がユーザーの権限を制御します。さらに、このオプションを使えば、大量のシステム・アクティビティに対するセキュリティの集中管理による各種利点が生れます。たとえば、MVS Oracle の管理者は RACF グループを、UNIX Oracle の管理者は UNIX グループを、また VMS Oracle の管理者は権限の識別子をそれぞれ使ってデータベース・ユーザーのロールを識別させることができます。

ユーザーのデータベース・ロールを識別するためにオペレーティング・システムを使うことの主な欠点は、権限管理がロール・レベルでしか実行できないことです。つまり、個々の権限は、オペレーティング・システムを使っては付与できませんが、GRANT 文を使ってデータベース内で付与できます。

この機能を使う際の第 2 の欠点は、オペレーティング・システムがロールを管理している場合に、デフォルトではユーザーがマルチスレッド・サーバーまたはその他のネットワーク接続を介してデータベースに接続できないということです。ただし、このデフォルトを変更することができます。21-25 ページの「オペレーティング・システムによるロール管理でネットワーク接続を使用する」を参照してください。

関連項目：ここで説明した機能は、一部のオペレーティング・システムでしか使えません。この情報は、オペレーティング・システムによって異なります。オペレーティング・システム固有の Oracle のマニュアルを参照してください。

オペレーティング・システムのロール識別機能を使用する

オペレーティング・システムを使って、セッションの作成時に各ユーザーのデータベース・ロールを識別できるように、データベースを運用するために、初期化パラメータ OS_ROLES を TRUE に設定してください（そして、インスタンスが実行中であれば、再起動してください）。ユーザーがセッションを作成しようとすると、Oracle はオペレーティング・システムによって識別されるデータベース・ロールを使って、そのユーザーのセキュリティ定義域を初期化します。

ユーザーに対するデータベース・ロールを識別するために、各 Oracle ユーザーのオペレーティング・システム・アカウントは、そのユーザーが使えるデータベース・ロールを示すオペレーティング・システム識別子を必ず持っていなければなりません（この識別子とは、グループ、権限識別子、その他の類似する名前です）。ロールの指定により、ユーザーのデフォルトのロールであるのか、ADMIN OPTION を使用可能なロールであるのかということまで把握できます。使っているオペレーティング・システムには関係なく、オペレーティング・システム・レベルでのロールの指定形式は次のとおりです。

ORA_<ID>_<ROLE>[_[D][A]]

ここで

ID

ID の定義はオペレーティング・システムによって異なります。たとえば、VMS 上では、ID はデータベースのインスタンス識別子です。MVS 上では、ID はマシン・タイプです。

D

このオプション文字は、このロールがデータベース・ユーザーのデフォルト・ロールであることを示します。

A

このオプション文字は、ロールが ADMIN OPTION 付きでユーザーに付与されることを示します。これによって、ユーザーはこのロールを他のロールにだけ付与できます（オペレーティング・システムを使ってロールを管理している場合は、ロールをユーザーに付与できません）。

注意： 文字 D または A のどちらかを指定する場合には、アンダースコアをその文字の直前に指定しなければなりません。

たとえば、オペレーティング・システム・アカウントは、プロファイルで識別される次のロールを持っている可能性があります。

```
ORA_PAYROLL_ROLE1
ORA_PAYROLL_ROLE2_A
ORA_PAYROLL_ROLE3_D
ORA_PAYROLL_ROLE4_DA
```

対応するユーザーが Oracle の PAYROLL インスタンスに接続するとき、ROLE3 と ROLE4 はデフォルトであり、ROLE2 と ROLE4 は ADMIN OPTION で適用できます。

オペレーティング・システムのロール管理機能を使用する

オペレーティング・システムによって管理されているロールを使う場合は、データベース・ロールがオペレーティング・システムのユーザーに付与されるということに注意してください。OS ユーザーが接続できるデータベース・ユーザーは、認可されたデータベース・ロールを使えるようになります。このため、OS_ROLES = TRUE を使っている場合は、権限が付与されている OS アカウントにデータベース・アカウントを対応付けるために、すべての Oracle ユーザーを IDENTIFIED EXTERNALLY として定義することを考慮してください。

OS_ROLES=TRUE の場合にロールを付与する、取り消す

OS_ROLES が TRUE に設定されている場合には、オペレーティング・システムはユーザーに対するロールの付与と取消しを完全に管理しています。以前になされた GRANT 文によるユーザーへのロールの付与は適用されません。ただし、それらはデータ・ディクショナリには記述されたままです。オペレーティング・システム・レベルでのユーザーへのロールの付与だけが適用されます。ユーザーは権限をロールとユーザーに付与できます。

注意： オペレーティング・システムによって ADMIN OPTION 付きでロールが付与された場合、ユーザーはそのロールを他のロールに限り付与できます。

OS_ROLES=TRUE の場合にロールを使用可能、使用禁止にする

OS_ROLES が TRUE に設定されている場合、オペレーティング・システムによって付与されたロールは、SET ROLE コマンドを使って動的に使用可能にできます。ロールがパスワードやオペレーティング・システム認証を必要とするように定義されていた場合、それは適用されます。しかし、OS_ROLES = FALSE のときに GRANT 文を使ってロールが付与されていても、ユーザーのオペレーティング・システム・アカウントで識別されないロールは、SET ROLE 文に指定できません（このようなロールを指定しても Oracle では無視されます）。

OS_ROLES = TRUE のとき、ユーザーはパラメータ MAX_ENABLED_ROLES で指定されている数のロールを使用可能にできます。

オペレーティング・システムによるロール管理でネットワーク接続を使用する

オペレーティング・システムにロールを管理させる場合、デフォルトでは、ユーザーはマルチスレッド・サーバーを通じてデータベースに接続できません。リモート・ユーザーは安全性に欠けた接続を介して別のオペレーティング・システム・ユーザーのふりをする可能性があるため、この制限がデフォルトになっています。

セキュリティの危険性の心配がなく、マルチスレッド・サーバーまたはその他のネットワーク接続でオペレーティング・システム・ロール管理を使う場合、データベースのパラメータ・ファイル内のパラメータ REMOTE_OS_ROLES を TRUE に設定してください。この変更は、次回インスタンスを起動し、データベースをマウントするときに有効となります。（パラメータのデフォルト設定は FALSE です。）

権限とロールの情報を記述

オブジェクトに対応する権限付与をリスト表示するために、次のデータ・ディクショナリ・ビューを問い合わせることができます。

- ALL_COL_PRIVS, USER_COL_PRIVS, DBA_COL_PRIVS
- ALL_COL_PRIVS_MADE, USER_COL_PRIVS_MADE
- ALL_COL_PRIVS_RECD, USER_COL_PRIVS_RECD
- ALL_TAB_PRIVS, USER_TAB_PRIVS, DBA_TAB_PRIVS
- ALL_TAB_PRIVS_MADE, USER_TAB_PRIVS_MADE
- ALL_TAB_PRIVS_RECD, USER_TAB_PRIVS_RECD
- DBA_ROLES
- USER_ROLE_PRIVS, DBA_ROLE_PRIVS
- USER_SYS_PRIVS, DBA_SYS_PRIVS
- COLUMN_PRIVILEGES

- ROLE_ROLE_PRIVS, ROLE_SYS_PRIVS, ROLE_TAB_PRIVS
- SESSION_PRIVS, SESSION_ROLES

関連項目：これらのデータ・ディクショナリ・ビューの詳細は、『Oracle8 Server リファレンス・マニュアル』を参照してください。

権限とロールの情報を記述する例

次の例は、次の指定文の発行が前提条件となっています。

```
CREATE ROLE security_admin IDENTIFIED BY honcho;

GRANT create profile, alter profile, drop profile,
      create role, drop any role, grant any role, audit any,
      audit system, create user, become user, alter user, drop user
      TO security_admin WITH ADMIN OPTION;

GRANT SELECT, DELETE ON sys.aud$ TO security_admin;

GRANT security_admin, create session TO swilliams;

GRANT security_admin TO system_administrator;

GRANT create session TO jward;

GRANT SELECT, DELETE ON emp TO jward;

GRANT INSERT (ename, job) ON emp TO swilliams, jward;
```

付与されているすべてのシステム権限を記述する

次の問合せの結果、ロールとユーザーに対して付与されているシステム権限がすべて表示されます。

```
SELECT * FROM sys.dba_sys_privs;
```

GRANTEE	PRIVILEGE	ADM
-----	-----	----
SECURITY_ADMIN	ALTER PROFILE	YES
SECURITY_ADMIN	ALTER USER	YES
SECURITY_ADMIN	AUDIT ANY	YES
SECURITY_ADMIN	AUDIT SYSTEM	YES
SECURITY_ADMIN	BECOME USER	YES
SECURITY_ADMIN	CREATE PROFILE	YES
SECURITY_ADMIN	CREATE ROLE	YES
SECURITY_ADMIN	CREATE USER	YES
SECURITY_ADMIN	DROP ANY ROLE	YES
SECURITY_ADMIN	DROP PROFILE	YES
SECURITY_ADMIN	DROP USER	YES
SECURITY_ADMIN	GRANT ANY ROLE	YES
SWILLIAMS	CREATE SESSION	NO
JWARD	CREATE SESSION	NO

付与されているすべてのロールを記述する

次の問合せの結果、ユーザーと他のロールに対して付与されたロールがすべて戻されます。

```
SELECT * FROM sys.dba_role_privs;
```

GRANTEE	GRANTED_ROLE	ADM
-----	-----	----
SWILLIAMS	SECURITY_ADMIN	NO

ユーザーに付与されているオブジェクト権限を記述する

次の問合せの結果、特定のユーザーに対して付与されたオブジェクト権限（列固有権限は含まれていません）がすべて戻されます。

```
SELECT table_name, privilege, grantable FROM sys.dba_tab_privs
       WHERE grantee = 'JWARD';
```

TABLE_NAME	PRIVILEGE	GRANTABLE
-----	-----	-----
EMP	SELECT	NO
EMP	DELETE	NO

あらかじめ付与されている列固有の権限をすべて表示するためには、次の問合せを使ってください。

```
SELECT grantee, table_name, column_name, privilege
FROM sys.dba_col_privs;
```

GRANTEE	TABLE_NAME	COLUMN_NAME	PRIVILEGE
-----	-----	-----	-----
SWILLIAMS	EMP	ENAME	INSERT
SWILLIAMS	EMP	JOB	INSERT
JWARD	EMP	NAME	INSERT
JWARD	EMP	JOB	INSERT

セッションの現行の権限定義域を記述する

次の問合せの結果、発行者が現在使えるすべてのロールが表示されます。

```
SELECT * FROM session_roles;
```

SWILLIAMS が SECURITY_ADMIN ロールを使用可能にしてあり、さらに上記の問合せを発行すると、次の情報が戻されます。

```
ROLE
-----
SECURITY_ADMIN
```

次の問合せの結果、この発行者のセキュリティ定義域で現在使用可能なシステム権限がすべて表示されます（明示的に付与されている権限や使用可能なロールはいずれも含まれています）。

```
SELECT * FROM session_privs;
```

SWILLIAMS が SECURITY_ADMIN ロールを使用可能にしてあり、さらに上記の問合せを発行する場合に、次の結果が戻されます。

```
PRIVILEGE
-----
AUDIT SYSTEM
CREATE SESSION
CREATE USER
BECOME USER
ALTER USER
DROP USER
CREATE ROLE
DROP ANY ROLE
GRANT ANY ROLE
AUDIT ANY
CREATE PROFILE
ALTER PROFILE
DROP PROFILE
```

SECURITY_ADMIN ロールが SWILLIAMS に対して使用禁止となっている場合、最初の問合せではまったく行が戻されませんが、2 番目の問合せで CREATE SESSION 権限に対して 1 行だけが戻されます。

データベースのロールをリストする

DBA_ROLES データ・ディクショナリ・ビューを使って、データベースのすべてのロールと各ロールに対して使われている認証が表示されます。たとえば、次の問合せの結果、データベース内のすべてのロールが表示されます。

```
SELECT * FROM sys.dba_roles;
```

ROLE	PASSWORD
-----	-----
CONNECT	NO
RESOURCE	NO
DBA	NO
SECURITY_ADMIN	YES

ロールの権限定義域の情報を記述する

ROLE_ROLE_PRIVS、ROLE_SYS_PRIVS、ROLE_TAB_PRIVS の各データ・ディクショナリ・ビューは、ロール権限の定義域を含むことができます。

たとえば、次の問合せは SYSTEM_ADMIN ロールに付与されているロールをすべてリストします。

```
SELECT granted_role, admin_option
FROM role_role_privs
WHERE role = 'SYSTEM_ADMIN';
```

GRANTED_ROLE	ADM
-----	----
SECURITY_ADMIN	NO

次の問合せの結果、SECURITY_ADMIN ロールに対して付与されたシステム権限すべてが表示されます。

```
SELECT * FROM role_sys_privs WHERE role = 'SECURITY_ADMIN';
```

ROLE	PRIVILEGE	ADM
-----	-----	---
SECURITY_ADMIN	ALTER PROFILE	YES
SECURITY_ADMIN	ALTER USER	YES
SECURITY_ADMIN	AUDIT ANY	YES
SECURITY_ADMIN	AUDIT SYSTEM	YES
SECURITY_ADMIN	BECOME USER	YES
SECURITY_ADMIN	CREATE PROFILE	YES
SECURITY_ADMIN	CREATE ROLE	YES
SECURITY_ADMIN	CREATE USER	YES
SECURITY_ADMIN	DROP ANY ROLE	YES
SECURITY_ADMIN	DROP PROFILE	YES
SECURITY_ADMIN	DROP USER	YES
SECURITY_ADMIN	GRANT ANY ROLE	YES

次の問合せの結果、SECURITY_ADMIN ロールに対して付与されたオブジェクト権限がすべて表示されます。

```
SELECT table_name, privilege FROM role_tab_privs
WHERE role = 'SECURITY_ADMIN';
```

TABLE_NAME	PRIVILEGE
-----	-----
AUD\$	DELETE
AUD\$	SELECT

データベース使用の監査

この章では、Oracle 監査機能の使用方法について説明します。トピックは次のとおりです。

- 監査機能のガイドライン
- データベース監査証跡ビューの作成、削除
- 監査証跡情報の管理
- データベース監査証跡情報の参照
- データベース・トリガーによる監査

監査機能のガイドライン

ここでは、監査機能のガイドラインについて説明します。トピックは次のとおりです。

- データベースまたはオペレーティング・システムで監査する
- 監査済み情報を監理しやすい状態に維持する

データベースまたはオペレーティング・システムで監査する

すべてのデータベースのデータ・ディクショナリには SYS.AUD\$ という名前の表があり、通常、これをデータベースの監査証跡と言います。

データベース監査証跡またはオペレーティング・システム監査証跡のどちらかで、文監査、権限監査、オブジェクト監査の結果として生成された監査レコードをすべて格納できます。

なお、オペレーティング・システムによっては、オペレーティング・システム監査証跡のためのデータベース監査がサポートされているものと、そうでないものがある可能性があります。このオプションが適用可能な場合には、データベース監査レコードを格納するために、データベース監査証跡またはオペレーティング・システム監査証跡のどちらかを適用するに当たり、それぞれの長所および短所を考慮しておかなければなりません。

データベース監査証跡を使う利点は、次のとおりです。

- データ・ディクショナリ中で事前に定義された監査証跡ビューを使って、選択した一部の監査証跡を参照できる。
- Oracle Tools(Oracle Reports など)を使用すると、監査レポートを生成できる。

また、オペレーティング・システム監査証跡によって、Oracle やその他のアプリケーションなどの複数のソースから監査レコードを整理統合できます。したがって、監査レコードがすべて 1 か所にまとめられるため、システム・アクティビティの検証がより効率的に実行されます。

関連項目：オペレーティング・システムには、オペレーティング・システムの監査機能で生成される監査レコードを格納する監査証跡が含まれていることもあります。ただし、この機能はオペレーティング・システムによって異なります。使用しているオペレーティング・システム固有の Oracle のマニュアルを参照してください。

監査済み情報を監理しやすい状態に維持する

監査には比較的成本がかかりますが、可能な限り監査するイベントの回数を制限しておかなければなりません。制限することによって、監査される文の実行に対するパフォーマンスの影響を最小限に抑え、監査証跡のサイズも最小限に抑えられます。

インスタンスとデータベースを起動する前に、次の手順を実行してください。

- 監査の目的を検討する。
監査の目的を明確にしておくと、適切な監査方針を打ち出せるため、不必要な監査をしなくて済みます。

たとえば、疑わしいデータベース・アクティビティの調査のために監査すると仮定します。この情報だけでは特定できません。具体的にどのデータベース・アクティビティが疑わしいと考えているのか、またはそれに気づいていたのか、といった情報が必要です。たとえば、焦点を絞り込んだ監査目的というのは、データベース内の表から許可されていない削除の監査である可能性もあります。この目的であれば、監査の対象となるアクションのタイプや疑わしいアクティビティによって影響を受けるオブジェクトのタイプを制限できます。

- 十分に理解した上での監査。

ターゲットとなる情報を入手するのに必要な最小数の文またはユーザー、オブジェクトを監査します。これによって、不必要な監査情報が重要な情報を混乱させたり、SYSTEM 表領域内の貴重な領域を消費したりすることもあります。セキュリティ情報収集の必要性と、その情報を格納および処理する能力とのバランスを保たなければなりません。

たとえば、データベース・アクティビティに関する情報を収集するために監査する場合には、トラッキングしたいアクティビティのタイプを正確に決定し、目的のアクティビティだけを監査し、必要な情報を収集するために必要な時間数だけを監査します。たとえば、各セッションの論理 I/O 情報だけに関心があるのであれば、オブジェクトを監査しないでください。

疑わしいデータベース・アクティビティを監査する

監査の目的がデータベース・アクティビティを監視することである場合には、次のガイドラインを考慮に入れてください。

- 全体的に監査してから特定の監査。

通常、疑わしいデータベース・アクティビティの監査を開始する時点では、対象とする特定のユーザーまたはスキーマ・オブジェクトに対して使用可能な情報はあまりありません。このため、通常は、最初に全体的な監査オプションを設定しなければなりません。一度予備的な監査情報が記録され分析されれば、この全体的な監査オプションの使用を終了し、特定の監査オプションを使用可能にします。この処理は、疑わしいデータベース・アクティビティの原因に対する具体的な結論が出るまで（その十分な裏付けが収集できるまで）は、必ず続行しなければなりません。

- 監査証拠を保護する。

疑わしいデータベース・アクティビティを監査する場合は、監査証拠を保護して、監査情報が監査されていない状態では追加または変更、削除されないようにします。

関連項目：監査証拠の詳細は、22-16 ページの「監査証拠を保護する」を参照してください。

通常のデータベース・アクティビティを監査する

監査目的が特定のデータベース・アクティビティに関する履歴情報を収集することである場合には、次のガイドラインを考慮に入れてください。

- 関連性のあるアクションだけを監査する。

重要な情報の中に役に立たない監査レコードが入っていることによる混乱を避け、監査証跡の管理操作の量を削減するために、目的のデータベース・アクティビティだけを監査してください。

- 監査レコードをアーカイブし、監査証跡を消去する。

一度必要な情報を収集すれば、目的の監査レコードをアーカイブし、この情報の監査証跡を消去します。

データベース監査証跡ビューの作成、削除

ここでは、データベース監査証跡ビューの作成および削除方法について説明します。トピックは次のとおりです。

- 監査証跡ビューを作成する
- 監査証跡ビューを削除する

データベース監査証跡 (SYS.AUD\$) は、各 Oracle データベースのデータ・ディクショナリ内にある単一の表です。この表の中には、重要な監査情報の参照に役立つように、規定のビューがいくつか用意されています。監査証跡は監査機能を使用するために作成されています。監査機能を使用しない場合には、後で監査証跡を削除できます。

関連項目：ほとんどのオペレーティング・システムでは、監査証跡ビューはデータ・ディクショナリを使用して自動的に作成されます。使用しているオペレーティング・システム固有の Oracle のマニュアルを参照してください。

監査証跡ビューを作成する

監査機能を適用する場合には、SYS として接続した上で CATAUDIT.SQL スクリプトを稼働して、監査ビューを作成します。このスクリプトにより、次のビューが作成されます。

- STMT_AUDIT_OPTION_MAP
- AUDIT_ACTIONS
- ALL_DEF_AUDIT_OPTS
- DBA_STMT_AUDIT_OPTS
- USER_OBJ_AUDIT_OPTS, DBA_OBJ_AUDIT_OPTS
- USER_AUDIT_TRAIL, DBA_AUDIT_TRAIL
- USER_AUDIT_SESSION, DBA_AUDIT_SESSION
- USER_AUDIT_STATEMENT, DBA_AUDIT_STATEMENT
- USER_AUDIT_OBJECT, DBA_AUDIT_OBJECT
- DBA_AUDIT_EXISTS
- USER_AUDIT_SESSION, DBA_AUDIT_SESSION

- USER_TAB_AUDIT_OPTS

関連項目：これらのビューの詳細は、『Oracle8 Server リファレンス・マニュアル』を参照してください。

監査情報の解釈例の詳細は、22-16 ページの「データベース監査証跡情報の参照」を参照してください。

監査証跡ビューを削除する

監査を使用禁止にし、その監査証跡ビューが必要なくなったのであれば、SYS としてデータベースへ接続し、CATNOAUD.SQL スクリプト・ファイルを実行して、それらのビューを削除します。この CATNOAUD.SQL スクリプトの名前と位置は、オペレーティング・システムによって異なります。

監査証跡情報の管理

ここでは、監査証跡情報の管理について説明します。トピックは次のとおりです。

- デフォルトで監査されるイベント
- 監査オプションを設定する
- データベース監査を使用可能、使用禁止にする
- 監査証跡の成長とサイズを制御する
- 監査証跡を保護する

監査されるイベントや設定される監査オプションによっては、監査証跡レコードに複数の異なるタイプの情報が含まれることがあります。次に示す情報は特定の監査アクションに対して意味のある情報である場合には、常に各監査証跡レコードに含まれます。

- ユーザー名
- セッション識別子
- 端末識別子
- アクセスされたオブジェクトの名前
- 実行または試行された操作
- 操作の完了コード
- 日付と時刻のタイムスタンプ

オペレーティング・システム監査証跡に書き込まれた監査証跡レコードには、判読不可能なコードが含まれています。これらのコードは次のように解読できます。

アクション・コード

このコードは、実行または試行された操作を記述しています。AUDIT_ACTIONS データ・ディクショナリ表に、これらのコードとその説明のリストが含まれています。

使用する権限

操作の実行に使用された権限を記述しています。SYSTEM_PRIVILEGE_MAP 表に、これらのコードとその説明がすべて記述されています。

完了コード

試行した操作の結果を記述しています。操作が正常に終了すると値 0 が戻され、失敗すると操作が失敗した原因を記述した Oracle エラー・コードが戻されます。

関連項目：エラー・コードは、『Oracle8 Server エラー・メッセージ』に記述されています。

デフォルトで監査されるイベント

データベース監査が使用可能かどうかにかかわらず、Oracle Server では常にデータベースに関連する特定のアクションを監査し、オペレーティング・システム監査証跡に記録します。このようなイベントは次のとおりです。

インスタンスの起動	インスタンスを起動した OS ユーザー、そのユーザーの端末識別子、日付と時刻のタイムスタンプ、データベース監査が使用可能かどうかなどを記述した監査レコードが生成されます。データベース監査証跡は起動処理が正常に完了するまでは使用できないため、この監査レコードは OS 監査証跡に記録されます。起動時のデータベース監査の状態を記録しておくことで、監査されていないアクションを管理者が実行できるようにするために、データベース監査が使用禁止の状態ですべてデータベースを再起動することを防止します。
インスタンス・シャットダウン	インスタンスを停止した OS ユーザー、そのユーザーの端末識別子、日付と時刻のタイムスタンプを記述した監査レコードが生成されます。
接続先を持つデータベース管理者権限	SYSOPER または SYSDBA として Oracle に接続した OS ユーザーについて記述した監査レコードが生成されます。SYSOPER または SYSDBA により、ユーザーのアカウントにシステム権限が付与されます。

Oracle が OS の監査証跡にアクセスできないオペレーティング・システムでは、これらの監査証跡レコードはバックグラウンド・プロセス・トレース・ファイルと同じディレクトリにある Oracle の監査証跡ファイルに記録されます。

監査オプションを設定する

監査オプションの設定に応じて、監査レコードにさまざまなタイプの情報を記録できます。なお、全監査オプションにより、次の情報が生成されます。

- 監査文を実行したユーザーに関する情報
- ユーザーによって実行された監査文を示すアクション・コード（コード番号）に関する情報
- 監査文で参照されたオブジェクトに関する情報
- 監査文が実行された日時に関する情報

監査証跡は、監査された文に関連するデータ値に関する情報は格納しません。たとえば、UPDATE 文の監査時には、更新された行の新旧のデータ値は格納されません。ただし、この特殊な監査機能は、データベース・トリガーを使って、表に関連する DML 文で実行できます。

Oracle では、次の 3 つのレベルで監査オプションを設定できます。

文	SQL 文のタイプに基づいて監査します。たとえば、表に関する SQL 文を監査します（CREATE TABLE 文、TRUNCATE TABLE 文、DROP TABLE 文をそれぞれ記録します）。
権限	特定のシステム権限の使用を監査します。たとえば、CREATE TABLE を監査します。
オブジェクト	特定のオブジェクトに関する特定の文を監査します。たとえば、EMP 表に関する ALTER TABLE を監査します。

関連項目：この特殊な監査機能を実行するためのトリガーの使用法の例は、22-20 ページの「データベース・トリガーによる監査」を参照してください。

文監査オプション

AUDIT 文と NOAUDIT 文に指定できる有効な文監査オプションは、『Oracle8 Server SQL リファレンス』にリストしています。

文監査オプションのためのショート・カット ショート・カットは、1 語で複数の関連する文オプションを指定するために用意されています。

ショートカットとは、文オプション自体ではなく、AUDIT 文および NOAUDIT 文に、関連する複数の文オプションを 1 語で指定するためのものです。システム権限と文オプションのショート・カットは、『Oracle8 Server SQL リファレンス』で詳述しています。

接続と切断を監査する

SESSION 文オプション（および CONNECT ショートカット）は、特殊なタイプの文の発行時には監査レコードを生成しないため、一意となります。このオプションは、インスタンスへの接続によって作成されたセッションごとに、単一の監査レコードを生成します。監査レコードは、接続時に監査証跡に挿入され、切断時に更新されます。接続時刻、切断時刻、論理 I/O や物理 I/O の処理などのセッションに関する蓄積情報は、そのセッションに対応する単一の監査レコード内に格納されます。

関連項目：『Oracle8 Server SQL リファレンス』には、ショート・カットでカバーされないその他の監査オプションもリストしています。

権限監査オプション

権限監査オプションは、対応するシステム権限と正確に一致します。たとえば、DELETE ANY TABLE 権限の使用を監査するためのオプションは、DELETE ANY TABLE です。このオプションを有効にするには、次のような文を使用するとよいでしょう。

```
AUDIT DELETE ANY TABLE  
    BY ACCESS  
    WHENEVER NOT SUCCESSFUL;
```

Oracle のシステム権限のリストは、21-2 ページの「システム権限」に示されています。

オブジェクト監査オプション

『Oracle8 Server SQL リファレンス』には、有効なオブジェクト監査オプションと、各オプションが使用できるスキーマ・オブジェクトのタイプをリストしています。

表 22-1 に、各オブジェクト・オプションによって監査される SQL 文を示します。

表 22-1 データベース・オブジェクト監査オプションにより監査される SQL 文

オブジェクト・オプション	表
ALTER	ALTER オブジェクト (表または順序)
AUDIT	AUDIT(形式 II) オブジェクト
COMMENT	COMMENT オブジェクト (表またはビュー)
DELETE	DELETE FROM オブジェクト (表またはビュー)
EXECUTE	EXECUTE オブジェクト (プロシージャ ¹)
GRANT	GRANT(形式 II) 権限 ON オブジェクト
INDEX	CREATE INDEX ON オブジェクト (表だけ)
INSERT	INSERT INTO オブジェクト (表またはビュー)
LOCK	LOCK オブジェクト (表またはビュー)
RENAME	RENAME オブジェクト (表、ビュー、プロシージャ)
SELECT	SELECT .. FROM オブジェクト (表、ビュー、スナップショット)
UPDATE	UPDATE オブジェクト (表またはビュー)
¹ プロシージャは、スタンドアロン・ストアド・プロシージャ、およびファンクション、パッケージを意味します。	

オブジェクト監査オプションのためのショートカット ALL ショートカット使用して、スキーマ・オブジェクトに対して指定可能なすべてのオブジェクト監査オプションを指定できます。このショートカットは、オプション自体ではなく、AUDIT 文および NOAUDIT 文の中の 1 語ですべてのオブジェクト監査オプションを指定するための手段です。

AUDIT オプションを使用可能にする

SQL コマンド AUDIT は、文監査オプションおよび権限監査オプション、監査オプションを使用可能にします。文監査オプションと権限監査オプションを設定する監査文には、BY USER オプションを含めて、ユーザーのリストを指定し、文監査オプションと権限監査オプションの有効範囲を制限できます。SQL コマンド AUDIT によって、監査オプションが使用可能になります。文オプションと権限オプションを設定するために、この文を使用するには、AUDIT SYSTEM 権限を持っていない限りなりません。オブジェクト監査オプションを設定するために、この文を使用するには、監査対象のオブジェクトを所有しているか、または AUDIT ANY 権限を持っていない限りなりません。

任意の監査オプションを設定し、監査機能として次の条件を指定できます。

- WHENEVER SUCCESSFUL/WHENEVER NOT SUCCESSFUL
- BY SESSION/BY ACCESS

新しいデータベース・セッションが作成されると、このセッションはデータ・ディクショナリの監査オプションを使用します。これらの監査オプションは、データベースに接続している間は有効です。新しいシステム監査オプションまたはオブジェクト監査オプションを設定すると、それ以降のデータベース・セッションでは新たに設定されたオプションが使用されます。既存のセッションでは、セッションの作成時に指定された監査オプションが引き続き使用されます。

警告： AUDIT コマンドを実行しても、監査オプションがオンになるだけで、監査機能の全体が使用可能になるわけではありません。監査オプションを使用可能にして、現在設定されている監査オプションに基づいた、Oracle による監査レコードの生成を制御するには、データベースのパラメータ・ファイルにパラメータ AUDIT_TRAIL を設定します。

関連項目： AUDIT コマンドの詳細は、『Oracle8 Server SQL リファレンス』を参照してください。

監査を使用可能および使用禁止にすることの詳細は、22-13 ページの「データベース監査を使用可能、使用禁止にする」を参照してください。

文権限監査を使用可能にする ユーザーに関係なく、正常終了および異常終了したデータベースとのすべての接続、また正常終了および異常終了したデータベースからのすべての切断について、BY SESSION(このオプションのデフォルト値かつ唯一の値)を指定して監査するには、次の文を指定します。

```
AUDIT SESSION;
```

このオプションをユーザーごとに選択的に設定することもできます。たとえば、次の例に示すとおりです。

```
AUDIT SESSION  
BY scott, lori;
```

DELETE ANY TABLE システム権限の正常な使用および異常終了した使用をすべて監査するためには、次の文を指定します。

```
AUDIT DELETE ANY TABLE;
```

すべてのデータベース・ユーザーを対象に、BY ACCESS で、すべての表で異常終了した SELECT、INSERT、DELETE の文をすべて、さらに EXECUTE PROCEDURE システム権限の異常終了した使用をすべて監査するためには、次の文を指定します。

```
AUDIT SELECT TABLE, INSERT TABLE, DELETE TABLE,  
EXECUTE PROCEDURE  
BY ACCESS  
WHENEVER NOT SUCCESSFUL;
```

文監査オプションまたは権限監査オプションの設定には、AUDIT SYSTEM システム権限が必要です。通常、セキュリティ管理者だけがこの権限を付与されている唯一のユーザーです。

オブジェクト監査を使用可能にする BY SESSION(デフォルト値) によって、SCOTT.EMP 表に関する正常終了および異常終了した DELETE 文をすべて監査するためには、次の文を指定します。

```
AUDIT DELETE ON scott.emp;
```

ユーザーの JWARD が所有している DEPT 表に関する正常終了した SELECT、INSERT、DELETE の文をすべて監査するためには、次の文を指定します。

```
AUDIT SELECT, INSERT, DELETE  
  ON jward.dept  
  BY ACCESS  
  WHENEVER SUCCESSFUL;
```

BY SESSION (デフォルト値) によって、異常終了した全 SELECT 文を監査するためのデフォルトのオブジェクト監査オプションを設定するためには、次の文を指定します。

```
AUDIT SELECT  
  ON DEFAULT  
  WHENEVER NOT SUCCESSFUL;
```

ユーザーは、自分のスキーマ内にあるオブジェクトのオブジェクト監査オプションを設定できます。他のユーザーのスキーマ内にあるオブジェクトのオブジェクト監査オプションを設定したり、デフォルトのオブジェクト監査オプションを設定したりするには、AUDIT ANY システム権限が必要です。通常、このシステム権限はセキュリティ管理者だけに付与されています。

AUDIT オプションを使用禁止にする

NOAUDIT コマンドを使って、Oracle の各種監査オプションの設定を解除します。文監査オプション、権限監査オプション、オブジェクト監査オプションの設定を解除するために、この監査オプションを使用します。文監査オプションや権限監査オプションを設定する NOAUDIT 文には、ユーザー・リストを規定する BY USER オプションを指定して、その文監査オプションおよび権限監査オプションの範囲を制限できます。

NOAUDIT 文では、WHENEVER 句を使って監査オプションを選択的に使用禁止にできます。この句を指定しなければ、正常終了または異常終了のどちらの場合にも監査オプションは完全に使用禁止となります。

BY SESSION/BY ACCESS オプションは、NOAUDIT コマンドにはサポートされていません。このため、監査オプションの設定方法には関係なく、監査オプションは適切な NOAUDIT 文によってその設定が解除されます。

警告： NOAUDIT コマンドを実行しても、監査オプションがオフになるだけで、監査機能の全体が使用禁止になるわけではありません。監査オプションを使用禁止にして、現在監査オプションが設定されている状態で、Oracle による監査レコードの生成を停止するには、データベースのパラメータ・ファイルにパラメータ AUDIT_TRAIL を設定します。

関連項目：NOAUDIT コマンド完全な構文リストは、『Oracle8 Server SQL リファレンス』を参照してください。

また、22-13 ページの「データベース監査を使用可能、使用禁止にする」を参照してください。

文監査と権限監査を使用禁止にする

次の文は、対応する監査オプションを使用禁止にします。

```
NOAUDIT session;
NOAUDIT session BY scott, lori;
NOAUDIT DELETE ANY TABLE;
NOAUDIT SELECT TABLE, INSERT TABLE, DELETE TABLE,
EXECUTE PROCEDURE;
```

次の文は、すべての文（システム）監査オプションおよび権限監査オプションを使用禁止にします。

```
NOAUDIT ALL;
NOAUDIT ALL PRIVILEGES;
```

文監査オプションまたは権限監査オプションを使用禁止にするには、AUDIT SYSTEM システム権限が必要です。

オブジェクト監査を使用禁止にする 次の文は、対応する監査オプションをオフにします。

```
NOAUDIT DELETE
ON emp;
NOAUDIT SELECT, INSERT, DELETE
ON jward.dept;
```

さらに、EMP 表に関するオブジェクト監査オプションをすべて使用禁止にするには、次の文を入力します。

```
NOAUDIT ALL
ON emp;
```

デフォルトのオブジェクト監査オプションを使用禁止にする デフォルトのオブジェクト監査オプションをすべてオフにするには、次の文を入力します。

```
NOAUDIT ALL
ON DEFAULT;
```

なお、この NOAUDIT 文を発行する前に作成されたすべてのスキーマ・オブジェクトは、そのスキーマ・オブジェクトの作成後に明示的に NOAUDIT 文でオーバーライドしない限り、作成時に実施されたデフォルトのオブジェクト監査オプションが引き続き使用されます。

特定のオブジェクトのオブジェクト監査オプションを使用禁止にするには、そのスキーマ・オブジェクトの所有者でなければなりません。なお、別のユーザーのスキーマ内のオブジェクトのオブジェクト監査オプションを使用禁止にしたり、デフォルトのオブジェクト監査オプションを使用禁止にするためには、AUDIT ANY システム権限を持っていない限りなりません。オブジェクトのオブジェクト監査オプションを使用禁止にする権限を持っているユーザーは、任意のユーザーによって設定されたオプションを置き換えることができます。

データベース監査を使用可能、使用禁止にする

許可されたデータベース・ユーザーであれば、文および権限、オブジェクト監査オブジェクトを任意で設定できますが、データベース監査が使用可能な状態でないと、Oracle は監査証跡に監査レコードを生成したり、格納したりしません。通常、セキュリティ管理者がこの操作を担当します。

データベース監査は、データベースのパラメータ・ファイル内の AUDIT_TRAIL 初期化パラメータによって使用可能にしたり、使用禁止にされたりします。このパラメータには次の値を設定できます。

DB	データベース監査を使用可能にして、データベース監査証跡のためのすべての監査レコードを管理します。
OS	データベース監査を使用可能にして、オペレーティング・システム監査証跡のためにすべての監査レコードを管理します。
NONE	監査を使用禁止します（この値はデフォルトです）。

パラメータ・ファイルを編集したならば、データベース監査を使用可能または使用禁止にするために、データベース・インスタンスを再起動してください。

関連項目：パラメータ・ファイルの編集の詳細は、『Oracle8 Server リファレンス・マニュアル』を参照してください。

監査証跡の成長とサイズを制御する

監査証跡が完全に満杯になり、これ以上監査レコードを挿入できなくなった場合は、証跡が消去されない限り、監査された文を正常に実行できません。監査された文を発行するユーザー全員に対して警告が戻されます。このため、セキュリティ管理者は必ず監査証跡の成長とサイズを制御しなければなりません。

監査を使用可能にして監査レコードを生成する際、次の 2 つの要因に応じて、監査証跡は増加します。

- 使用可能になっている監査オプションの数
- 監査文の実行頻度

監査証跡の成長を制御するためには、次の方法を使用できます。

- データベース監査を使用可能および使用禁止にする。使用可能にすると、監査レコードが生成され監査証跡に格納されます。使用禁止の場合、監査レコードが生成されません。
- 使用可能となっている監査オプションを任意に選択可能にする。選択的な監査を実施すると、不要な監査情報は生成されず、監査証跡に格納されません。
- オブジェクト監査を実行する機能を厳密に制御する。これには、2 種類の方法があります。
 - セキュリティ管理者はすべてのオブジェクトを有し、その AUDIT ANY システム権限を決して他のユーザーに付与しない。そのかわりに、すべてのスキーマ・オブジェクトは、対応するユーザーが CREATE SESSION 権限を有していないスキーマに属するようにできます。
 - すべてのオブジェクトを、実際のデータベース・ユーザーに対応していないスキーマ内に収録しておき（つまり、CREATE SESSION 権限が対応するユーザーに付与されていない）、セキュリティ管理者だけが、AUDIT ANY システム権限を付与された唯一のユーザーとなる。

上記のとおりを実施すれば、オブジェクト監査についてはセキュリティ管理者が完全に制御できます。

データベース監査証跡（SYS.AUD\$ 表）の最大サイズは、データベース作成中に事前に定義されます。特に何も指定しなければ、この表には最大 99 エクステンツ、各 10KB ずつ割り当てることができます。

監査証跡の成長とサイズを制御する手段として、SYS.AUD\$ を他の表領域に移動するという方法は使用できません。しかし、SYS.AUD\$ 内のデフォルトの記憶領域パラメータ（INITIAL を除く）は修正できます。

関連項目：監査レコードをオペレーティング・システムの監査証跡に出力する場合の、オペレーティング・システム監査証跡の管理方法の詳細は、使用しているオペレーティング・システム固有の Oracle のマニュアルを参照してください。

関連項目：SYS.AUD\$ 記憶領域パラメータの詳細は、『Oracle8 Server リファレンス・マニュアル』を参照してください。

監査証跡から監査レコードを消去する

監査を使用可能にした後でも、セキュリティ管理者はレコードをデータベース監査証跡から削除して、監査証跡の領域を解放し、監査証跡の管理を容易にできます。

たとえば、監査証跡からすべての監査レコードを削除するためには、次の文を指定します。

```
DELETE FROM sys.aud$;
```

また、EMP 表の監査の結果、生成された監査証跡からすべての監査レコードを削除するためには、次の文を指定します。

```
DELETE FROM sys.aud$  
WHERE obj$name='EMP';
```

監査証跡を履歴の目的で必ずアーカイブしなければならない場合には、セキュリティ管理者は通常のデータベース表へ関連するレコードをコピーしたり（たとえば、「INSERT INTO table SELECT ... FROM sys.aud\$...」を使用して）、またはオペレーティング・システム・ファイルへ監査証跡表をエクスポートしたりできます。

DELETE ANY TABLE 権限を持っている SYS ユーザー、または SYS が SYS.AUD\$ の DELETE 権限をあらかじめ付与してあるユーザーに限り、データベース監査証跡からレコードを削除できます。

注意： 監査証跡が完全に満杯になり接続が監査される場合（つまり、SESSION オプションを設定している場合）、その接続に対応する監査レコードを監査証跡に挿入できないため、通常のユーザーはデータベースに接続できません。この場合には、セキュリティ管理者は必ず SYS として接続し（SYS によって操作は監査されません）、監査証跡で使用可能な領域を作成しなければなりません。

関連項目：表のエクスポートの詳細は、『Oracle8 Server ユーティリティ』を参照してください。

監査証跡のサイズを削減する

データベース監査証跡からレコードが削除された後に、データベース表を使用すると、この表に割り当てられたエクステントはそのまま存在します。

データベース監査証跡が表に対して多数のエクステントを割り当てていても、そのうちの大半が使用されていないければ、データベース監査証跡用に割り当ててある領域は、次の手順に従って、削減できます。

1. 現在、監査証跡内に情報を保存したい場合には、別のデータベース表にその情報をコピーするか、または EXPORT ユーティリティを使用してその情報をエクスポートする。
2. 管理者権限を使用して接続する。
3. TRUNCATE コマンドを使用して、SYS.AUD\$ を切り捨てる。
4. 手順 1 によって生成されたアーカイブ済みの監査証跡レコードをロードしなおす。

SYS.AUD\$ の新しいバージョンは、現在の監査証跡レコードを記録するのに必要なエクステンツの数だけ割り当てられます。

注意： 直接修正できる SYS オブジェクトは SYS.AUD\$ だけです。

監査証跡を保護する

なんらかの疑わしいデータベース・アクティビティを監査する場合は、監査証跡のレコードの整合性を保護することによって、正確で完全な監査情報を保証してください。

データベースの監査証跡を未許可の削除処理から保護するため、DELETE ANY TABLE システム権限はセキュリティ管理者に対してだけ付与します。

データベース監査証跡に対してなされた変更を監査するには、次の文を使用します。

```
AUDIT INSERT, UPDATE, DELETE
ON sys.aud$
BY ACCESS;
```

SYS.AUD\$ 表に対してオブジェクト監査オプションを設定した結果生成された監査レコードを削除できるのは、管理者権限で接続しているユーザーだけです。なお、この表自体は許可されていない使用に対して保護されています。監査証跡の最終的な保護基準として、管理者権限で接続しているときに実行された操作は、使用可能であれば、オペレーティング・システム監査証跡で監査されます。

関連項目： オペレーティング・システム監査証跡の可用性とその使用方法の詳細は、使用しているオペレーティング・システム固有の Oracle のマニュアルを参照してください。

データベース監査証跡情報の参照

ここでは、監査証跡情報の検査および解釈方法の具体例を説明します。トピックは次のとおりです。

- アクティブな文監査オプションを記述する
- 特定のオブジェクトのアクティブな権限監査オプションを記述する
- 特定のオブジェクトのアクティブなオブジェクト監査オプションをリストする
- デフォルトのオブジェクト監査オプションを記述する

- 監査レコードを記述する
- AUDIT SESSION オプションの監査レコードを記述する

次のような疑わしいアクティビティが存在する場合は、データベースを監査する必要があります。

- 許可なく変更されているデータベース・ユーザーに対するパスワードおよび表領域の設定、割当て制限。
- おそらく排他的に表ロックを必要ではないかというユーザーのために、頻発に発生しているデッドロックの回数。
- SCOTT のスキーマ内の EMP 表から削除されている行。

たとえば、ユーザー JWARD および SWILLIAMS が不正なアクションを行った疑いがある場合、データベース管理者は、次の文を（示されている順序で）発行します。

```
AUDIT ALTER, INDEX, RENAME ON DEFAULT
    BY SESSION;
CREATE TABLE scott.emp . . . ;
CREATE VIEW scott.employee AS SELECT * FROM scott.emp;
AUDIT SESSION BY jward, swilliams;
AUDIT ALTER USER;
AUDIT LOCK TABLE
    BY ACCESS
    WHENEVER SUCCESSFUL;
AUDIT DELETE ON scott.emp
    BY ACCESS
    WHENEVER SUCCESSFUL;
```

JWARD ユーザーによって次の文が発行されます。

```
ALTER USER tsmith QUOTA 0 ON users;
DROP USER djones;
```

SWILLIAMS ユーザーによって次の文が発行されます。

```
LOCK TABLE scott.emp IN EXCLUSIVE MODE;
DELETE FROM scott.emp WHERE mgr = 7698;
ALTER TABLE scott.emp ALLOCATE EXTENT (SIZE 100K);
CREATE INDEX scott.ename_index ON scott.emp (ename);
CREATE PROCEDURE scott.fire_employee (empid NUMBER) AS
BEGIN
    DELETE FROM scott.emp WHERE empno = empid;
END;
/

EXECUTE scott.fire_employee(7902);
```

この後の部分に、データ・ディクショナリ内の監査証跡ビューを使用して表示できる情報を示します。

アクティブな文監査オプションを記述する

次の問合せの結果、設定されている文監査オプションがすべて戻されます。

```
SELECT * FROM sys.dba_stmt_audit_opts;
```

USER_NAME	AUDIT_OPTION	SUCCESS	FAILURE
-----	-----	-----	-----
JWARD	SESSION	BY SESSION	BY SESSION
SWILLIAMS	SESSION	BY SESSION	BY SESSION
	LOCK TABLE	BY ACCESS	NOT SET

このビューにより、文監査オプションが正常終了または異常終了（あるいはその両方）のどちらかに設定され、それぞれ BY SESSION または BY ACCESS のどちらかに設定されているかなど、設定が明らかになります。

特定のオブジェクトのアクティブな権限監査オプションを記述する

次の問合せの結果、設定されている権限がすべて戻されます。

```
SELECT * FROM sys.dba_priv_audit_opts;
```

USER_NAME	AUDIT_OPTION	SUCCESS	FAILURE
-----	-----	-----	-----
ALTER USER	BY SESSION	BY SESSION	

特定のオブジェクトのアクティブなオブジェクト監査オプションをリストする

次の問合せの結果、SCOTT のスキーマ内に収録されたオブジェクトに対する監査オプションがすべて戻されます。

```
SELECT * FROM sys.dba_obj_audit_opts
WHERE owner = 'SCOTT' AND object_name LIKE 'EMP%';
```

OWNER	OBJECT_NAME	OBJECT_TY	ALT	AUD	COM	DEL	GRA	IND	INS	LOC	...
-----	-----	-----	---	---	---	---	---	---	---	---	---
SCOTT	EMP	TABLE	S/S	-/-	-/-	A/-	-/-	S/S	-/-	-/-	...
SCOTT	EMPLOYEE	VIEW	-/-	-/-	-/-	A/-	-/-	S/S	-/-	-/-	...

このビューにより、指定したオブジェクトに対するすべての監査オプションの情報が戻されます。このビューの情報は次のように解釈します。

- 文字 "-" は、監査オプションが何も設定されていないことを示す。
- 文字 "S" は、監査オプションが BY SESSION に設定されていることを示す。
- 文字 "A" は、監査オプションが BY ACCESS に設定されていることを示す。

- 各監査オプションには、"/" で区切られた WHENEVER SUCCESSFUL と WHENEVER NOT SUCCESSFUL の 2 つの設定がある。たとえば、SCOTT.EMP に対する DELETE 監査オプションは、正常終了した削除に対して BY ACCESS を設定し、異常終了した削除文に対しては何も設定していません。

デフォルトのオブジェクト監査オプションを記述する

次の問合せは、デフォルトのオブジェクト監査オプションをすべて戻します。

```
SELECT * FROM all_def_audit_opts;

ALT AUD COM DEL GRA IND INS LOC REN SEL UPD REF EXE
---
S/S -/- -/- -/- -/- S/S -/- -/- S/S -/- -/- -/-
```

このビューにより、USER_OBJ_AUDIT_OPT と DBA_OBJ_AUDIT_OPTS の各ビューに類似する情報が戻されます（先の例を参照してください）。

監査レコードを記述する

次の問合せの結果、文監査オプションとオブジェクト監査オプションで生成された監査レコードが一覧で表示されます。

```
SELECT * FROM sys.dba_audit_object;
```

AUDIT SESSION オプションの監査レコードを記述する

次の問合せの結果、AUDIT SESSION 文監査オプションに対応する監査オプションが一覧で表示されます。

```
SELECT username, logoff_time, logoff_lread, logoff_pread,
       logoff_lwrite, logoff_dlock
FROM sys.dba_audit_session;
```

USERNAME	LOGOFF_TI	LOGOFF_LRE	LOGOFF_PRE	LOGOFF_LWR	LOGOFF_DLO
JWARD	02-AUG-91	53	2	24	0
SWILLIAMS	02-AUG-91	3337	256	630	0

データベース・トリガーによる監査

トリガーを使用して、Oracle の組み込まれた監査機能を補足できます。トリガーを使用して、AUDIT コマンドで書き込んだ情報と同じようなレコード情報を書き込むことができますが、これはより詳細な監査情報が必要な場合にだけ使用します。たとえば、トリガーを使用して、表の行単位に値ベースで監査できます。

注意： いくつかの分野では、Oracle の AUDIT コマンドはセキュリティ監査機能と見なされますが、トリガーはファイナンシャル監査機能を提供できます。

データベース・アクティビティを監査するトリガーを作成するかどうかを決める際に、トリガーによる監査機能と比較した場合の、標準 Oracle データベースの監査機能による利点を考慮しておきます。

- 標準監査オプションは、すべてのタイプのスキーマ・オブジェクトと構造に関する DML と DDL 文の監査が可能である。これに対して、トリガーは表に対して発行された DML 文に限り監査できます。
- データベースの監査情報はすべて集中的に記録され、Oracle の監査機能に自動的に使用される。
- 標準 Oracle の機能を使用して使用可能となる監査機能は、宣言と保守が簡単であり、トリガーが定義する監査機能と比較しても、エラーはまず発生しない。
- 既存の監査オプションの変更を監査して、不当なデータベース・アクティビティを防止できる。
- データベースの監査機能を使用して、監査文が発行されるたび (BY ACCESS)、または監査文を発行するセッションごとに (BY SESSION)、レコードを作成できる。トリガーはセッションごとには監査できません。トリガー監査表が参照されるたびに、監査レコードが作成されます。
- データベース監査では、失敗したデータ・アクセスを監査できる。ただし、トリガー文がロールバックされると、トリガーが生成した監査情報もロールバックされます。
- 標準データベース監査機能を使用して、接続と切断、およびセッション・アクティビティ (物理 I/O、論理 I/O、デッドロックなど) を記録できる。

トリガーを使用して高度な監査を実行するには、通常 AFTER トリガーを使用します。AFTER トリガーによって、整合性制約に従うトリガー文の後に、監査情報を記録できます。監査処理において、整合性制約の例外を生成する文の無意味な実行を防止します。

AFTER 行と AFTER 文トリガーの使い分けは監査情報に応じて異なります。たとえば、表の中の行の値を行単位で監査するには、行トリガーが便利です。トリガーを使用して、監査済み SQL 文を発行して「理由コード」を出力できます。これは、行と文レベルの監査状況の両方に便利です。

次は、EMP 表に対する修正を行単位で監査するトリガーの例です。この例では、更新前に、「理由コード」をグローバル・パッケージ変数に格納する必要があります。このトリガーでは、次のことが例示されます。

- トリガーを使用して値ベースの監査を実行する方法
- パブリック・パッケージ変数の使用方法

コメントには、トリガーの機能が記述されています。

```
CREATE TRIGGER audit_employee
AFTER INSERT OR DELETE OR UPDATE ON emp
FOR EACH ROW
BEGIN
  /* AUDITPACKAGE is a package with a public package
   variable REASON. REASON could be set by the
   application by a command such as EXECUTE
   AUDITPACKAGE.SET_REASON(reason_string). Note that a
   package variable has state for the duration of a
   session and that each session has a separate copy of
   all package variables. */
  IF auditpackage.reason IS NULL THEN
    raise_application_error(-20201,'Must specify reason with ',
    'AUDITPACKAGE.SET_REASON(reason_string)');
  END IF;

  /* If the above conditional evaluates to TRUE, the
   user-specified error number and message is raised,
   the trigger stops execution, and the effects of the
   triggering statement are rolled back. Otherwise, a
   new row is inserted into the pre-defined auditing
   table named AUDIT_EMPLOYEE containing the existing
   and new values of the EMP table and the reason code
   defined by the REASON variable of AUDITPACKAGE. Note
   that the "old" values are NULL if triggering
   statement is an INSERT and the "new" values are NULL
   if the triggering statement is a DELETE. */
  INSERT INTO audit_employee VALUES
    (:old.ssn, :old.name, :old.job_classification, :old.sal,
    :new.ssn, :new.name, :new.job_classification, :new.sal,
    auditpackage.reason, user, sysdate );
END;
```

更新ごとに理由コードを強制的に設定したい場合、任意に理由コードを NULL に設定し直すこともできます。次の AFTER 文トリガーは、トリガー文の実行後に、理由コードを NULL に設定し直します。

```
CREATE TRIGGER audit_employee_reset
AFTER INSERT OR DELETE OR UPDATE ON emp
BEGIN
```

```
auditpackage.set_reason(NULL);  
END;
```

上記のトリガーは2つとも同じタイプの SQL 文によって実行されます。ただし、トリガー文の実行が終了している場合は、AFTER 文トリガーは一度しか実行されないのに対し、AFTER 行トリガーは、トリガー文が処理対象とする表の各行ごとに一度実行されます。

REDO 情報のアーカイブ

この章では、アーカイブ済み REDO ログの作成およびメンテナンスの方法を説明します。トピックは次のとおりです。

- NOARCHIVELOG モードと ARCHIVELOG モードの選択
- アーカイブの切り替え
- アーカイブの調整
- アーカイブ状態を表示
- アーカイブ済み REDO ログのファイル名フォーマットと宛先を指定

関連項目： Oracle を Parallel Server で使っている環境でのアーカイブ方法の詳細は、『Oracle8 Parallel Server 概要および管理』を参照してください。

この章には、Oracle Enterprise Manager について述べている箇所がいくつかあります。Enterprise Manager/GUI または Server Manager/LineMode を使って特定のタスクを実行する方法の詳細は、『Oracle Enterprise Manager ユーザーズ・ガイド』を参照してください。

NOARCHIVELOG モードと ARCHIVELOG モードの選択

ここでは、データベースを NOARCHIVELOG モードまたは ARCHIVELOG モードで稼働する際の考慮点について説明します。トピックは次のとおりです。

- データベースを NOARCHIVELOG モードで稼働する
- データベースを ARCHIVELOG モードで稼働する

データベースを NOARCHIVELOG モードで稼働する

データベースを NOARCHIVELOG モードで稼働すると、オンライン REDO ログのアーカイブは使用禁止になります。データベースの制御ファイルに格納されている情報は、満杯のグループをアーカイブする必要がないことを示します。その結果、満杯のグループがアクティブではなくなり、ログ・スイッチのチェックポイントが完了すると、LGWR がそのグループを再使用できるようになります。

NOARCHIVELOG モードでは、データベースはインスタンス障害からだけは保護されますが、ディスク（メディア）障害からは保護されません。オンライン REDO ログのグループに格納されている、データベースへの変更のうち、最新の変更だけがインスタンスの回復に有効です。つまり、NOARCHIVELOG モードでは、最後に全体バックアップしたときの状態にしかデータベースを復元（回復ではない）できません。それ以降のトランザクションは回復できません。

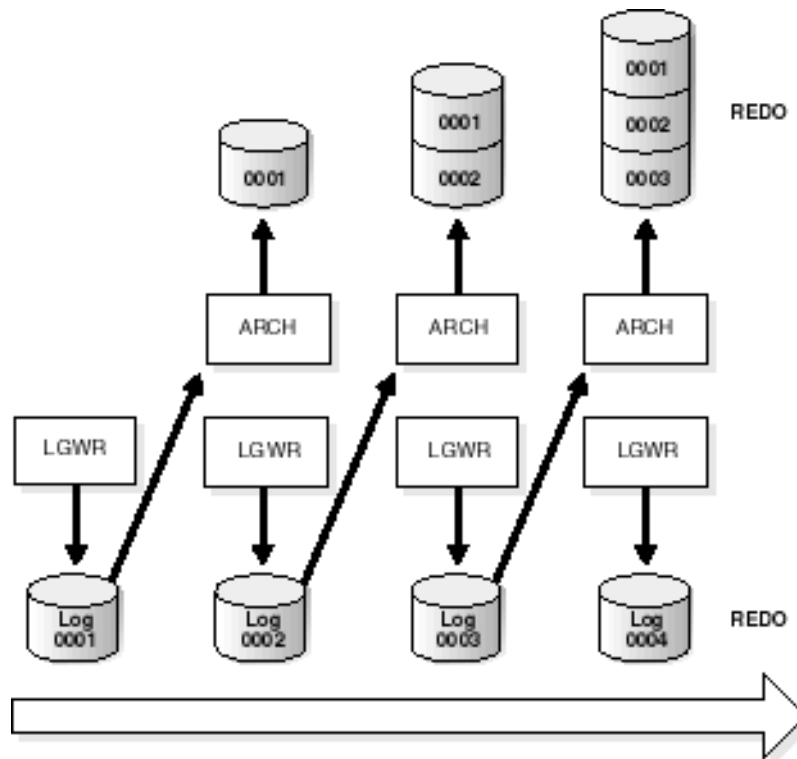
また、NOARCHIVELOG モードでは、オンライン表領域のバックアップを実行できません。さらに、データベースを ARCHIVELOG モードで運用していたときに作られたオンライン表領域のバックアップも使えません。データベースがクローズしている状態で作られた全体バックアップだけが、NOARCHIVELOG モードで運用するデータベースの復元に使えます。したがって、NOARCHIVELOG モードでデータベースを運用する場合、データベースの全体バックアップが定期的に作られていることを確認してください。

データベースを ARCHIVELOG モードで稼働する

データベースを ARCHIVELOG モードで稼働すると、オンライン REDO ログのアーカイブは使用可能になります。データベースの制御ファイルに格納されている情報は、満杯のオンライン REDO ログ・ファイルのグループがアーカイブされるまでは、LGWR でこのグループを使えないことを示します。満杯のグループは、ログ・スイッチの発生直後（グループがアクティブではなくなった時点）に、アーカイブを実行するプロセスで使えるようになります。アーカイブを実行するプロセスでは、ログ・スイッチのチェックポイントが完了するのを待たずに、アクティブではないグループにアクセスしてアーカイブできます。

図 23-1 に、満杯のグループ（この図では ARCH）をアーカイブするプロセスによってデータベースのオンライン REDO ログが生成される様子を示します。

図 23-1 ARCHIVELOG モードでオンライン REDO ログ・ファイルを使う



ARCHIVELOG モードでは、ディスク障害やインスタンス障害から完全に回復できます。これは、データベースへの変更がすべて、アーカイブ済み REDO ログに永続的に保存されるためです。

分散データベース内のデータベースをすべて ARCHIVELOG モードで運用している場合、調整した分散データベース回復を実行できます。ただし、分散データベース内のデータベースのどれかが NOARCHIVELOG モードで運用されている場合は、（すべてデータベースの整合性を維持するために）グローバルな分散データベースの回復は、NOARCHIVELOG モードで運用しているデータベースの最新の全体バックアップによって制限されます。

また、ARCHIVELOG モードでは、データベース全体または一部のバックアップや回復作業をしながら、データベース全体をオープンし、通常の用途で使えます。アーカイブ済み REDO ログ・ファイルを管理するには追加の管理操作をする必要があること、および ARCHIVELOG モードでデータベースを運用している場合にアーカイブ済み REDO ログ・ファイルを格納するための専用のテープ・ドライブまたは追加のディスク領域が必要になることに注意してください。

また、満杯のオンライン REDO ログ・グループのアーカイブ方法も決定しなければなりません。満杯のオンライン REDO ログ・ファイルのアーカイブを、自動または手動できるように設定できます。

関連項目： REDO ログ・ブロックをアーカイブするときに、これらのブロックを検証するように Oracle を構成することもできます。詳細は、5-13 ページの「REDO ログ・ファイル内のブロックの検証」を参照してください。

アーカイブの切り替え

ここでは、アーカイブについて説明します。トピックは次のとおりです。

- 初期データベース・アーカイブ・モードを設定する
- データベース・アーカイブ・モードを変更する
- 自動アーカイブを使用可能にする
- 自動アーカイブを使用禁止にする
- 手動アーカイブを実行する

データベースの最初のアーカイブ・モードはデータベース作成の一部として設定します。多くの場合、データベースを作るときに生成される REDO 情報をアーカイブする必要はなく、またデータベースを迅速に作るため NOARCHIVELOG モード（デフォルト）を指定します。データベースを作ったら、最初のアーカイブ・モードを変更するかどうかを決定します。

データベースの作成後、必要に応じてデータベースのアーカイブ・モードを切り替えることができます。ただし、通常はアーカイブ・モードを切り替えてはいけません。

関連項目： Oracle をインストールするときにデータベースが自動的に作られる場合、そのデータベースの最初のアーカイブ・モードは、オペレーティング・システムによって異なります。使っているオペレーティング・システム固有の Oracle のマニュアルを参照してください。

初期データベース・アーカイブ・モードを設定する

データベースを作るときに CREATE DATABASE 文に REDO ログのアーカイブ・モードを初期設定します。ARCHIVELOG または NOARCHIVELOG のいずれかを設定します。NOARCHIVELOG モードがデフォルトです。

関連項目： データベースの作成の詳細は、第 2 章「Oracle データベースの作成」を参照してください。

データベース・アーカイブ・モードを変更する

アーカイブ・モードを NOARCHIVELOG モードまたは ARCHIVELOG モードに切り替えるには、ARCHIVELOG オプションまたは NOARCHIVELOG オプション付きで SQL コマンド ALTER DATABASE を指定します。次の文は、データベースのアーカイブ・モードを NOARCHIVELOG から ARCHIVELOG に切り替えます。

```
ALTER DATABASE ARCHIVELOG;
```

データベースのアーカイブ・モードを切り替える前に、次の操作を実行します。

1. データベースのインスタンスを停止する。

データベースがオープンされている場合、データベースのアーカイブ・モードを切り替える前にクローズしてデスマウントしなければなりません。メディア回復を必要とするデータ・ファイルがある場合は、アーカイブを使用禁止にできません。

2. データベースをバックアップする。

データベースを変更する前に、データベース・データを保護するため必ずバックアップしてください。

3. オペレーティング・システム固有のステップを実行する（オプション）。

このような手順では、既存の Enterprise Manager を終了し、Oracle が満杯のグループをアーカイブする方法を設定する可能性もあります。これが終了したら Enterprise Manager を再起動して、ステップ 4 に進みます。

4. 新しいインスタンスを起動し、データベースをマウントし、オープンはしない。

アーカイブを使用可能または使用禁止にするには、データベースをマウントして、オープンしないようにする必要があります。

注意： Oracle Parallel Server を使っている場合、データベースのアーカイブ・モードを切り替えるには、1 つのインスタンスを使ってデータベースを排他的にマウントしなければなりません。

5. データベースのアーカイブ・モードを切り替える。

ALTER DATABASE コマンドを使ってデータベースのアーカイブ・モードを切り替えた後、データベースをオープンして通常の操作を実行できます。ARCHIVELOG モードに切り替えた場合、アーカイブ・オプションも設定しなければなりません。満杯のオンライン REDO ログ・ファイル・グループが Oracle により自動アーカイブされるようにするかどうかを決めてください。

オペレーティング・システムによっては、満杯のグループをアーカイブするために、この時点でさらにいくつかの手順を実行しなければならない場合があります。使っているシステムの詳細は、オペレーティング・システム固有の Oracle マニュアルを参照してください。

データベースのバックアップの詳細は、『Oracle8 Server バックアップおよびリカバリ』を参照してください。

Oracle Parallel Server を使用している場合のアーカイブ・モードの切替えの詳細は、『Oracle8 Parallel Server 概要および管理』を参照してください。

自動アーカイブを使用可能にする

オペレーティング・システムが許可する場合には、オンライン REDO ログ・ファイルの自動アーカイブを使用可能にできます。このオプションを使うと、グループが満杯になってもグループをコピーするための操作は必要ありません。グループが満杯になると、Oracle が自動的にグループをアーカイブします。自動アーカイブはこのように便利なので、満杯になったオンライン REDO ログ・ファイルのグループのアーカイブには、通常自動アーカイブを選択します。

インスタンスの起動後に、自動アーカイブを使用可能にするには、管理者権限を使って Oracle に接続してください。

警告： Oracle は、データベースも ARCHIVELOG モードでないと、ログ・ファイルを自動的にアーカイブしません。

インスタンスの起動の前後に自動アーカイブを使用可能にできます。

関連項目：これが Oracle Server の有効なオプションであるかどうかを判断するには、オペレーティング・システム固有の Oracle のマニュアルを参照してください。

自動アーカイブを使用可能にするときは、アーカイブ済みの REDO ログのアーカイブ先とファイル名の形式を必ず指定します。23-11 ページの「アーカイブ済み REDO ログのファイル名フォーマットと宛先を指定」を参照してください。

自動アーカイブが使用可能になっている場合でも、手動アーカイブは可能です。23-8 ページの「手動アーカイブを実行する」を参照してください。

インスタンス起動時に自動アーカイブを使用可能にする

インスタンスを起動するたびに自動アーカイブを使用可能にするには、次のようにデータベースのパラメータ・ファイルの LOG_ARCHIVE_START パラメータを TRUE に設定します。

```
LOG_ARCHIVE_START=TRUE
```

新しい値は、次回データベースを起動するときに有効になります。

インスタンス起動後に自動アーカイブを使用可能にする

現行インスタンスを停止せずに、満杯のオンライン REDO ログ・グループの自動アーカイブを使用可能にするには、ARCHIVE LOG START を指定した SQL コマンド ALTER SYSTEM を使います。この場合、アーカイブ先を任意で指定できます。

```
ALTER SYSTEM ARCHIVE LOG START;
```

上記のどちらかのオプションを使うと、自動アーカイブを使用可能にするためにインスタンスを停止させる必要がなくなります。ただし、自動アーカイブが使用可能になった後にインスタンスを停止して再起動すると、そのインスタンスはパラメータ・ファイルの設定値に

よって再度初期化されます。この設定値では、自動アーカイブは使用可能になっていることもなっていないこともあります。

自動アーカイブを使用禁止にする

いつでもオンライン REDO ログ・グループの自動アーカイブを使用禁止にできます。ただし、一度自動アーカイブを使用禁止にすると、オンライン REDO ログ・ファイルのグループを適時手動でアーカイブしなければなりません。データベースが ARCHIVELOG モードで運用され、自動アーカイブが使用禁止になっている場合、オンライン REDO ログ・ファイルのグループがすべて満杯になりアーカイブを行わないと、LGWR はオンライン REDO ログ・グループのアクティブしていないグループを再使用して REDO ログ・エントリの書込みを続行できなくなります。したがって、必要なアーカイブがなされるまでデータベース操作は一時的に停止されます。

インスタンスの起動後に、自動アーカイブを使用禁止にするために、管理者権限および ALTER SYSTEM 権限を持っている必要があります。

インスタンス起動時またはその後で自動アーカイブを使用禁止にできます。

インスタンス起動時に自動アーカイブを使用禁止にする

データベース・インスタンスが起動するたびに、満杯のオンライン REDO ログ・グループの自動アーカイブを禁止するには、次のようにデータベースのパラメータ・ファイルの LOG_ARCHIVE_START パラメータを FALSE に設定します。

```
LOG_ARCHIVE_START=FALSE
```

新しい値は、次回データベースを起動するときに有効になります。

インスタンス起動後に自動アーカイブを使用禁止にする

現行インスタンスを停止せずに、満杯のオンライン REDO ログ・グループの自動アーカイブを使用可能にするには、ARCHIVE LOG STOP パラメータを指定した SQL コマンド ALTER SYSTEM を使います。次の文はアーカイブを停止します。

```
ALTER SYSTEM ARCHIVE LOG STOP;
```

自動アーカイブを使用禁止にする際に、ARCH が REDO ログ・グループをアーカイブしている場合、ARCH は現在のグループのアーカイブを完了し、次の満杯のオンライン REDO ログ・グループのアーカイブは開始されません。

自動アーカイブを使用禁止にするためにインスタンスを停止する必要はありません。ただし、自動アーカイブが使用禁止になった後にインスタンスを停止して再起動すると、そのインスタンスはパラメータ・ファイルの設定値によって再度初期化されます。この設定値では、自動アーカイブは使用可能になっていることもなっていないこともあります。

手動アーカイブを実行する

データベースを ARCHIVELOG モードで運用している場合、満杯のオンライン REDO ログ・ファイルのアクティブでないグループをアーカイブする必要があります。自動アーカイブの設定にかかわらず、オンライン REDO ログのグループを手動でアーカイブできます。

- 自動アーカイブが使用可能にされていない場合、満杯のオンライン REDO ログ・ファイルのグループを適時手動でアーカイブしなければならない。オンライン REDO ログ・グループがすべて満杯になってアーカイブされない場合、LGWR はオンライン REDO ログ・メンバーのアクティブでないグループを再使用して、REDO ログエントリの書込みを続行できません。したがって、必要なアーカイブがなされるまでデータベース操作は一時的に停止されます。
- 自動アーカイブは使用可能になっているが、満杯のオンライン REDO ログ・メンバーのアクティブでないグループを別の位置にもう一度アーカイブしたい場合、手動アーカイブを使える。(しかし、手動アーカイブの終了前に、REDO ログ・グループを再使用してファイルを上書きするようインスタンスによって決められることがあります。このような場合、Oracle は ALERT ファイルにエラー・メッセージを入れます。)

満杯のオンライン REDO ログ・グループを手動でアーカイブするには、管理者権限を使って接続してください。

アクティブでない満杯のオンライン REDO ログ・メンバーのグループを手動でアーカイブするには、ARCHIVE LOG 句を指定した SQL コマンド ALTER SYSTEM を使います。

次の文は、アーカイブされていないログ・ファイルをすべてアーカイブします。

```
ALTER SYSTEM ARCHIVE LOG ALL;
```

関連項目：Oracle Parallel Server を使っている場合を除いて、手動または自動アーカイブを実行する際にスレッドを指定する必要はありません。詳細は、『Oracle8 Parallel Server 概要および管理』を参照してください。

アーカイブの調整

ここでは、アーカイブ・プロセスの調整について説明します。トピックは次のとおりです。

- システム・パフォーマンスへの影響を最小にする
- アーカイブを高速にする

ほとんどのデータベースでは、アーカイブ・プロセスが、全体のシステムパフォーマンスに影響を及ぼすことはありません。ただし、大規模なデータベース・サイトでは、アーカイブがシステム・パフォーマンスに影響を及ぼす可能性もあります。一方では、アーカイバが非常に高速な処理を実行すると、CPU サイクルがアーカイブに使われるため、アーカイバの実行中は全体のシステム・パフォーマンスが低下する可能性があります。もう一方では、アーカイバが極端に低速で実行すると、システム・パフォーマンスへの悪影響はほとんどありませんが、REDO ログ・ファイルのアーカイブに時間がかかるようになります。それらがアー

カイブされるのを待機するために、すべての REDO ログ・グループが使用不可能になると、ボトルネックになる可能性があります。

これら大規模なデータベース・サイトに対しては、アーカイブを調整して、ボトルネックとならずに可能な限り低速でアーカイブされるように、または実質的にシステム・パフォーマンスを低下させずに可能な限り高速でアーカイブされるようにできます。そのためには、初期化パラメータ LOG_ARCHIVE_BUFFERS (アーカイブに割り当てられるバッファ数) と LOG_ARCHIVE_BUFFER_SIZE (そのようなバッファのサイズ) を調整します。

注意： LOG_ARCHIVE_BUFFERS または LOG_ARCHIVE_BUFFER_SIZE の値を変更すると、新しい値は次回インスタンスを起動するときに有効になります。

システム・パフォーマンスへの影響を最小にする

システムが REDO ログを待機させずに、できる限りアーカイバの処理を低速にするには、まずアーカイブ・バッファの数 (LOG_ARCHIVE_BUFFERS) を 1 に設定し、各バッファのサイズ (LOG_ARCHIVE_BUFFER_SIZE) を可能な最大値に設定します。

アーカイバの作動中にシステムのパフォーマンスが著しく低下する場合には、アーカイバの実行時にシステム・パフォーマンスが低下しなくなるまで、LOG_ARCHIVE_BUFFER_SIZE を小さく調整してください。

注意： アーカイブをなるべく低速に設定したいが、REDO ログ・ファイルを再使用できるようになる前にそれらがアーカイブされるまで Oracle が待つ必要が頻繁に生じるときは、REDO ログ・ファイル・グループをさらに追加することを考慮してください。追加のグループによって、グループが常に Oracle に対して使用可能にできます。

アーカイブを高速にする

アーカイブ・パフォーマンスを向上させるには (たとえば、入力をテープ・ドライブに流す場合など)、複数のアーカイブ・バッファを使って、アーカイバ・プロセスによる出力ログへの書込みとアーカイブ・ログの読取りが同時に実行できるようにします。

LOG_ARCHIVE_BUFFERS を 2 に設定できますが、非常に高速なテープ・ドライブに対しては、3 以上に設定してもかまいません。アーカイブ・バッファのサイズは適当な数に設定し、それからシステム・パフォーマンスを損なわずに、アーカイブが望ましい速度になるまで大きくしてください。

関連項目： この最大設定値は、オペレーティング・システムによって異なります。詳細は、オペレーティング・システム固有の Oracle のマニュアルを参照してください。

これらのパラメータの詳細は、『Oracle8 Server リファレンス・マニュアル』を参照してください。

アーカイブ状態を表示

現行のアーカイブ・モードを確認するには、V\$DATABASE ビューを問い合わせます。

```
SELECT log_mode FROM sys.v$database;
```

```
LOG_MODE
-----
NOARCHIVELOG
```

また、データ・ディクショナリ・ビュー V\$ARCHIVE と V\$LOG にはデータベースの状態情報が含まれています。次の問合せは、データベースのログ・グループをすべて表示し、アーカイブすべき残りのグループを示します。

```
SELECT group#, archived
FROM sys.v$log;
```

```
GROUP#      ARC
-----  ---
1             YES
2             NO
```

LIST パラメータを指定した ARCHIVE LOG コマンドは、接続しているインスタンスのアーカイブ情報も表示します。

```
ARCHIVE LOG LIST;
```

```
Database log mode                ARCHIVELOG
Automatic archival               ENABLED
Archive destination              destination
Oldest online log sequence      30
Next log sequence to archive    32
Current log sequence number     33
```

この表示は、現行インスタンスの REDO ログの設定に関するすべての必要な情報を以下のように示します。

- このデータベースは現在 ARCHIVELOG モードで運用されている。
- 自動アーカイブは使用可能になっている。
- (オペレーティング・システムに固有の) アーカイブ済み REDO ログのアーカイブ先は *destination* である (LOG_ARCHIVE_DEST または上書きされたアーカイブ先に対応する)。
- 満杯の最も古いオンライン REDO ログ・グループの順序番号は 30 である。
- 次にアーカイブされる満杯のオンライン REDO ログ・グループの順序番号は 32 である。
- 現行のオンライン REDO ログ・ファイルの順序番号は 33 である。

Next log sequence to archive 以上、*Current log sequence number* 未満の順序番号を持つオンライン REDO ログ・グループをすべてアーカイブしなければなりません。この例では順序番号が 32 のオンライン REDO ログ・ファイルをアーカイブしなければなりません。

アーカイブ済み REDO ログのファイル名フォーマットと宛先を指定

データベースを ARCHIVELOG モードで運用している場合、アーカイブ済み REDO ログのファイル名の形式とアーカイブ先を、Oracle に指定しなければなりません。この結果、一意に命名されたアーカイブ済み REDO ログ・ファイルが、自動または手動アーカイブによって適切な記憶領域に作られます。

アーカイブ済み REDO ログ・ファイルに名前を付けるには、LOG_ARCHIVE_FORMAT 初期化パラメータに一意の名前を指定します。ファイル名の形式はオペレーティング・システムに応じて異なります。通常はテキスト文字列、1 つまたは複数のパラメータ、ファイル名の拡張子が使われます。オンライン REDO ログ・グループがアーカイブされる際、アーカイブ・プロセスによって、指定したパラメータの戻り値がテキスト文字列に連結され、一意に識別されるアーカイブ済み REDO ログ・ファイルが作られます。パラメータには、オペレーティング・システムに応じて異なる上限がそれぞれあります。

表 23-1 に、ファイル名の形式で入力できるパラメータと、アーカイブ・プロセスで作られるファイル名に対するそのパラメータの効果の例を示します。

表 23-1 アーカイブ済み REDO ログ・ファイル名の形式パラメータ

パラメータ	説明	例 ¹
%T	スレッド番号、スレッド番号の左の空白はゼロが埋まる	arch0000000001
%t	スレッド番号、空白は埋めない	arch1
%S	ログ順序番号 左にゼロが埋め込まれる	arch0000000251
%s	ログ順序番号、空白は埋めない	arch251
¹ パラメータには LOG_ARCHIVE_FORMAT=arch% と設定し、パラメータの上限はすべて 10 文字と想定します。		

アーカイブ済み REDO ログのファイル名を必要に応じて変更するための、様々なオプションが用意されています。たとえばオペレーティング・システムのソート・アルゴリズムによる並び替えを考慮して、ファイル名を付けることができます。

%T および %t は、Oracle Parallel Server を使っている場合に限り有効です。Parallel Server 以外の構成で、アーカイブ済み REDO ログ・ファイルを一意に識別するためには、%S または %s のどちらかを使ってください。次に、代表的なアーカイブ済み REDO ログのファイル名の形式を示します。

LOG_ARCHIVE_FORMAT = arch%S.arc

ここで、arch はファイル名、%S はゼロが埋め込まれたログ順序番号パラメータ、.arc はファイル拡張子です。たとえば %S の上限を 4 と想定すると、この形式に基づいて次の形式のアーカイブ済み REDO ログのファイル名が生成されます。

```
arch0001.arc  
arch0002.arc  
arch0003.arc  
.  
.
```

アーカイブ・ファイル名の形式を指定する際には、オペレーティング・システムのファイル名の最大長を考慮してください。ARCH またはユーザー・プロセスでファイルをアーカイブする場合に、指定されたファイル名が長すぎると、そのプロセスによるファイルのアーカイブは失敗します。

注意： LOG_ARCHIVE_FORMAT でアーカイブ済みファイル名の形式が指定されていないと、デフォルトのファイル名形式が使われます。デフォルトの形式はオペレーティング・システムによって異なります。

アーカイブ済み REDO ログのアーカイブ先もオペレーティング・システムによって異なります。ほとんどのオペレーティング・システムでは、アーカイブ済み REDO ログ・ファイルのアーカイブ先はディスク・ドライブとファイル・ディレクトリになります。Oracle Server によって許可されていれば、Oracle 専用のテープ・ドライブをアーカイブ先に指定して、満杯の REDO ログ・ファイルをアーカイブできます。

アーカイブ済み REDO ログのアーカイブ先は、インスタンス起動時に LOG_ARCHIVE_DEST 初期化パラメータで指定します。インスタンス起動中に変更できません。

- データベースのパラメータ・ファイルを編集して LOG_ARCHIVE_DEST パラメータにアーカイブ先を指定した場合、新しいパラメータ・ファイルを読み込むために、現行インスタンスを停止してから起動する。
- 現行インスタンスを停止できないが、自動アーカイブに対してアーカイブ済み REDO ログのアーカイブ先を指定または変更しなければならない場合は、ALTER SYSTEM ARCHIVE LOG START 'destination' 文を使って、自動アーカイブのアーカイブ先を変更する。
- 手動アーカイブを実行する場合、アーカイブ済み REDO ログのデフォルトのアーカイブ先は、指定したアーカイブ先によって上書きされる。ただし、自動アーカイブは現行のアーカイブ先を引き続き使います。アーカイブ先を指定しないと、インスタンスの起動に使われるパラメータ・ファイルの LOG_ARCHIVE_DEST パラメータに指定したアーカイブ先が自動的に使われます。LOG_ARCHIVE_DEST パラメータでアーカイブ先が指定されていないなら、デフォルトのアーカイブ先（オペレーティング・システムによって異なる）が使われます。

関連項目：初期化パラメータ LOG_ARCHIVE_FORMAT と LOG_ARCHIVE_DEST、およびデフォルトのアーカイブ済み REDO ログ・ファイル名の形式とアーカイブ先の詳細は、使っているオペレーティング・システム固有の Oracle のマニュアルを参照してください。

ファイル名の形式パラメータおよび「スレッド」の詳細は、『Oracle8 Parallel Server 概要および管理』を参照してください。

スキーマ・オブジェクトの領域の見積もり

この付録では、特定のスキーマ・オブジェクトの領域を概算する際に役立つ公式を説明します。見積もりを計算する際に使う定数は、オペレーティング・システムによって異なります。

注意： これらの公式は、スキーマ・オブジェクトのサイズの見積もりには、有用ですが、計算結果は近似値なので、実際の数値とは異なる場合があります。

クラスタ化されてない表に必要な領域の見積もり

ここでは、クラスタ化されてない表にデータを挿入する際に、そのデータの保持に必要な全体のデータ・ブロック数を見積もる手順について説明します。この計算例では、並行処理を前提としていないため、ユーザ - が挿入と同時に削除や更新をしないものとします。

注意： このシナリオは、ユーザ - が削除や更新を実行せずに行を挿入する場合にだけ適用されます。

通常、一連の行を格納するのに必要な領域は、表内で更新や削除がなされると、この計算結果を超えてしまいます。複雑な作業に必要な実際の領域を求める場合は、経験に基づいて判断してから、表の行数によって調整する方法が適切です。一般に、1つのデータ・ブロックに対する同時実行のアクティビティの数が増加すると、余分なオーバーヘッド（トランザクション・レコードによる）が生じます。このため、経験から判断した値を調整する際にはこのようなアクティビティを考慮することが重要になります。

クラスタ化されてない表に必要な領域を計算する手順

- 1. 全体のブロック・ヘッダー・サイズを計算する。
- 2. データ・ブロックあたりの使用可能なデータ領域を計算する。
- 3. 行あたりの使用領域を計算する。
- 4. データ・ブロックに収める全体の行数を計算する。

ステップ 1: 全体のブロック・ヘッダー・サイズの計算

データ・ブロック・ヘッダーが必要とする領域は、次の公式によって算出される値になります。

ヘッダーより後の領域 (hsize)
=
DB_BLOCK_SIZE - KCBH - UB4 - KTBBH - ((INITRANS - 1) * KTBIT) - KDBH

ここで

DB_BLOCK_SIZE	V\$PARAMETER ビューに表示されるデータベース・ブロック・サイズ。
KCBH、UB4、KTBBH、KTBIT、KDBH	定数。これらの各サイズは、V\$TYPE_SIZE ビューのエントリから選択することによって取得できます。
INITRANS	表に割り当てられるトランザクション・エントリの初期数。

ステップ 2: データ・ブロックあたりの使用可能なデータ領域の計算

PCTFREE によって指定される、各データ・ブロック内でデータ用に確保されている領域は、次のように計算します。

使用可能データ領域 (availspace)
=
CEIL(hsize * (1 - PCTFREE/100)) - KDBT

ここで

CEIL	計算結果が整数になるよう切り上げる。
PCTFREE	表内で更新のために確保する領域の割合。
KDBT	定数。そのサイズは、V\$TYPE_SIZE ビューからエントリを選択することによって取得できます。

注意： KDBT の値を見つけれない場合は、かわりに UB4 の値を使ってください。

ステップ 3: 行あたりに使われる領域の計算

行あたりに使われる領域を計算する手順には、いくつかのステップがあります。

まず最初に、バイト長を加えた、列のサイズを計算する必要があります。

バイト長を加えた列サイズ
=
列サイズ + (列サイズ < 250 なら 1、それ以外は 3)

注意： 表の各列ごとに avg(vsize(colname)) を選択することにより、経験に基づいて列のサイズを決定することもできます。

次に、行のサイズを計算します。

行サイズ
=
行ヘッダー (3 * UB1) + バイト長を加えた列サイズの合計

これで、行あたりに使われる領域を計算できます。

行あたりに使われる領域 (row space)
=
MAX(UB1 * 3 + UB4 + SB2, 行サイズ) + SB2

ここで

UB1、UB4、SB2

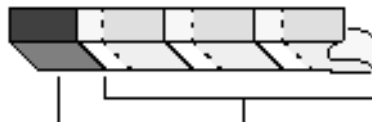
定数。これらの各サイズは、V\$TYPE_SIZE
ビューからエントリを選択することによって
取得できます。

行あたりの領域が、データ・ブロックあたりの使用可能な領域を上回り、更新用に確保されている領域がないデータ・ブロックあたりの使用可能な領域（たとえば、PCTFREE=0 が指定されている場合の使用可能な領域など）を下回る場合には、各行はそれぞれのブロックに格納されることになります。

行あたりの領域が、更新用に確保されている領域がないデータ・ブロックあたりの使用可能な領域を上回る場合、複数の行が表に挿入されると、それらは連鎖されて 2 つ以上の部分を構成するため、この記憶領域のオーバーヘッドが増加します。

図 A-1 は表の行の中の要素を表したものです。

図 A-1 行サイズの計算



ステップ 4: データ・ブロックに収める全体の行数の計算

次の公式を使うと、データ・ブロックに収める全体の行数を計算できます。

ブロック中の行数
=
`FLOOR(available / rowsize)`

ここで

`FLOOR`

計算結果が整数になるよう切り捨てる。

なお、この手順では妥当な表のサイズを見積もっているものであって、正確なブロック数またはバイト数を算出しているわけではありません。対応する `CREATE TABLE` 文に `INITIAL` 記憶領域パラメータ（表の初期エクステンツのサイズ）を指定するときに、この公式を使って見積もった表のサイズを使えます。

関連項目：この手順で与えられている定数と大幅な違いがある場合は、使っているオペレーティング・システム固有の Oracle のマニュアルを参照してください。

使用中の表の領域要件

表を作ってから使っているうちに、表に必要な領域は、計算で出された見積もりよりも大きくなっていくのが普通です。Oracle によるデータベース内の空き領域の管理方法によっては、さらに大きい領域が必要になります。

索引の領域の見積もり

次の手順における計算は、索引を構成する列の平均の列長に依存しています。そのため、表の各行の中の列長が、索引付きの列に関して比較的一定である場合に、次の手順によって計算される見積もりがより正確なものになります。また、次の係数によって、その計算が正確に調整できます。

- ポート固有の変数
- ブランチ・ブロック部分の 5% 乗数（任意の変数）
- 内部断片化率

索引の領域を見積もる手順

- 1. 全体のブロック・ヘッダー・サイズを計算する。
- 2. データ・ブロックあたりの使用可能なデータ領域を計算する。
- 3. 平均索引値の結合列の長さを計算する。
- 4. 全体の平均索引値のサイズを計算する。
- 5. 索引に必要なブロック数とバイト数を計算する。

注意： 最終的な見積もりを求めるために、いくつかの計算が必要です。
また、与えられる定数の一部（* で示されている）はオペレーティング・システムによって異なります。見積もりの結果は実際の値と大きく異なってはいけません。

関連項目：次の手順で与えられている定数と大幅な違いがある場合は、使っているオペレーティング・システム固有の Oracle のマニュアルを参照してください。

ステップ 1: 全体のブロック・ヘッダー・サイズの計算

図 A-2 に次の計算で使う索引ブロックの要素を示します。索引データを収めるために、ブロックのデータ・ブロック・ヘッダーに必要な領域は、次の公式で与えられます。

ブロック・ヘッダーのサイズ = 固定長ヘッダー + 可変長トランザクション・ヘッダー

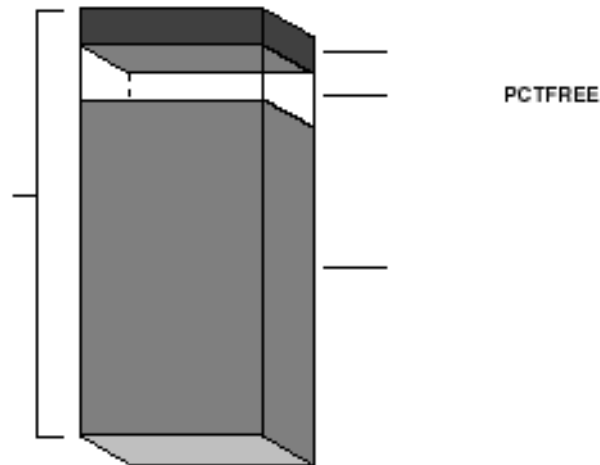
ここで

固定長ヘッダー	113 バイト
可変長トランザクション・ヘッダー	24*I。I は索引の INITRANS の値。

INITRANS = 2（索引のデフォルト）と想定すると、前の公式は簡略化できます。

ブロック・ヘッダー = 113 + (24*2) バイト
= 161 バイト

図 A-2 索引の領域の計算



ステップ 2: データ・ブロックあたりの使用可能なデータ領域の計算

各データ・ブロック内で索引データ用に確保されている領域（PCTFREE によって指定されるもの）は、ブロック・サイズからブロック・ヘッダーを引いた部分の割合として計算します。

ブロックあたりの
使用可能なデータ領域
$$= (\text{ブロック・サイズ} - \text{ブロック・ヘッダー}) - ((\text{ブロック・サイズ} - \text{ブロック・ヘッダー}) * (\text{PCTFREE}/100))$$

データベースのブロック・サイズはデータベース作成中に設定されます。必要であれば、Server Manager コマンド SHOW を使って判別できます。

```
SHOW PARAMETERS db_block_size;
```

与えられた索引に対して 2KB のデータ・ブロック・サイズと PCTFREE=10 を想定すると、その索引に割り当てられる、データ・ブロック内の新しいデータに対する全体の領域は、次のようになります。

ブロックあたりの使用可能データ領域

$$\begin{aligned}
 &= (2048 \text{ バイト} - 161 \text{ バイト}) - ((2048 \text{ バイト} - 161 \text{ バイト}) * (10/100)) \\
 &= (1887 \text{ バイト}) - (188.7 \text{ バイト}) \\
 &= 1887 \text{ バイト} - 188.7 \text{ バイト} \\
 &= 1698.3 \text{ バイト}
 \end{aligned}$$

ステップ 3: 結合列の長さの計算

ステップ 4 で全体の行サイズを計算する前に、索引の平均値で必要な領域を計算する必要があります。このステップは、索引列の平均結合列の長さを計算する必要があるという点以外は、表サイズを計算する手順のステップ 3 と同じです。

ステップ 4: 全体の平均索引値サイズの計算

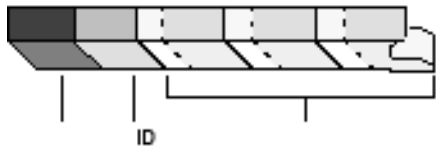
図 A-3 に次の計算で使われる索引エントリの要素を示します。平均的な索引エントリの結合列の長さが計算されると、次の公式によって全体の平均エントリ・サイズを計算できます。

エントリあたりのバイト数 = エントリ・ヘッダー + ROWID 長 + F + V + D

各項目は、次のとおりです。

エントリ・ヘッダー	2 バイト
ROWID 長	6 バイト
F	127 バイト以下のデータを格納するすべての列の長さの合計 (バイト数)。このタイプの列に必要な長さは 1 バイトです。
V	127 バイトを超えるデータを格納するすべての列の長さの合計 (バイト数)。このタイプの列に必要な長さは 2 バイトです。
D	全索引列の結合データ領域 (ステップ 3 による)。

図 A-3 索引エントリの平均サイズの計算



たとえば、D を計算すると 22 バイトとなり、索引が 3 つの VARCHAR(10) 列によって構成されているとした場合、その索引の平均エントリ・サイズの合計は、次のようになります。

平均エントリ・サイズ = 2 + 6 + (1 * 3) + (2 * 0) + 22 バイト
= 33 バイト

注意：一意でない索引の場合は、ROWID が別の列と見なされて、長さバイトが 1 個必要になります。

ステップ 5: ブロック数とバイト数の計算

次の公式を使って、索引の格納に必要なブロック数を計算します。

索引のブロック数 =

$$1.05 \times \frac{\# \text{ null ではない行}}{\text{FLOOR}(\text{ブロックあたりの使用可能なデータサイズ} / \text{平均エンリ・サイズ})}$$

注意： この結果に加えた追加の 5% (つまり、乗数 1.05) は、索引のランチ・ブロックに必要な余計な領域のための計算です。

たとえば、前の例で、索引を付けられた表のうち、索引を構成する列の値が NULL でない行の数が 10000 行になるとします。

索引のブロック数 =

$$1.05 \times \frac{10000 \times 33 \text{ バイト}}{\text{FLOOR}((1700 \text{ バイト} / 33 \text{ バイト}) \times (33 \text{ バイト}))}$$

この結果は 204 ブロックになります。バイト数は、ブロック数にデータ・ブロック・サイズを掛けることによって計算できます。

なお、この手順では妥当な索引のサイズを見積もっているのであって、正確なブロック数、またはバイト数を算出しているわけではありません。索引のサイズを見積もったなら、対応する CREATE INDEX 文で INITIAL 記憶領域パラメータ (索引の初期エクステントのサイズ) を指定するときに、この情報を使えます。

索引作成に必要な一時領域

ロードする表に対して索引を作るとき、索引をソートするために一時セグメントが作られます。索引のソートに必要な領域の量は一定ではありませんが、索引サイズの 110% にまでする可能性があります。

注意： NOSORT オプションを CREATE INDEX コマンドで指定すると、一時領域は必要ありません。ただし、クラスタ索引を作るときには、このオプションを指定できません。

クラスタで必要な領域の見積もり

次の手順では、クラスタ内の一連の表が必要とする初期領域を見積もる方法を具体的に説明します。この手順では、クラスタに必要な初期領域の容量だけを見積もります。これらの見積もりを使う場合は、次のようなことが見積もりの正確さに影響を及ぼす可能性がありますので注意してください。

- 後続 NULL は格納されず、長さバイトもない。
- 単一のデータ・ブロックよりも大きな列を持っている表だけでなく、行の挿入、更新、削除も、断片化と行断片の連鎖を引き起こす可能性がある。そのため、著しい断片化が起こった場合、次に示す見積もりは、必要となる実際の領域より小さくなる傾向にあります。

一度、次の手順を使って表のサイズを計算し、10 ~ 20 パーセントの予備領域を追加して、作業する表の初期エクステント・サイズを計算するとよいでしょう。

クラスタで必要となる領域を見積もる手順

1. 全体のブロック・ヘッダー・サイズと表データで使用可能な領域を計算する。
2. クラスタ・キーあたりの平均的な行の結合列の長さを計算する。
3. すべてのクラスタ化した表の平均的な行サイズを計算する。
4. 平均のクラスタ・ブロック・サイズを計算する。
5. クラスタに必要な全体のブロック数を計算する。

ステップ 1: 全体のブロック・ヘッダー・サイズと表データで使用可能な領域を計算する。

次の公式は、1 つのブロックの使用可能な領域の容量を戻します。

注意： 最終的な見積もりを求めるために、いくつかの計算が必要です。また、与えられる定数の一部（* で示されている）はオペレーティング・システムによって異なります。見積もりの結果は実際の値と大きく異なってはけません。次の手順で与えられている定数と大幅な違いがある場合は、使っているオペレーティング・システム固有の Oracle のマニュアルを参照してください。

ブロックの中でヘッダーの後の残り領域 (hspace)

= BLOCKSIZE - KCBH - UB4 - KTBH - KTBIT*(INITTRANS - 1) - KDBH

ここで KCBH、KTBBH、KTBIT、KDBH、UB4 のサイズは、v\$type_size 表から全列 (*) を選択すると求めることができます。

注意： これが上記のクラスタ・セグメントでなく、表セグメントの場合、表ディレクトリは単に 4 になります。

その後、次の公式に従って、表データ用に使用可能な領域を計算します。

表データとして使用可能な領域

$$= \text{hspace} * (1 - \text{PCTFREE} / 100) - 4 * (\text{NTABLES} + 1) * \text{ROWSINBLOCK}$$

ここで

BLOCKSIZE	データ・ブロックのサイズ。
INITTRANS	オブジェクトのトランザクション・エントリの初期数。
PCTFREE	更新のためブロック内に確保する領域の割合。
NTABLES	クラスタ内の表の数。
ROWS INBLOCK	ブロック内の行数。

ステップ 2: 行に必要な領域の計算

「クラスタ化されてない表で必要な領域の見積もり」にある手順のステップ 3 を使って、A-2 ページのこの値を計算してください。次の警告に注意してください。

- クラスタ内の各表の平均的な行が必要とするデータ領域を計算してください。たとえば、表 T1 と表 T2 を含むクラスタでは、両方の表の平均的な行サイズを計算してください。
- 上の計算のどちらにもクラスタ・キーが必要とする領域を含めないでください。ただし、ステップ 5 では、平均のクラスタ・キー値を格納するために必要な領域に注意してください。たとえば、クラスタ・キーを格納するために必要となる領域を含めないで、表 T1 の平均的な行が必要とするデータ領域を計算してください。
- 行ヘッダーが必要とする領域（つまり各列に対する長さバイト）も含めないでください。この領域は、次のステップで計算されます。

たとえば、2 つのクラスタ化した表が次の文で作られると想定します。

```
CREATE TABLE t1 (a CHAR(10), b DATE, c NUMBER(10,2))
CLUSTER t1_t2 (c);
```

```
CREATE TABLE t2 (c NUMBER(10,2), d CHAR(10))
CLUSTER t1_t2 (c);
```

なお、クラスタ・キーは各表の列 C です。

この例を検討すると、表 T1 の平均的な行 (D1) に必要な領域と表 T2 の平均的な行 (D2) に必要な領域は次のようになります。

$$\begin{aligned} D1 \text{ (平均行領域)} &= (a + b) \\ &= (10 + 7) \text{ バイト} \\ &= 17 \text{ バイト} \end{aligned}$$

$$\begin{aligned} D2 \text{ (平均行領域)} &= (d) \\ &= 10 \text{ バイト} \end{aligned}$$

ステップ 3: 全体の平均行サイズの計算

次の式に従って、クラスタ化した表の行が必要とする最小領域を計算できます。

$$S_n \text{ 行あたりバイト数} = \text{行ヘッダー} + F_n + V_n + D_n$$

各項目は、次のとおりです。

row header*	クラスタ化表の行あたり 4 バイト。
F_n	250 バイト以下のデータの列の長さの合計 (バイト数)。このタイプの列で必要な長さは 1 バイトです。
V_n	表 n のうち、250 バイトを超えるデータを格納するすべての列の長さの合計 (バイト数)。このタイプの列で必要な長さは 3 バイトです。
D_n	表 n のすべての列の結合データ領域 (ステップ 3 による)。

注意： クラスタ内の任意の表の変数 F または V に、クラスタ・キーの列の長さを含めてはなりません。この領域はステップ 5 で計算されます。

たとえば、クラスタ化した表 T1 と T2 の平均的な行サイズの合計は次のようになります。

$$\begin{aligned} S1 &= (4 + (1 * 2) + (3 * 0) + 17) \text{ バイト} \\ &= 23 \text{ バイト} \end{aligned}$$

$$\begin{aligned} S &= (4 + (1 * 1) + (3 * 0) + 10) \text{ バイト} \\ &= 15 \text{ バイト} \end{aligned}$$

注意: クラスタ化された行の絶対最小行サイズは 10 バイトであり、これはオペレーティング・システムによって異なります。そのため、表の全体の平均的な行サイズとして計算した値がこの絶対最小行サイズより小さい場合、平均的な行サイズとしてこの最小値を次の計算で使ってください。

ステップ 4: 平均のクラスタ・ブロック・サイズの計算

平均のクラスタ・ブロック・サイズを計算するには、最初にクラスタ・キーあたりの平均の行数を（すべての表に対して）見積もります。それが求まったら、次の公式を使って、平均のクラスタ・ブロック・サイズを計算してください。

平均クラスタ・ブロック・サイズ (バイト数) =
 $((R1 * S1) + (R2 * S2) + \dots + (Rn * Sn)) + \text{key header} + Ck + Sk + 2Rt$

ここで

Rn	クラスタ・キーに対応付けられる表 n の平均行数。
Sn	表 n の平均行サイズ（ステップ 4 を参照）。
キー・ヘッダー *	19。
Ck	クラスタ・キーの列の長さ。
Sk	平均クラスタ・キー値を格納するために必要な領域。
Rt	平均クラスタ・キー (R1 + R2 ...+ Rn) に対応付けられる行の総数。これは、ブロック内の各行のデータ・ブロック・ヘッダーに必要な領域のための計算です。

たとえば、表 T1 と T2 を含むクラスタを検討します。平均クラスタ・キーは、表 T1 では 1 行、表 T2 では 20 行です。また、クラスタ・キーのデータ型は NUMBER であり（列の長さが 1 バイト）平均は 4 桁（3 バイト）です。この情報と先の結果を検討すると、平均クラスタ・キー・サイズは次のようになります。

SIZE = ((1 * 23) + (20 * 15) + 19 + 1 + 3 + (2 * 21)) バイト
 = 388 バイト

CREATE CLUSTER コマンドを使ってクラスタを作るときに、見積もったサイズを SIZE オプションに指定してください。こうして、平均クラスタ・キーとその対応付けられた行を保持するのに必要となる領域を指定します。Oracle は、ある特定のデータ・ブロックに割り当てられるクラスタ・キーの数を制限するために、SIZE の値を使います。平均のクラスタ・キー・サイズ (SIZE) を見積もった後、クラスタ・キーに必要な領域を高めの見積もりで計算するために、予測される平均サイズより少し大きい SIZE を選択してください。

データ・ブロックに入るクラスタ・キーの数を見積もるには、次の公式を使ってください。この式では、使用可能なデータ領域についてステップ 2 で計算した値、および平均クラスタ・キーに対応する行数 (Rt)、SIZE を使います。

ブロックあたりのクラスタ・キー数
= FLOOR(使用可能データ領域 + 2R / SIZE + 2Rt)

たとえば、SIZE を (前のステップで計算した 388 バイトを切り上げて) 400 バイトとし、Rt の見積もりを 21、データ・ブロックあたり使用可能な領域を 1742-2R バイト (ステップ 2) とすると、結果は次のようになります。

ブロックあたりのクラスタ・キー数
= FLOOR((1936 - 2R + 2R) / (400 + 2 * 21))

= FLOOR(1936 / 442)
= FLOOR(4.4)
= 4

ステップ 5: 全体のブロック数の計算

クラスタのブロック数の合計を計算するには、まずクラスタ内のクラスタ・キーの数を見積もらなければなりません。それを計算したなら、次の公式を使ってクラスタに必要なブロック数の合計を計算してください。

ブロック数 = CEIL(クラスタ・キー数 / ブロックあたりクラスタ・キー数)

注意 : テスト・データベースがある場合、ANALYZE コマンドで生成される統計情報を使うことによって、クラスタ・キー値の数がわかります。17-3 ページの「表、索引、クラスタの分析」を参照してください。.

たとえば、T1_T2 クラスタ内に約 500 のクラスタ・キーがあるとします。

T1_T2 のブロック数 = CEIL(500/3)
= CEIL(166.7)
= 167

ブロック数をバイト数に変換するには、ブロック数にデータ・ブロック・サイズを掛けてください。

この手順では妥当なクラスタのサイズを見積もっているのであって、正確なブロック数またはバイト数を算出しているわけではありません。クラスタの領域を見積もったならば、対応する CREATE CLUSTER 文の中で INITIAL 記憶領域パラメータ (クラスタの初期エクステンツのサイズ) を指定するときに、この情報を使えます。

使用中のクラスタ化した表の領域要件

クラスタ化した表を作って使っているうちに、表に必要な領域は、前の部分で計算した見積もりより大きくなるのが普通です。Oracle がデータベース内の空き領域を管理する方法によって、より多くの領域が必要となります。

ハッシュ・クラスタで必要となる領域の見積もり

索引クラスタの場合と同じように、ハッシュ・クラスタ内のデータに必要な記憶領域を見積もることは重要です。次の注意点も考慮しながら、-10 ページの " クラスタで必要となる領域の見積もり " で説明されている手順を使ってください。

- 手順の目標の 1 つは各クラスタ・キーの SIZE を決定することです。ただし、ハッシュ・クラスタの場合は、各ハッシュ・キーの SIZE を決定することも一つの目標となります。そのため、クラスタ・キー値あたりの行数を検討するだけでなく、クラスタ内のハッシュ・キー上のクラスタ・キーの分布も検討する必要があります。
- ステップ 3 では、クラスタ・キー値のために必要な領域を必ず含めてください。索引クラスタとは違って、ハッシュ・クラスタ内に入れられる各行とともにクラスタ・キー値が格納されます。
- ステップ 5 では、クラスタ・キー・サイズではなくハッシュ・キー・サイズの平均を計算しています。そのため、各ハッシュ値に何個のクラスタ・キーが対応するかを考慮する必要があります。また、クラスタ・キー値 (Ck) のために必要な追加領域は無視します。この値はすでにステップ 3 で計算されています (前項を参照)。

A

- ADD PARTITION 句
 - ALTER TABLE コマンド, 11-5
- ADMIN OPTION
 - 取り消す, 21-19
 - 説明, 21-16
- AFTER トリガー
 - 監査と~, 22-20
- ALERT ファイル
 - 位置, 4-14
 - 書き込む時期, 4-14
 - サイズ, 4-14
 - 使用, 4-13
 - セッション高水位標, 20-6
 - 説明, 4-13
- ALL_INDEXES ビュー
 - データを入れる, 17-5
- ALL_TAB_COLUMNS ビュー
 - データを入れる, 17-5
- ALL_TABLES ビュー
 - データを入れる, 17-5
- ALTER CLUSTER コマンド
 - ALLOCATE EXTENT オプション, 15-9
 - MAXTRANS オプション, 10-8
 - 索引クラスタに対して使う, 15-8
 - ハッシュ・クラスタの使用, 16-8
- ALTER DATABASE コマンド
 - ADD LOG MEMBER オプション, 5-6
 - ADD LOGFILE オプション, 5-5
 - ARCHIVELOG オプション, 23-4
 - DATAFILE...OFFLINE DROP オプション, 9-8
 - DROP LOGFILE MEMBER オプション, 5-10
 - DROP LOGFILE オプション, 5-8
 - MOUNT オプション, 3-7
 - NOARCHIVELOG オプション, 23-4
 - OPEN オプション, 3-7
 - RENAME FILE オプション
 - 複数の表領域のデータ・ファイル, 9-10
 - ユーザーが部分的に使えるデータベース, 3-6
- ALTER FUNCTION コマンド
 - COMPILE オプション, 17-23
- ALTER INDEX コマンド, 11-10
 - MAXTRANS オプション, 10-8
 - MOVE PARTITION 句, 11-4
 - REBUILD PARTITION 句, 11-4, 11-14
 - 説明, 14-8
- ALTER PACKAGE コマンド
 - COMPILE オプション, 17-24
- ALTER PROCEDURE コマンド
 - COMPILE オプション, 17-23
- ALTER PROFILE コマンド
 - COMPOSITE_LIMIT オプション, 20-19
 - リソース制限の変更, 20-18
- ALTER RESOURCE COST コマンド, 20-19
- ALTER ROLE コマンド
 - 認可方法の変更, 21-15
- ALTER ROLLBACK SEGMENT コマンド
 - OFFLINE オプション, 18-12
 - ONLINE オプション, 18-11, 18-12
 - PUBLIC オプション, 18-9
 - STORAGE 句, 18-9
 - 記憶領域パラメータの変更, 18-9
- ALTER SEQUENCE コマンド, 13-10
- ALTER SESSION コマンド
 - SET SQL_TRACE パラメータ, 4-13
- ALTER SYSTEM コマンド
 - ARCHIVE LOG ALL オプション, 23-8
 - ARCHIVE LOG START オプション、アーカイブ先の指定, 23-12

ARCHIVE LOG STOP オプション, 23-7
CHECKPOINT オプション, 5-12
ENABLE RESTRICTED SESSION オプション, 3-8
SET LICENSE_MAX_SESSIONS オプション, 20-4
SET LICENSE_MAX_USERS オプション, 20-5
SET LICENSE_SESSIONS_WARNING
 オプション, 20-4
SET MTS_DISPATCHERS オプション, 4-10
SET MTS_SERVERS オプション, 4-9
SET RESOURCE_LIMIT オプション, 20-21
SWITCH LOGFILE オプション, 5-12
ALTER TABLESPACE コマンド
 ADD DATAFILE パラメータ, 9-5
 ONLINE オプション
 例, 8-8
 READ ONLY オプション, 8-10
 READ WRITE オプション, 8-11
 RENAME DATA FILE オプション, 9-9
ALTER TABLE コマンド
 ADD PARTITION 句, 11-5
 ALLOCATE EXTENT オプション, 12-8
 DISABLE ALL TRIGGERS オプション, 17-12
 DISABLE 整合性制約オプション, 17-18
 DROP PARTITION 句, 11-5
 DROP 整合性制約オプション, 17-20
 ENABLE ALL TRIGGERS オプション, 17-11
 ENABLE 整合性制約オプション, 17-19
 MAXTRANS オプション, 10-8
 MODIFY PARTITION 句, 11-4
 SPLIT PARTITION 句, 11-5, 11-9
 TRUNCATE PARTITION 句, 11-7
 例, 12-8
ALTER TRIGGER コマンド
 DISABLE オプション, 17-12
 ENABLE オプション, 17-11
ALTER USER 権限, 20-14
ALTER USER コマンド
 パスワードの変更, 20-15
ALTER VIEW コマンド
 COMPILE オプション, 17-23
ANALYZE コマンド
 CASCADE オプション, 17-8
 COMPUTE STATISTICS オプション, 17-6
 ESTIMATE STATISTICS SAMPLE オプション, 17-7
 LIST CHAINED ROWS オプション, 17-8
 STATISTICS オプション, 17-4

 VALIDATE STRUCTURE オプション, 17-8
 共有 SQL, 17-7
 クラスタ・キーの数, A-14
ARCHIVE LOG コマンド
 LIST パラメータ, 5-8
ARCHIVELOG モード
 アーカイブ, 23-2
 使用可能にする, 23-4
 データ・ファイルをオフラインおよびオンラインにする, 9-8
 データベース作成時の設定, 23-4
AUDIT_TRAIL パラメータ
 設定, 22-13
AUDIT コマンド, 22-9
 システム権限, 22-10
 スキーマ・オブジェクト, 22-11
 文監査, 22-10

B

BACKGROUND_DUMP_DEST パラメータ, 4-14

C

CASCADE オプション
 一意キーまたは主キーの削除時, 17-18
 整合性制約, 15-10
CATAUDIT.SQL
 実行, 22-4
CATBLOCK.SQL スクリプト, 4-11
CATNOAUD.SQL
 実行, 22-5
CHAR データ型
 使用領域, 10-16
 列の長さの拡張, 12-7
CHECKPOINT_PROCESS パラメータ
 設定, 4-15
CHECK 制約, 17-17
CKPT, 4-15
CLEAR LOGFILE 句, 5-13
COMPUTE STATISTICS オプション, 17-6
CONNECT ロール, 21-12
CONTROL_FILES パラメータ
 既存の制御ファイルを上書きする, 2-9
 設定
 データベース作成前, 2-9, 6-4
 名前, 6-2

CREATE CLUSTER コマンド
 HASH IS オプション, 16-5
 HASHKEYS オプション, 16-6
 SIZE オプション, 16-5, A-13
 ハッシュ・クラスタ, 16-4
 例, 15-6

CREATE CONTROLFILE コマンド
 NORESETLOGS オプション, 6-6
 RESETLOGS オプション, 6-6
 説明, 6-5
 不整合の検出, 6-7

CREATE DATABASE コマンド
 CONTROLFILE REUSE オプション, 6-4
 MAXLOGFILES オプション, 5-4
 MAXLOGMEMBERS オプション, 5-4
 例, 2-7

CREATE INDEX コマンド
 ON CLUSTER オプション, 15-7
 回復不能 (UNRECOVERABLE), 14-4
 制約を指定, 14-6
 必要な一時領域, A-9
 明示的, 14-7

CREATE PROFILE コマンド
 COMPOSITE_LIMIT オプション, 20-19
 説明, 20-17

CREATE ROLE コマンド
 IDENTIFIED BY オプション, 21-13
 IDENTIFIED EXTERNALLY オプション, 21-14

CREATE ROLLBACK SEGMENT コマンド
 説明, 18-8
 チューニングのガイドライン, 2-14

CREATE SCHEMA コマンド
 必要な権限, 17-2
 複数の表とビュー, 17-2

CREATE SEQUENCE コマンド, 13-9

CREATE SYNONYM コマンド, 13-11

CREATE TABLESPACE コマンド
 データ・ファイル名, 8-4
 例, 8-4

CREATE TABLE コマンド
 CLUSTER オプション, 15-7
 PARTITION 句, 11-2
 回復不能 (UNRECOVERABLE), 12-4
 説明, 12-6

CREATE USER コマンド
 IDENTIFIED BY オプション, 20-11
 IDENTIFIED EXTERNALLY オプション, 20-11

CREATE VIEW コマンド
 OR REPLACE オプション, 13-8
 WITH CHECK OPTION, 13-2
 説明, 13-2

D

DATE データ型, 10-17

DB_BLOCK_BUFFERS パラメータ
 データベース作成前の設定, 2-11

DB_BLOCK_CHECKSUM, 9-12

DB_BLOCK_SIZE パラメータ
 作成前の設定, 2-10
 データベース・バッファのキャッシュ・サイズ, 2-11

DB_DOMAIN パラメータ
 データベース作成前の設定, 2-9

DB_NAME パラメータ
 MTS_SERVICE と~, 4-6
 データベース作成前の設定, 2-9

DBA, 1-2

DBA_DATA_FILES, 8-13, 9-13

DBA_EXTENTS, 9-12

DBA_FREE_SPACE, 8-13, 9-12

DBA_FREE_SPACE_COALESCED ビュー, 8-7

DBA_INDEXES ビュー
 データを入れる, 17-5

DBA_ROLLBACK_SEGS ビュー, 18-14, 18-15

DBA_SEGMENTS, 8-13, 9-12

DBA_TAB_COLUMNS ビュー
 データを入れる, 17-5

DBA_TABLESPACES, 8-13, 9-13

DBA_TABLESPACES ビュー, 8-12

DBA_TABLES ビュー
 データを入れる, 17-5

DBA_TS_QUOTAS, 8-13, 9-13

DBA_USERS, 8-13, 9-12

DBA ロール, 1-5, 21-12

DBMS_JOB パッケージ
 REMOVE プロシージャと~, 7-11
 ジョブ・キューと~, 7-4
 ジョブの強制的な実行, 7-14
 ジョブの変更, 7-11
 ジョブを送る, 7-5

DBMS_UTILITY.ANALYZE_SCHEMA()
 実行, 17-7

DROP CLUSTER コマンド

CASCADE CONSTRAINTS オプション, 15-10
INCLUDING TABLES オプション, 15-10

削除

ハッシュ・クラスタ, 16-8

表を含まないクラスタ, 15-10

DROP PARTITION 句

ALTER TABLE コマンド, 11-5

DROP PROFILE コマンド, 20-20

DROP ROLE コマンド, 21-15, 21-16

DROP ROLLBACK SEGMENT コマンド, 18-14

DROP SYNONYM コマンド, 13-11

DROP TABLESPACE コマンド, 8-12

DROP TABLE コマンド

CASCADE CONSTRAINTS オプション, 12-8

クラスタ化された表のための~, 15-10

説明, 12-8

DROP USER 権限, 20-16

DROP USER コマンド, 20-16

E

Enterprise, 1-16

Enterprise Manager

オペレーティング・システム・アカウント, 1-4

Enterprise Manager, 1-16

ESTIMATE STATISTICS オプション, 17-6

EXP_FULL_DATABASE ロール, 21-12

Export ユーティリティ

制限モードで使う, 3-4

F

FOREIGN KEY 制約

使用可能にする, 17-18

G

GRANT OPTION

取り消し, 21-19

説明, 21-17

GRANT コマンド

ADMIN オプション, 21-16

GRANT オプション, 21-17

SYSOPER/SYSDBA 権限, 1-12

オブジェクト権限, 21-17

システム権限とロール, 21-16

実施されるとき, 21-22

H

HOST

Server Manager のコマンド, 5-7

I

I/O

分散, 2-15

IMP_FULL_DATABASE ロール, 21-12

Import ユーティリティ

制限モードで使う, 3-4

INITIAL 記憶領域パラメータ, 10-7

変更, 12-7

INTRANS 記憶領域パラメータ

デフォルト, 10-8

トランザクション・エントリと~, 10-8

変更, 12-7

~を設定するためのガイドライン, 10-9

INSERT 権限

取消し, 21-19

付与, 21-18

INTERNAL

OSOPER と OSDBA, 1-7

インスタンスの起動, 3-2

セキュリティ, 19-7

停止のための接続, 3-9

データベースの作成, 2-6

~の代替, 1-7

INTERNAL 日付ファンクション

ジョブの実行と~, 7-8

J

JO ロック, 7-10

L

LGWR, 4-14

LICENSE_MAX_SESSIONS パラメータ

インスタンスの稼働中に変更, 20-4

設定, 20-4

データベース作成前の設定, 2-12

LICENSE_MAX_USERS パラメータ

設定, 20-5

データベース作成前の設定, 2-12

データベースの稼働中に変更, 20-5

LICENSE_SESSION_WARNING パラメータ
データベース作成前の設定, 2-12
LICENSE_SESSIONS_WARNING パラメータ
インスタンスの稼働中に変更, 20-4
設定, 20-4
LIST CHAINED ROWS オプション, 17-8
LOG_ARCHIVE_BUFFER_SIZE パラメータ
設定, 23-9
LOG_ARCHIVE_BUFFERS パラメータ
設定, 23-9
LOG_ARCHIVE_DEST パラメータ
設定, 23-12
LOG_ARCHIVE_FORMAT パラメータ
設定, 23-11
LOG_ARCHIVE_START パラメータ, 23-6
設定, 23-6, 23-7
LOG_CHECKPOINT_INTERVAL パラメータ
設定, 5-11
LOG_CHECKPOINT_TIMEOUT パラメータ
設定, 5-11
LOG_FILES パラメータ
ログ・ファイルの数と~, 5-4
LONG データ型, 10-17

M

MAX_DUMP_FILE_SIZE パラメータ, 4-14
MAX_ENABLED_ROLES パラメータ
デフォルト・ロール, 21-15
ロールを使用可能にする, 21-15
MAXDATAFILES パラメータ
変更, 6-5
MAXEXTENTS 記憶領域パラメータ
説明, 10-8
データ・ディクショナリに合わせて設定, 17-25
MAXINSTANCES パラメータ
変更, 6-5
MAXLOGFILES パラメータ
変更, 6-5
ログ・ファイルの数と~, 5-4
MAXLOGHISTORY
変更, 6-5
MAXLOGMEMBERS パラメータ
変更, 6-5
ログ・ファイルの数と~, 5-4
MAXTRANS 記憶領域パラメータ
デフォルト, 10-8

トランザクション・エントリと~, 10-8
変更, 12-7
~を設定するためのガイドライン, 10-9
MINEXTENTS 記憶領域パラメータ
説明, 10-8
変更, 12-7
MODIFY PARTITION 句
ALTER TABLE コマンド, 11-4
MONITOR コマンド
ROLLBACK オプション, 18-15
MOVE PARTITION 句
ALTER TABLE コマンド, 11-4
MTS_DISPATCHERS パラメータ
初期設定, 4-6
MTS_LISTENER_ADDRESS パラメータ
新しいディスパッチャの開始と~, 4-10
設定, 4-5
MTS_MAX_DISPATCHERS パラメータ, 4-8
設定, 4-8
MTS_MAX_SERVERS パラメータ
設定, 4-9
MTS_SERVERS パラメータ
最小値, 4-8
設定, 4-8
MTS_SERVICE パラメータ
設定, 4-6
デフォルトの DB_NAME パラメータ, 4-6

N

NEXT 記憶領域パラメータ, 10-7
データ・ディクショナリに合わせて設定, 17-25
NOARCHIVELOG モード
アーカイブ, 23-2
データ・ファイルをオフラインにする, 9-8
データベース作成時の設定, 23-4
NOAUDIT コマンド
監査オプションを使用禁止にする, 22-11
権限, 22-12
スキーマ・オブジェクト, 22-12
文, 22-12
NOT NULL 制約, 17-18
NUMBER データ型, 10-16

O

OPTIMAL 記憶領域パラメータ, 18-6

Oracle, 20-10
Oracle Parallel Server, 5-12
Oracle8 Server
 インストール, 1-17
 プロセス
 オペレーティング・システム名, 4-12
 監視, 4-10
 チェックポイント (CKPT), 4-15
 ディスパッチャのサービス名, 4-6
 トレース・ファイル fpr, 4-13
 ライセンス契約に従う, 20-2
 リリースの識別, 1-20
Oracle8 Server プロセス
 プロセス
 識別と管理, 4-10
 専用サーバー・プロセス, 4-2
Oracle ブロック, 2-10
ORAPWD ユーティリティ, 1-8
OS_ROLES パラメータ
 REMOTE_OS_ROLES, 21-25
 オペレーティング・システム認可, 21-14
 使用, 21-23
OS 認証, 1-7

P

Parallel Server
 ALTER CLUSTER..ALLOCATE EXTENT, 15-9
 LOG_CHECKPOINT_TIMEOUT と~, 5-12
 V\$THREAD ビュー, 5-14
 アーカイブ・ログ・ファイル名の形式, 23-11
 アーカイブのためのスレッドの指定, 23-8
 インスタンスのデータ・ファイル数の上限, 9-3
 順序番号と~, 13-10
 セッションと警告制限, 20-3
 独自のロールバック・セグメント, 18-2
 名前付きユーザー, 2-13
 名前付きユーザーでの制限, 20-5
 ライセンス・セッションの制限, 2-12
 ローカル・インスタンスに対してチェックポイント
 を強制する, 5-13
PARALLEL_MAX_SERVERS パラメータ, 4-16
PARALLEL_MIN_SERVERS パラメータ, 4-16
PARALLEL_SERVER_IDLE_TIME パラメータ, 4-16
PARTITION 句
 CREATE TABLE コマンド, 11-2
PCTFREE 記憶領域パラメータ

PCTUSED と~, 10-6
機能, 10-2
クラスタ化された表, 10-4
クラスタ化されていない表, 10-4
索引, 10-4
デフォルト, 10-3
ブロック・オーバーヘッドと~, 10-6
変更, 12-7
 ~を設定するためのガイドライン, 10-3
PCTINCREASE 記憶領域パラメータ
 説明, 10-8
 データ・ディクショナリに合わせて設定, 17-25
 変更, 10-10
PCTUSED 記憶領域パラメータ
 PCTFREE と~, 10-6
 機能, 10-4
 デフォルト, 10-5
 ブロック・オーバーヘッドと~, 10-6
 変更, 12-7
 ~を設定するためのガイドライン, 10-5
PL/SQL プログラム・ユニット
 置き換えられたビューと~, 13-8
 削除した表と~, 12-9
PRIMARY KEY 制約
 削除時の外部キー参照, 17-18
 作成時に使用可能にする, 14-6
 使用可能にする, 17-18
 使用禁止, 17-17
 対応する索引の削除, 14-9
 対応付けられる索引の記憶領域, 14-6
 ~と対応付けられる索引, 14-6
privileges
 revoking
 ADMIN OPTION, 21-19
 GRANT OPTION, 21-19
PROCESSES パラメータ
 データベース作成前の設定, 2-11
PUBLIC_DEFAULT プロファイル
 使用, 20-17
 プロファイルの削除, 20-20
PUBLIC ユーザー・グループ
 権限の付与と取消し, 21-21
 プロシージャ, 21-21

R

REBUILD PARTITION 句

- ALTER INDEX コマンド, 11-4, 11-14
- REDO ログ
 - アーカイブされた REDO ログ, 23-2
- REDO ログ・ファイル
 - アーカイブされた REDO ログ・ファイル, 23-4
 - オンライン, 5-2
 - 多重化
 - 概要, 5-2
 - 表示, 2-8
 - ログ順序番号
 - 定義, 5-2
- REFERENCES 権限
 - CASCADE CONSTRAINTS オプション, 21-20
 - 取消し, 21-19, 21-20
- REMOTE_LOGIN_PASSWORDFILE パラメータ, 1-10
- REMOTE_OS_AUTHENT パラメータ
 - 設定, 20-9
- REMOTE_OS_ROLES パラメータ
 - 設定, 21-14, 21-25
- RENAME コマンド, 17-2
- RESOURCE_LIMIT パラメータ
 - 制限を使用可能、使用禁止にする, 20-20
- RESOURCE ロール, 21-12
- RESTRICTED SESSION 権限
 - 制限モードで使う, 3-4
 - 制限モードのインスタンス, 3-7
 - セッション制限, 20-3
- REVOKE コマンド, 21-18
 - 実施されるとき, 21-22
- roles
 - revoking ADMIN OPTION, 21-19
- ROLLBACK_SEGMENTS パラメータ
 - データベース作成前の設定, 2-11
 - ロールバック・セグメントの追加, 18-8
- ROWID 疑似列, 10-17
- ROWID データ型, 10-17

S

- SCN, 9-13
- SEQUENCE_CACHE_ENTRIES パラメータ, 13-10
- Server Manager の起動, 2-6
- SET ROLE コマンド
 - オペレーティング・システム・ロールを使うとき, 21-25
 - パスワードの設定方法, 21-13
- SET TRANSACTION コマンド

- USE ROLLBACK SEGMENT オプション, 18-13
- SGA
 - キャッシュのバッファを決定する, 2-11
- SHUTDOWN コマンド, 3-8
 - ABORT オプション, 3-12
 - IMMEDIATE オプション, 3-11
 - NORMAL オプション, 3-11
- SNP バックグラウンド・プロセス
 - 説明, 7-2
- SORT_AREA_SIZE パラメータ
 - 索引作成と~, 14-3
- SPLIT PARTITION 句, 11-10
 - ALTER INDEX コマンド, 11-10
 - ALTER TABLE コマンド, 11-5, 11-9
- SQL*Loader
 - 索引と~, 14-3
 - 説明, 1-16
- SQL_TRACE パラメータ
 - トレース・ファイルと~, 4-13
- SQL トレース機能
 - 使用可能にするとき, 4-15
- SQL 文
 - 監査オプションを使用可能にする, 22-10
 - 監査オプションを使用禁止にする, 22-12
 - 必要な権限, 21-10
- STALE 状態
 - REDO ログ・メンバーの~, 5-9
- STARTUP コマンド, 3-2
 - FORCE オプション, 3-5
 - MOUNT オプション, 3-4
 - NOMOUNT オプション, 3-3
 - OPEN オプション, 3-4
 - RECOVER オプション, 3-5
 - RESTRICT オプション, 3-5
 - データベース名の指定, 3-2
 - パラメータ・ファイルの指定, 3-3
- SYS
 - 権限, 1-5
 - 初期パスワード, 1-4
 - 所有オブジェクト, 1-5
 - 保護の方針, 19-7
 - ユーザー, 1-5
- SYS.AUD\$
 - 監査証跡, 22-2
 - 作成と削除, 22-4
- SYSOPER/SYSDBA 権限
 - 権限所有者の判別, 1-12

接続, 1-13
付与と取消し, 1-12
ユーザーをパスワード・ファイルに追加する, 1-11

SYSTEM

初期パスワード, 1-4
所有オブジェクト, 1-5
保護の方針, 19-7
ユーザー, 1-5

SYSTEM 表領域

オフラインにすることの制限, 9-7
削除できない, 8-12
初期ロールバック・セグメント, 18-2
作られるとき, 8-3
非データ・ディクショナリ表と~, 12-3

SYSTEM ロールバック・セグメント

記憶領域パラメータの変更, 18-9
追加, 18-3
ロールバック・セグメント, 18-3

T

TRANSACTIONS_PER_ROLLBACK_SEGMENT パラメータ

使用, 18-2

TRANSACTIONS パラメータ

使用, 18-2

TRUNCATE PARTITION 句

ALTER TABLE コマンド, 11-7

TRUNCATE コマンド, 17-9

DROP STORAGE オプション, 17-10

REUSE STORAGE オプション, 17-10

Trusted Oracle7 Server

データベース・アクセスの制御, 20-1
表領域とデータ・ファイルの管理, 9-2
ユーザーとリソースの管理, 20-1

U

UNIQUE キー制約

削除時の外部キー参照, 17-18
作成時に使用可能にする, 14-6
使用可能にする, 17-18
使用禁止, 17-17
対応する索引の削除, 14-9
対応付けられる索引の記憶領域, 14-6
~と対応付けられる索引, 14-6

UNLIMITED TABLESPACE 権限, 20-13

UPDATE 権限

取消し, 21-19

USER_DUMP_DEST パラメータ, 4-14

USER_EXTENTS, 9-12

USER_FREE, 8-13, 9-12

USER_INDEXES ビュー

データを入れる, 17-5

USER_SEGMENTS, 8-13, 9-12

USER_TAB_COLUMNS ビュー

データを入れる, 17-5

USER_TABLESPACES, 8-13, 9-13

USER_TABLES ビュー

データを入れる, 17-5

UTLCHAIN.SQL, 17-8

UTLLOCKT.SQL スクリプト, 4-11

V

V\$ARCHIVE ビュー, 23-10

V\$DATABASE ビュー, 23-10

V\$DATAFILE, 8-13, 9-13

V\$DBFILE ビュー, 2-8

V\$DISPATCHER ビュー

ディスパッチャ・プロセス・ロードの制御, 4-9

V\$LICENSE ビュー, 20-6

V\$LOGFILE ビュー, 2-8

V\$LOG ビュー

アーカイブ状態の表示, 23-10

オンライン REDO ログと~, 5-14

V\$PWFILE_USERS ビュー, 1-12

V\$QUEUE ビュー

ディスパッチャ・プロセス・ロードの制御, 4-9

V\$ROLLNAME

PENDING OFFLINE セグメントの検索, 18-16

V\$ROLLSTAT

PENDING OFFLINE セグメントの検索, 18-16

V\$SESSION, 7-15

V\$SESSION ビュー, 4-20

V\$THREAD ビュー, 5-14

VALIDATE STRUCTURE オプション, 17-8

VARCHAR2 データ型, 10-16

使用領域, 10-16

W

WORM デバイス

～と読み専用表領域, 8-11

あ

アーカイブ

アーカイブ先の指定, 23-11

権限

手動アーカイブ, 23-8

使用可能にする, 23-6

使用禁止, 23-7

システム・パフォーマンスへの影響を最小にする, 23-9

手動, 23-8

使用可能および使用禁止にする, 23-4

自動

起動時に使用禁止にする, 23-7

使用可能にする, 23-6

使用禁止, 23-7

設定後に使用禁止にする, 23-7

自動アーカイブを使用可能にする, 23-6

情報の表示, 23-10

スピードを上げる, 23-9

短所, 23-2

長所, 23-2

調整, 23-8

モードの変更, 23-4

アーカイブされた REDO ログ

アーカイブ・モード, 23-4

アーカイブ先の指定, 23-11

アーカイブ先の設定, 23-12

自動アーカイブ, 23-6

自動アーカイブを使用可能にする, 23-6

自動アーカイブを使用禁止にする, 23-7

状態の表示, 23-10

ファイル名の形式, 23-11

アカウント

オペレーティング・システム

データベース管理者, 1-4

ロール識別機能, 21-23

ユーザー

SYS と SYSTEM, 1-4

空き領域

合わせる, 8-6

使用可能エクステンツのリスト表示, 17-31

表領域と～, 8-14

アクセス

オブジェクト

権限のタイプ, 21-9

権限の取消し, 21-19

権限の付与, 21-17

データ

システム権限, 21-2

データベース

権限の取消し, 21-18

権限の付与, 21-16

制限, 3-4

データベース管理者のアカウント, 1-4

アプリケーション

メンテナンス操作時に休止する, 11-15

アプリケーション開発

セキュリティ, 19-9

アプリケーション管理者, 1-3

データベース管理者, 19-10

アプリケーションの開発者

権限, 19-8

ロール, 19-9

暗号化

Oracle パスワード, 20-7

い

移行

データベースの移行, 2-3

依存性

表示, 17-30

位置

ロールバック・セグメント, 18-7

一時セグメント

索引作成と～, 14-3

一時領域

割り当てる, 12-6

位置の移動

制御ファイル, 6-4

データ・ファイル, 9-8, 9-10

移動

位置の移動, 9-8

索引パーティション, 11-4

制御ファイル, 6-4

表のパーティション, 11-4

インスタンス

起動, 3-2

即時停止, 3-11

中断, 3-12

データベース作成の前に起動する, 2-6

インスタンス識別子

プロセス名と～, 4-13

インスタンスの起動

INTERNAL として接続, 3-2

一般手順, 3-2

回復, 3-5

強制的, 3-5

システム起動時に自動起動, 3-6

自動アーカイブを使用可能にする, 23-6

制限モード, 3-4

通常の方法, 3-4

ディスパッチャ・プロセスと～, 4-6

データベースのクローズとマウント, 3-3

データベースの作成, 3-3

データベースのマウントとオープン, 3-4

データベース名の指定, 3-2

データベース名の対立, 2-9

データベースをマウントしない, 3-3

排他モード, 3-5

発生する問題, 3-5

パラメータ・ファイル, 3-3

パラレル・モード, 3-5

マルチスレッド・サーバー, 3-2

マルチスレッド・サーバーを使う, 4-5

リモート・インスタンスの起動, 3-6

例, 3-5

インスタンスの構成

専用サーバー・プロセス, 4-2

インスタンスの停止

INTERNAL として接続, 3-9

一般手順, 3-8

インスタンスの中断, 3-12

接続と～, 3-8

即時, 3-11

通常の方法, 3-10

例, 3-11

インストール

Oracle8 Server, 1-17

チューニングに関する推奨事項, 2-14

データベースの作成, 2-3

インダウト・トランザクション

ロールバック・セグメントと～, 18-11

インポート

ジョブ, 7-7

え

英数字データ型, 10-16

エクステンツ

削除した表と～, 12-9

使用可能エクステンツの表示, 17-31

情報の表示, 17-31

データ・ディクショナリ・ビュー, 17-28

割り当てる

クラスタ, 15-9

索引作成, 14-5

表, 12-8

エラー

ALERT ファイルと～, 4-13

ORA-00028, 4-19

ORA-00114, 4-6

ORA-01090, 3-8

ORA-01173, 6-8

ORA-01176, 6-8

ORA-01177, 6-8

ORA-1215, 6-8

ORA-1216, 6-8

ORA-1547, 17-27

ORA-1628 ~ 1630, 17-27

インスタンス起動時, 3-5

制御ファイル作成時, 6-8

データベース作成時, 2-8

トレース・ファイルと～, 4-13

古すぎるスナップショット, 18-6

お

置換え

ビュー, 13-8

オフライン・データ・ファイル, 9-8

オフライン・ロールバック・セグメント

オンラインにする, 18-11

説明, 18-10

使うとき, 18-10

オフラインにする

表領域, 8-8

オフライン表領域

変更, 8-7

優先順位, 8-8

ロールバック・セグメントと～, 18-10

オブジェクト、スキーマ

権限, 21-9

- 権限の取消し, 21-19
- 権限の付与, 21-17
- 削除されたユーザーが所有する, 20-16
- デフォルト表領域, 20-12
- 取り消された表領域, 20-13
- 取消しでの連鎖的影響, 21-20
- オペレーティング・システム
 - Oracle8 プロセス名, 4-12
 - アカウント, 21-23
 - オープン・ファイル数の制限, 9-2
 - セキュリティ, 19-3
 - データ・ファイルの削除, 8-12
 - データベース管理者の要件, 1-4
 - 認証, 21-22
 - ファイルの改名と再配置, 9-9
 - ロール, 21-22
 - ロール識別機能, 21-23
 - ロールを使用可能、使用禁止にする, 21-25
 - ~を使う監査, 22-2
- オンライン REDO ログ, 5-2
 - STALE メンバー, 5-9
 - 位置, 5-3
 - グループの削除, 5-8
 - グループを作る, 5-5
 - 権限
 - グループの削除, 5-8
 - グループの追加, 5-5
 - メンバーの削除, 5-9
 - ログ・スイッチの強制, 5-12
 - 構成のガイドライン, 5-2
 - 情報の表示, 5-14
 - 多重化, 5-2
 - データ・ファイルと分離した格納, 9-4
 - データベースのオープン中には使えない, 3-3
 - ファイルの移動, 5-8
 - ファイルの改名, 5-8
 - ファイルの数, 5-4
 - メンバーの改名, 5-6
 - メンバーの削除, 5-9
 - メンバーを作る, 5-6
 - ログ・スイッチの強制, 5-12
- オンライン REDO ログの多重化, 5-2
 - シンメトリック・グループ, 5-2
- オンライン・データ・ファイル, 9-8
- オンライン・ロールバック・セグメント
 - オフラインにする, 18-12
 - 新規作成時, 18-8

- 説明, 18-10
- ロールバック・セグメントをオンラインにする, 18-11
- オンラインにする
 - 表領域, 8-7
- オンライン表領域
 - 変更, 8-7

か

- ガイドライン
 - ロールバック・セグメントの管理, 18-2
- 開発者、アプリケーション, 19-8
- 回復
 - アーカイブの影響, 23-2
 - 新しい制御ファイルの作成, 6-5
 - 自動回復を実行する起動, 3-5
- 回復不能
 - 表, 12-4
- 回復不能索引
 - 索引, 14-4
- 改名
 - オンライン REDO ログ・メンバー, 5-6
 - スキーマ・オブジェクト, 17-2
 - 制御ファイル, 6-4
 - 単一の表領域のデータ・ファイル, 9-9
 - データ・ファイル, 9-8, 9-10
- 書込み可能表領域, 8-11
- 監査, 22-2
 - AUDIT コマンド, 22-9
 - 疑わしいアクティビティ, 22-3
 - オブジェクト監査に必要な権限, 22-11
 - オブジェクト~のためのショートカット, 22-9
 - オプションを使用可能にする, 22-9, 22-13
 - オプションを使用可能にすることとの関係, 22-10
 - オプションを使用禁止にする, 22-11, 22-12, 22-13
 - オプションを使用禁止にすることとの関係, 22-12
 - オペレーティング・システム監査証跡, 22-6
 - 監査オプションのレベル, 22-7
 - 監査証跡の管理, 22-4
 - 監査証跡レコード, 22-5
 - ガイドライン, 22-2
 - 権限監査オプション, 22-8
 - システム監査に必要な権限, 22-11
 - システム権限, 22-10
 - システムのショートカット, 22-7
 - 情報を監理しやすい状態に維持する, 22-2

- スキーマ・オブジェクト, 22-11
- スキーマ・オブジェクト型, 22-8
- セッション・レベル, 22-8
- データベースを使う, 22-2
- デフォルトのオプション, 22-11
- デフォルトのオプションを使用禁止にする, 22-13
- トリガーと~, 22-20
- 表示
 - アクティブなオブジェクト・オプション, 22-18
 - アクティブな権限オプション, 22-18
 - アクティブな文オプション, 22-18
 - デフォルトのオブジェクト・オプション, 22-19
- ビュー, 22-4
- 文, 22-10
- 文レベル, 22-7
- 方針, 19-17
- 履歴情報, 22-3
- 監査証跡, 22-14
 - アーカイブ, 22-15
 - 解釈, 22-16
 - サイズの削減, 22-15
 - サイズの制御, 22-14
 - 最大サイズ, 22-14
 - 作成と削除, 22-4
 - 整合性の保護, 22-16
 - ~内のレコード, 22-7
 - ビュー, 22-4
 - ビューの削除, 22-5
 - 変更の監査, 22-16
 - 変更の記録, 22-16
 - レコードの消去, 22-15
 - ~を保持する表, 22-2
- 監視
 - インスタンスのプロセス, 4-10
 - データ・ファイル, 9-12
 - パフォーマンス表, 4-12
 - 表領域, 9-12
 - ロールバック・セグメント, 18-6, 18-15
 - ロック, 4-11
- 管理
 - オブジェクト依存性, 17-22
 - 監査, 22-1
 - クラスタ, 15-1
 - クラスタ化された表, 15-1
 - クラスタ索引, 15-1

- 索引, 14-1, 14-9
- シノニム, 13-11
- 順序, 13-9
- ジョブ, 7-4
- 表, 12-1
- ビュー, 13-1, 13-9
- プロファイル, 20-17
- ユーザー, 20-10
- ロール, 21-11
- ロールバック・セグメント, 18-1

き

キー

- クラスタ, 15-2

キー保存表

- 結合ビュー, 13-5

記憶領域

- 表領域の取消し, 20-13
- 表領域の変更, 8-5
- 無制限の割当て制限, 20-13
- 割当て制限, 20-13

記憶領域パラメータ

- INITIAL, 10-7, 12-7
- INITRANS, 10-9, 12-7
- MAXEXTENTS, 10-8
- MAXTRANS, 10-8, 12-7
- MINEXTENTS, 10-8, 12-7
- NEXT, 10-7
- OPTIMAL (ロールバック・セグメントでの), 18-6
- PCTFREE, 12-7
- PCTINCREASE, 10-8
- PCTUSED, 12-7
- SYSTEM ロールバック・セグメント, 18-9
- 一時セグメント, 10-11
- 設定値の変更, 10-10
- 適用できるオブジェクト, 10-7
- データ・ディクショナリ, 17-25
- デフォルト, 10-7
- 優先順位, 10-11
- ロールバック・セグメント, 18-8
- 記憶領域パラメータの優先順位, 10-11
- 疑似列, 10-17
- 機密性
 - セキュリティ, 19-3
- キャラクタ・セット

- Oracle によるサポート, 10-16
- データベース作成時の指定, 2-2
- パラメータ・ファイルと~, 3-13
- マルチバイト・キャラクタ
 - ユーザー・パスワード, 20-11
 - ユーザー名, 20-11
 - ロール・パスワード, 21-13
 - ロール名, 21-11
- キューに送られたジョブの所有者, 7-7
- 行
 - 整合性制約違反, 17-14
 - ブロック間での連鎖, 10-4, 17-8
- 共有 SQL 領域
 - ANALYZE コマンド, 17-7
- 共有サーバー・プロセス
 - 数を変更する権限, 4-9
 - 起動時に開始する数, 4-8
 - 最小数の変更, 4-9
 - 最大数, 4-9
 - トレース・ファイル, 4-13
- 共有プール
 - ANALYZE コマンド, 17-7
 - マルチスレッド・サーバー, 4-5
- 共有モード
 - ロールバック・セグメントと~, 18-3
- 切捨て
 - クラスタ, 17-9
 - 権限, 17-10
 - パーティション・オブジェクト, 11-7
 - 表, 17-9

く

- クラスタ
 - 位置, 15-5
 - エクステントの割当て, 15-9
 - 管理, 15-1
 - 管理のガイドライン, 15-4
 - 概要, 15-2
 - キー, 15-2
 - 記憶領域パラメータ, 10-10
 - 切捨て, 17-9
 - クラスタ・キーの列, 15-4
 - 権限
 - 削除するための~, 15-10
 - 制御, 21-9
 - 作するための~, 15-6

- 構造の妥当性検査, 17-8
- 索引
 - ハッシュと対比, 16-2
 - 索引作成, 15-7
 - 索引と~, 14-2
 - 削除, 15-9
 - 削除した表と~, 12-9
 - 作成, 15-6
 - データの選択, 15-4
 - 統計分析, 17-3
 - ~のための PCTFREE の指定, 10-4
- ハッシュ
 - 索引と対比, 16-2
 - ハッシュ・クラスタ, 16-1
 - 変更, 15-8
 - 領域の見積もり, 15-5, 15-6, A-10
- クラスタ・キー
 - ANALYZE コマンド, A-14
 - SIZE パラメータ, 15-5
 - ~の列, 15-4
- クラスタ化された表, 15-10
- クラスタ化されてない表
 - サイズの見積もり, A-2
- グローバルな索引
 - パーティションを削除する, 11-5, 11-8
 - パーティションを分割する, 11-10
- グローバルなデータベース名, 2-9
- グローバルなユーザー, 20-10

け

- 計画
 - データベース, 1-18
 - データベース作成, 2-2
 - リレーショナル設計, 1-18
- 警告
 - CONTROL_FILES パラメータの設定, 2-9
 - 監査オプションを使用禁止にする, 22-12
 - 監査を使用可能にする, 22-10
 - データ・ディクショナリの記憶領域パラメータの変更, 17-25
 - ミラー化された制御ファイルの使用, 6-2
 - ロールバック・セグメントの作成, 2-11
- 結合ビュー, 13-4
 - DELETE 文, 13-6
 - キー保存表, 13-5
 - 変更

- 規則, 13-5
- 変更可能なとき, 13-4
- マージ可能, 13-5
- 権限, 21-2, 21-10
 - CREATE SCHEMA コマンド, 17-2
 - REDO ログ・グループの追加, 5-5
 - RESTRICTED SESSION システム権限, 3-4, 3-7
 - アプリケーションの開発者, 19-8
 - オブジェクト, 21-9
 - オブジェクト権限の取消し, 21-19
 - オブジェクトの監査, 22-11
 - オブジェクト~の取消し, 21-19
 - オペレーティング・システム
 - データベース管理者に必要な権限, 1-4
- 改名
 - REDO ログ・メンバー, 5-6
 - オブジェクト, 17-2
 - 表領域のデータ・ファイル, 9-9
 - 複数の表領域のデータ・ファイル, 9-10
- 管理の方針, 19-5
- 許可される SQL 文, 21-10
- 切捨て, 17-10
- クラスタ作成, 15-6
- 権限付与のリスト, 21-27
- 個々の権限名, 21-2
- 削除
 - REDO ログ・グループ, 5-8
 - オンライン REDO ログ・メンバー, 5-9
 - クラスタ, 15-10
 - 索引, 14-9
 - シノニム, 13-11
 - 順序, 13-10
 - 表, 12-8
 - ビュー, 13-9
 - ロール, 21-15
 - ロールバック・セグメント, 18-14
- 作成
 - シノニム, 13-11
 - 順序, 13-9
 - 表, 12-6
 - 表領域, 8-4
 - ビュー, 13-2
 - ユーザー, 20-10
 - ロール, 21-11
 - ロールバック・セグメント, 18-8
- システム, 21-2
- システムの監査, 22-11

- 手動アーカイブ, 23-8
- 使用の監査, 22-8
- 自動アーカイブを使用可能にする, 23-6
- 自動アーカイブを使用禁止にする, 23-7
- ジョブ・キューと~, 7-4
- セッション制限の変更, 20-4
- 選択した列, 21-19
- チェックポイントの強制, 5-12
- データ・ファイルをオフラインおよびオンラインにする, 9-7
- データベース管理者, 1-4
- トリガーを使用可能および使用禁止にする, 17-11
- 取消し, 21-19
 - オブジェクト権限, 21-20
 - システム権限, 21-18
- パッケージの再コンパイル, 17-24
- 表領域にデータ・ファイルを追加する, 9-5
- 表領域を合わせる, 8-6
- 表領域をオフラインにする, 8-8
- 表領域をオンラインにする, 8-7
- ビューの置換え, 13-8
- ビューの再コンパイル, 17-23
- 付与
 - オブジェクト権限, 21-17
 - システム権限, 21-16
 - 説明, 21-16
 - 必要な権限, 21-17
- 分析する、オブジェクトを, 17-3
- プロシージャの再コンパイル, 17-23
- プロファイルの削除, 20-20
- 変更
 - 索引, 14-8
 - 順序, 13-9
 - ディスパッチャ権限, 4-10
 - デフォルトの記憶領域パラメータ, 8-5
 - 名前付きユーザーの制限, 20-6
 - パスワード, 20-15
 - 表, 12-7
 - ユーザー, 20-14
 - ロール認証, 21-13
 - ロールバック・セグメント, 18-10
- リソース・コストの設定, 20-19
- リソース制限を使用可能、使用禁止にする, 20-20
- 列, 21-18
- 連鎖的な取消し, 21-20
- ロールで分類, 21-11
- ログ・スイッチの強制, 5-12

権限とロールの付与

- SYSOPER/SYSDBA 権限, 1-12
- オブジェクト権限のショートカット, 21-10
- 権限付与のリスト, 21-25

こ

高水位標

- セッション, 20-3
- 高速チェックポイント, 5-12
- 後続 NULL, A-10
- コスト
 - リソース制限, 20-19

さ

サーバー

- 専用
 - マルチスレッドと対比, 4-3
- マルチスレッド
 - 専用と対比, 4-3

サーバー単位

- 複合制限, 20-19

サービス名

- マルチスレッド・サーバーのディスパッチャの~, 4-6

再コンパイル

- 自動的, 17-23
- パッケージ, 17-24
- ビュー, 17-23
- ファンクション, 17-23
- プロシージャ, 17-23

サイズ

- クラスタ, A-10
- クラスタ化されてない表, A-2
- データ・ファイル, 9-4
- ハッシュ・クラスタ, 16-3
- ロールバック・セグメント, 18-4

サイズの見積もり

- ハッシュ・クラスタ, 16-3
- 表, 12-5, A-5

索引

- INITRANS, 14-3
- MAXTRANS, 14-3
- PCTFREE, 14-3
- PCTUSED, 14-3
- SQL*Loader と~, 14-3

- 一時セグメントと~, 14-3

- 一時領域と~, A-9

- エクステント割当て, 14-5

- 管理, 14-1, 14-9

- 管理のガイドライン, 14-2

- 概要, 14-2

- 記憶領域パラメータ, 10-10

- 記憶領域パラメータの設定, 14-5

クラスタ

- 管理, 15-1
- 削除, 15-9
- 作成, 15-6
- 変更, 15-9

権限

- 削除するための~, 14-9

- 制御, 21-9

- 変更のための~, 14-8

- 構造の妥当性検査, 17-8

- サイズの見積もり, 14-5

- 索引作成の並行化, 14-4

- 削除, 14-9

- 削除した表と~, 12-9

作成

- 回復不能, 14-4

- 表データ挿入後, 14-2

- 明示的, 14-7

- 制約の使用禁止または削除と~, 14-6

- 正しい表および列, 14-6

- 統計分析, 17-3

- パーティションを追加する, 11-5

- 表あたりの制限, 14-3

- 表との分離, 12-5

- 表領域, 14-4

- 変更, 14-8

- 領域使用の監視, 14-8

索引パーティション

- 移動, 11-4

- 再構築する, 11-14

- 削除, 11-7

- 分割する, 11-10

- マージする, 11-11

索引構成表, 12-9

索引のパーティション化

- パーティションを再構築する, 11-14

- マージする, 11-11

削除

- オンライン REDO ログ・グループ, 5-8

- オンライン REDO ログ・メンバー, 5-9
- 監査証跡, 22-4
- クラスタ, 15-9
- クラスタ索引, 15-9
- 索引, 14-9
- 索引パーティション, 11-7
- シノニム, 13-11
- 順序, 13-10
- 制御ファイル, 6-8
- 整合性制約
 - 索引への影響, 14-6
 - 説明, 17-20
- データ・ファイル, 8-12
- データベース, 2-8
- ハッシュ・クラスタ, 16-8
- 表, 12-8
- 表の統計, 17-4
- 表のパーティション, 11-5
- 表領域
 - 説明, 8-12
 - 必要な権限, 8-12
- ビュー, 13-9
- プロファイル, 20-20
- ユーザー, 20-16
- ロール, 21-15
- ロールバック・セグメント, 18-11, 18-14

作成

- REDO ログ・メンバー, 5-6
- オンライン REDO ログ・グループ, 5-5
- 監査証跡, 22-4
- クラスタ, 15-6
- クラスタ化された表, 15-6
- クラスタ索引, 15-6
- 索引
 - 明示的, 14-7
- シノニム, 13-11
- 順序, 13-9
- 制御ファイル, 6-3
- データ・ファイル, 8-3, 9-4
- データベース, 1-19
 - CREATE DATABASE の実行, 2-6
 - アーカイブ・モード, 23-4
 - 新しいデータベースのバックアップ, 2-7
 - インストール中, 2-3
 - 異なるバージョンからの移行, 2-3
 - 準備, 2-2
 - 前提条件, 2-3

- 発生する問題, 2-8
- ハッシュ・クラスタ, 16-4
- ハッシュ・クラスタ表, 16-4
- パーティション・オブジェクト, 11-2
- パーティション表, 11-2
- パラメータ・ファイル, 2-4
- 表, 12-6
- 表領域, 8-3
 - 必要なロールバック・セグメント, 8-4
- ビュー, 13-2
- 複数のオブジェクト, 17-2
- プロファイル, 20-17
- ロールバック・セグメント
 - 記憶領域パラメータの指定, 18-8
 - 説明, 18-8

参照整合性制約

- 表のパーティションを切捨てる, 11-8
- 表のパーティションを削除する, 11-6

し

時間枠

- 履歴表内で移動する, 11-14

識別

- ユーザー, 20-7

システム・グローバル領域, 2-11

システム・グローバル領域 (SGA), 2-11

システム権限, 21-2

システム変更番号 (SCN)

- データ・ファイルを調べる, 9-13

事前定義済みのロール, 1-5

自動アーカイブ

- アーカイブ・ログのアーカイブ先, 23-6

シノニム

- 依存性の表示, 17-30
- 管理, 13-11
- 削除, 13-11
- 削除した表と~, 12-9
- 削除するための権限, 13-11
- 作成, 13-11
- 作するための権限, 13-11
- パブリック, 13-11
- プライベート, 13-11

手動アーカイブ

- ARCHIVELOG モード, 23-8

順序

- Parallel Server と~, 13-10

- 管理, 13-9
- 削除, 13-10
- 削除するための権限, 13-10
- 作成, 13-9
- 初期化パラメータ, 13-10
- 作するための権限, 13-9
- 変更, 13-10
- 変更するための権限, 13-9
- 使用可能にする
 - アーカイブ, 23-4
 - 監査オプション
 - 権限, 22-13
 - 説明, 22-9
 - 整合性制約
 - 違反が存在するとき, 17-14
 - 作成時, 17-17
 - 例, 17-18
 - 例外のレポート, 17-20
 - トリガー, 17-11
 - リソース制限, 20-20
- 使用禁止
 - アーカイブ, 23-4
 - 監査, 22-13
 - 監査オプション, 22-11, 22-12
 - 自動アーカイブ, 23-7
 - 整合性制約, 17-17
 - 索引への影響, 14-6
 - トリガー, 17-11
 - リソース制限, 20-20
- ショートカット
 - CONNECT、監査のため, 22-8
 - オブジェクト監査, 22-9
 - オブジェクト権限, 21-10
 - 文レベル監査オプション, 22-7
- 初期
 - SYS と SYSTEM のパスワード, 1-4
- 初期化パラメータ
 - 順序に影響する, 13-10
 - マルチスレッド・サーバー, 4-5
- ジョブ
 - INTERNAL 日付ファンクションと~, 7-8
 - インポート, 7-7
 - エクスポート, 7-7
 - 管理, 7-4
 - 強制的な実行, 7-13
 - 所有者, 7-7
 - 実行, 7-9

- ジョブ・キューからの削除, 7-11
- ジョブ定義, 7-8
- ジョブ番号, 7-8
- ジョブ・キューに送る, 7-5
- スケジューリング, 7-4
- 中止する, 7-15
- 中断された~, 7-12
- 中断されたジョブの実行, 7-13
- 中断されたジョブをマークする, 7-13
- データベース・リンクと~, 7-9
- トレース・ファイル, 7-10
- 変更, 7-11
- 問題解決, 7-10
- ジョブ・キュー, 7-2, 7-3
- ジョブの削除, 7-11
- ジョブのスケジューリング, 7-4
- 使う権限, 7-4
- ~内でのジョブの実行, 7-9
- 表示, 7-15
- ロック, 7-10
- ジョブのエクスポート, 7-7
- ジョブの環境, 7-7

す

- スキーマ・オブジェクト
 - アクセスするための権限, 21-9
 - オブジェクト間の依存性, 17-22
 - 改名, 17-2, 17-3
 - 改名する権限, 17-2
 - 監査, 22-8
 - 監査オプションを使用可能にする, 22-11
 - 監査オプションを使用禁止にする, 22-12
 - 情報のリスト表示, 17-28
 - タイプ別のリスト, 17-29
 - デフォルトの監査オプション, 22-11
 - 複数のオブジェクトを作る, 17-2
- ストアド・プロシージャ
 - PUBLIC に付与された権限を使う, 21-21
 - 再コンパイルのための権限, 17-23
- ストリーム
 - テープ・ドライブ, 23-9
- スナップショット
 - 記憶領域パラメータ, 10-10
 - 古すぎる
 - OPTIMAL 記憶領域パラメータと~, 18-6
- スナップショット・ログ

記憶領域パラメータ, 10-10

せ

制御ファイル

- 1 つは必要, 6-3
- 位置, 6-3
- 位置の移動, 6-4
- 移動, 6-4
- 改名, 6-4
- 数, 6-2
- ガイドライン, 6-2
- 既存のファイルの上書き, 2-9
- 起動時には使えない, 3-3
- サイズ, 6-3
- サイズの変更, 6-4
- 削除, 6-8
- 作成
 - 新しいファイル, 6-5
 - 初期, 6-4
 - 説明, 6-3
 - 追加の制御ファイル, 6-4
- 作成中のエラー, 6-8
- 追加, 6-4
- データ・ディクショナリとの不一致, 6-7
- データベース作成前に名前を指定する, 2-9
- デフォルト名, 2-9, 6-4
- 名前, 6-2
- ミラー化, 2-10
- ミラー化の重要性, 6-2

制限

- セッション、高水位標, 20-3
- 同時使用, 20-2
- 複合制限, 20-19
- リソース制限, 20-19

制限、データベースへのアクセス

- インスタンスの起動, 3-4

整合性制約

- 違反, 17-14
- 違反が存在する場合に使用可能にする, 17-14
- 管理, 17-14
- 削除, 17-20
- 削除および使用禁止, 14-6
- 作成時に使用可能にする, 17-17
- 作成時に使用禁止にする, 17-17
- 使用可能にする, 17-13
- 使用禁止, 17-13, 17-17

使用禁止にするととき, 17-14

表領域の削除と~, 8-12

例外, 17-20

整合性制約違反, 17-14

責任

- データベース・ユーザー, 1-3
- データベース管理者, 1-2

セキュリティ

- REMOTE_OS_ROLES パラメータ, 21-25
- アプリケーションの開発者, 19-8
- 一般のユーザー, 19-4
- オペレーティング・システムのセキュリティとデータベース, 19-3
- 監査証跡の保護, 22-16
- 監査方針, 19-17
- 管理者, 19-2
- 権限, 19-2
- 権限管理の方針, 19-5
- セキュリティ管理者, 1-3
- セキュリティを強制するロール, 19-5
- データ, 19-3
- データベース・セキュリティ, 19-2
- データベース・ユーザー, 19-2
- データベース管理者の責任, 1-4
- データベース管理者の方針, 19-7
- データベースへのアクセス, 19-2
- 機密性, 19-3
- 方針の設定, 19-1
- マルチバイト・キャラクタ
 - ユーザー・パスワード, 20-11
 - ユーザー名, 20-11
 - ロール・パスワード, 21-13
 - ロール名, 21-11
 - ユーザーの認証, 19-2

セグメント

- 一時記憶領域パラメータ, 10-11
- 監視, 18-15
- 情報の表示, 17-31
- データ・ディクショナリ, 17-25
- データと索引
 - デフォルトの記憶領域パラメータ, 10-9
- ロールバック, 18-1

セッション

- Parallel Server のセッション制限, 2-12
- インスタンスごとの制限, 20-2
- インスタンスでの最大数の設定, 20-4
- インスタンスの警告制限の設定, 20-4

- 権限定義域のリスト, 21-28
- 現行数と高水位標の参照, 20-6
- 接続と切断の監査, 22-8
- 同時実行セッションの数, 2-12
- メモリー使用の表示, 20-25
- セッション、ユーザー
 - アクティブ, 4-19
 - アクティブでない, 4-20
 - 停止, 4-18
 - 停止したセッションのビュー, 4-20
 - 停止のマークを付ける, 4-19
- セッション・モニター, 4-11
- セッションの制限、ライセンス
 - 初期設定, 2-12
- セッションの停止
 - アクティブでないセッション, 4-20
 - アクティブでないセッションの例, 4-20
 - アクティブなセッション, 4-19
 - セッションの識別, 4-19
- 接続
 - INTERNAL としてデータベースとの接続, 3-2
 - 監査, 22-8
 - 管理者権限, 3-9
 - 専用サーバー, 4-3
 - 停止処理中の, 3-8
- 切断
 - 監査, 22-8
- 前提条件
 - データベースの作成, 2-3
- 専用サーバー
 - マルチスレッド・サーバーと対比, 4-3
- 専用サーバー・プロセス
 - 構成, 4-2
 - 接続, 4-3
 - トレース・ファイル, 4-13

そ

- ソフトウェア・バージョン, 1-20

た

- 多重化
 - REDO ログ・ファイル, 5-2
 - オンライン REDO ログ, 5-2

ち

- チェックサム
 - REDO ブロックの~, 5-13
 - データ・ブロック, 9-12
- チェックポイント
 - 間隔の設定, 5-10
 - 強制的, 5-12
 - 高速チェックポイント, 5-12
 - 制御, 5-10
 - ログ・スイッチと~, 5-10
- チェックポイント・プロセス (CKPT)
 - 起動, 4-15
- 中止する
 - ジョブ, 7-15
- 中断
 - インスタンスの停止, 3-12
- 中断されたジョブ
 - 実行, 7-13
 - 説明, 7-12
 - マークする, 7-13
- 調整
 - アーカイブ, 23-8
 - 初期, 2-14
 - データベース, 1-20

て

- 停止
 - ユーザー・セッション, 4-18
- ディスパッチャ・プロセス
 - 開始する数, 4-6
 - 数の設定, 4-9
 - 数を変更する権限, 4-10
 - サービス名, 4-6
 - 最大数の計算, 4-8
 - 削除, 4-9
 - 新規に作る, 4-9
- データ
 - セキュリティ, 19-3
- データ・ディクショナリ
 - V\$DBFILE ビュー, 2-8
 - V\$DISPATCHER ビュー, 4-9
 - V\$LOGFILE ビュー, 2-8
 - V\$QUEUE ビュー, 4-9
 - 記憶領域パラメータの設定, 17-25
 - 記憶領域パラメータの変更, 17-27

- 削除した表と～, 12-9
- スキーマ・オブジェクト・ビュー, 17-28
- 制御ファイルとの不一致, 6-7
- セグメント, 17-25
- データ・ファイル
 - MISSING, 6-8
 - REDO ログ・ファイルと分離した格納, 9-4
 - 位置, 9-4
 - 位置の移動, 9-8, 9-10
 - オフライン, 9-8
 - オフラインにする権限, 9-7
 - オンライン, 9-8
 - オンラインおよびオフラインにする, 9-7
 - 改名, 9-8, 9-10
 - 改名する権限, 9-9
 - 数の最小値, 9-2
 - 監視, 9-12
 - 再使用, 9-5
 - サイズ, 9-4
 - 最大数, 9-2
 - 再配置の例, 9-10
 - 削除, 8-12
 - NOARCHIVELOG モード, 9-8
 - 作成, 8-3
 - 対応付けられた表領域を調べる, 8-14
 - 単一の表領域の場合の改名, 9-9
 - データ・ブロックを検証する, 9-12
 - データベース管理者のアクセス, 1-4
 - データベースのオープン中には使えない, 3-3
 - デフォルト・ディレクトリ, 9-5
 - 表示
 - V\$DBFILE および V\$LOGFILE ビュー, 2-8
 - 一般的な状態, 9-13
 - 表領域に追加する, 9-4
 - ファイル名の識別, 9-11
 - ファイル名を完全に指定する, 9-5
- データ・ブロック
 - PCTFREE 記憶領域パラメータ, 10-3
 - PCTUSED 記憶領域パラメータ, 10-5
 - オペレーティング・システムのブロックとの
 - 比較, 2-10
 - クラスタで共有される, 15-2
 - 検証する, 9-12
 - サイズ, 2-10
 - サイズの変更, 2-10
 - ～の使用領域の管理, 10-2
 - ～の領域使用方法の管理, 10-2
- データ型
 - DATE, 10-17
 - LONG, 10-17
 - NUMBER, 10-16
 - ROWID, 10-17
 - 個々の型名, 10-16
 - 使用領域, 10-16
 - まとめ, 10-18
 - 文字, 10-16
- データの整合性, 17-18
 - 整合性制約, 17-17
- データベース
 - CREATE DATABASE コマンド, 2-7
 - アクセスの制限, 3-4, 3-7
 - 移行, 2-3
 - インスタンスにマウントする, 3-6
 - オープンする
 - クローズされたデータベース, 3-7
 - 改名, 6-5
 - 可用性, 3-6
 - 監査, 22-1
 - 管理, 1-1
 - 起動
 - アクセスの制限, 3-4
 - 一般手順, 3-2
 - データベース作成前, 2-6
 - パラメータ・ファイル名, 3-3
 - グローバルなデータベース名
 - 説明, 2-9
 - 分散システム, 2-9
 - 計画, 1-18
 - 構造
 - 分散データベース, 1-18
 - 削除, 2-8
 - 作成
 - およびオープン, 1-19
 - 問題の解決, 2-8
 - 制御ファイルの指定, 2-9
 - 制御ファイル, 6-2
 - 設計
 - インプリメント, 1-19
 - 調整
 - 責任, 1-20
 - 大規模データベースのアーカイブ, 23-8
 - 停止, 3-8
 - テスト, 19-8
 - データ・ファイルと REDO ログ・ファイルの

- 表示, 2-8
- データベースのマウント, 3-3
- 名前
 - インスタンスの起動, 3-2
 - 説明, 2-9
 - 対立, 2-9
- ハードウェア評価, 1-18
- 排他モード, 3-5
- バックアップ
 - 全体バックアップ, 2-7
 - データベース作成後, 1-19
- パラレル・モード, 3-5
- 物理構造, 1-18
- 本番, 19-8, 19-10
- ユーザーの責任, 1-3
- 論理構造, 1-18
- データベース・リンク
 - ジョブ・キューと~, 7-9
 - 制御するための権限, 21-9
- データベース管理者, 1-2
 - アプリケーション管理者, 19-10
 - オペレーティング・システム・アカウント, 1-4
 - 初期優先順位, 1-17
 - 責任, 1-2
 - セキュリティ, 19-7
 - セキュリティ管理者との関係, 1-3, 19-2
 - セキュリティと権限, 1-4
 - パスワード・ファイル, 1-6
 - ユーザー名, 1-4
 - ユーティリティ, 1-16
- ロール
 - セキュリティ, 19-7
 - 説明, 1-5
- データベース設計のインプリメント, 1-19
- データベースのオープン
 - 作成後, 1-19
 - マウントされたデータベース, 3-7
- データベースの起動
 - 一般手順, 3-2
 - 説明, 3-1
- データベースの停止, 3-1
- データベースの物理構造, 1-18
- データベースのマウント, 3-3
 - 排他モード, 3-5
 - パラレル・モード, 3-5
- データベースの論理構造, 1-18
- テープ・ドライブ

- アーカイブのためのストリーミング, 23-9
- テスト
 - データベースのセキュリティ, 19-8
- デフォルト
 - 一時表領域, 20-12
 - 監査オプション, 22-11
 - 使用禁止, 22-13
 - 表領域割当て制限, 20-13
 - プロファイル, 20-17
 - ユーザー表領域, 20-12
 - ロール, 20-15

と

- 問合せサーバー・プロセス
 - 説明, 4-16
- 統計
 - 更新, 17-4
- 動的パフォーマンス表
 - 使用, 4-12
- 登録
 - データベース・ユーザー, 1-20
- トランザクション
 - 特定のロールバック・セグメントに割り当て
る, 18-13
 - ロールバック・セグメントと~, 18-13
- トランザクション・エントリ
 - 記憶領域のガイドライン, 10-9
- トリガー
 - 監査, 22-20
 - 削除した表と~, 12-9
 - 使用可能および使用禁止にする権限, 17-11
 - 使用可能にする, 17-11
 - 使用禁止, 17-11
 - 制御するための権限, 21-9
 - 例, 22-20
- 取消し
 - 権限とロール
 - SYSOPER/DBA 権限, 1-12
- 取消し、権限とロール
 - REVOKE コマンド, 21-18
 - オブジェクト権限のショートカット, 21-10
 - オペレーティング・システム・ロールを使う
とき, 21-24
 - 選択した列, 21-19
- トレース・ファイル
 - 位置, 4-14

- 書き込む時期, 4-14
- サイズ, 4-14
- 使用, 4-13, 4-14
- ジョブの失敗と~, 7-10
- ログ・ライター, 4-14

な

- 名前付きユーザーの制限, 20-5
- 初期設定, 2-13

に

- 認可
 - オペレーティング・システムのロール管理, 21-14
 - ロール
 - 説明, 21-13
 - マルチスレッド・サーバー, 21-14
 - ロールに対して省略, 21-14
 - ロールに対する変更, 21-15

- 認証
 - オペレーティング・システム, 1-7
 - データベースの管理する, 20-7
 - パスワード・ファイル, 1-8
 - パスワード方針, 19-4
 - 変更, 20-15
 - ユーザー, 19-2, 20-7, 20-9
 - ユーザー作成時の指定, 20-11

ね

- ネットワーク・プロトコル
 - それぞれのディスパッチャ, 4-6

は

- ハードウェア
 - 評価, 1-18
- 排他モード
 - データベース, 3-5
 - 残りのユーザー・セッションを停止する, 4-19
 - ロールバック・セグメントと~, 18-3
- ハッシュ・クラスタ
 - 管理, 16-1
 - キーの選択, 16-5
 - 記憶領域の見積もり, 16-3
 - クラスタ, 16-1

- 削除, 16-8
- 作成, 16-4
- 使用方法, 16-2
- 変更, 16-8
- 領域使用の制御, 16-5
- 例, 16-7
- バージョン, 1-20
 - 他の Oracle ソフトウェア, 1-21

- バグ修正, 1-20
- バックアップ
 - アーカイブの影響, 23-2
 - データベース作成前, 2-4
 - データベースの新規作成後
 - ガイドライン, 1-19
 - 全体バックアップ, 2-7

- バックグラウンド・プロセス
 - Oracle8 プロセス, 4-12

- バッファ
 - SGA のバッファ・キャッシュ, 2-11

- パーティション
 - 索引から削除する, 11-7
 - 索引に追加する, 11-5
- パーティション・オブジェクト, 11-1 ~ 11-15
 - 移動, 11-4
 - 切捨て, 11-7
 - 作成, 11-2
 - 追加, 11-4
 - 定義, 11-2
 - パーティションを分割する, 11-9
 - マージする, 11-10
 - メンテナンス時にアプリケーションを休止する, 11-15
 - メンテナンスする, 11-3 to 11-15

- パーティション・ビュー
 - パーティション表に変換する, 11-13

- パーティション表
 - パーティション化されていない状態に変換する, 11-11
 - パーティションを追加する, 11-4
 - パーティションを分割する, 11-9
 - パーティションをマージする, 11-10, 11-12

- パスワード
 - REMOTE_LOGIN_PASSWORD パラメータを設定する, 1-10
 - SYS と SYSTEM の初期パスワード, 1-4
 - 認証ファイル, 1-8
 - パスワード・ファイル

- 1-11
- OS 認証, 1-6
- 位置の移動, 1-14
- 削除, 1-14
- 作成, 1-8
- 状態, 1-15
- 変更するための権限, 20-14
- ユーザー・パスワードの変更, 20-15
- ユーザー認証, 20-7
- ユーザーのセキュリティ方針, 19-4
- ロール, 21-13
- ロールに対する変更, 21-15
- ロールを変更するための権限, 21-13
- パッケージ
 - 再コンパイル, 17-24
 - 再コンパイルのための権限, 17-24
- パッチ・リリース番号, 1-21
- パフォーマンス
 - アーカイブの調整, 23-8
 - データ・ファイルの位置と~, 9-4
- パフォーマンス表
 - 動的パフォーマンス表, 4-12
- パブリック
 - シノニム, 13-11
- パブリック・ロールバック・セグメント
 - オフラインにする, 18-13
 - 使用可能にする, 18-10
- パラメータ・ファイル
 - 位置, 3-13
 - インスタンス起動のデフォルト, 3-3
 - 数, 3-13
 - 起動のため選択, 3-3
 - キャラクタ・セット, 3-13
 - 個々のパラメータ名, 2-9
 - サンプル, 3-13
 - 使用, 3-12
 - データベース作成のために作る, 2-4
 - データベース作成の前に編集する, 2-5
 - 編集, 3-12
- パラレル・モード
 - データベース, 3-5
- パラレル問合せオプション
 - サーバー・プロセスの数, 4-16
 - 索引作成の並行化, 14-4
 - 問合せサーバー, 4-16
 - 表作成の並行化, 12-4

ひ

ビュー

- FOR UPDATE 句と~, 13-2
- ORDER BY 句と~, 13-2
- WITH CHECK OPTION, 13-2
- 依存性の表示, 17-30
- エラー付きで作る, 13-3
- 置換え, 13-8
- 置き換えるための権限, 13-8
- 管理, 13-1, 13-9
- 権限, 13-2
- 再コンパイル, 17-23
- 再コンパイルのための権限, 17-23
- 削除, 13-9
- 削除した表と~, 12-9
- 削除するための権限, 13-9
- 作成, 13-2
- パーティション化
 - パーティション表に変換する, 11-13
- ワイルドカード, 13-3

表

- SYSTEM 表領域と~, 12-3
- 位置, 12-3, 12-6
- 一時領域と~, 12-6
- エクステンツの割当て, 12-8
- 回復不能 (UNRECOVERABLE), 12-4
- 管理, 12-1
- 管理のガイドライン, 12-1, 12-6
- キー保存, 13-5
- 記憶領域パラメータ, 10-10
- 切捨て, 17-9
- クラスタ化された, 15-2
- クラスタ化された表
 - 管理, 15-1
 - 記憶領域, A-15
 - 削除, 15-9
 - 削除するための権限, 15-10
 - 作成, 15-6
 - 変更, 15-9
- クラスタ化のスキーマ, 15-7
- 構造の妥当性検査, 17-8
- サイズの見積もり, 12-5, A-5
- 索引と~, 14-2
- 索引との分離, 12-5
- 索引の制限, 14-3
- 削除, 12-8

- 削除するための権限, 12-8
- 作成, 12-6
- 作成の並行化, 12-4
- 作成前の設計, 12-2
- 初期サイズの見積もり, A-2
- 作するための権限, 12-6
- 統計分析, 17-3
- トランザクション・パラメータ, 12-3
- ~のための PCTFREE の指定, 10-4
- ハッシュ・クラスタ
 - 管理, 16-1
 - 作成, 16-4
- パーティションを追加する, 11-5
- 表領域の指定, 12-3, 12-6
- 変更, 12-7, 12-8
- 変更するための権限, 12-7
- 列の長さの拡張, 12-7
- 評価
 - Oracle8 Server のハードウェア, 1-18
- 表のパーティション
 - 切捨て, 11-7
 - グローバルな索引を含む, 11-5
 - 削除, 11-5
 - 作成, 11-2
 - 分割する, 11-9
 - マージする, 11-10
 - 隣接パーティションをマージする, 11-12
 - 交換する, 11-11
- 表領域
 - SYSTEM 表領域, 8-3
 - 空き領域をリストする, 8-14
 - 合わせる, 8-6
 - 位置, 9-4
 - 一時, 20-12
 - 一時オフラインにする, 8-8
 - オフラインにする権限, 8-8
 - オンラインにする, 8-7
 - 書込み可能, 8-11
 - 可用性の変更, 8-7
 - 監視, 9-12
 - 管理のガイドライン, 8-2
 - 記憶領域設定の変更, 8-5
 - 削除
 - 説明, 8-12
 - 必要な権限, 8-12
 - 作成, 8-3
 - 作成するための権限, 8-4

- 追加の作成, 8-3
- 通常のアラインにする, 8-8
- データ・ファイルの追加, 9-4
- デフォルト、一時, 20-12
- デフォルト記憶領域パラメータを調べる, 8-14
- デフォルトの記憶領域パラメータの設定, 8-2
- デフォルトの割当て制限, 20-13
- ~のデフォルトの記憶領域パラメータ, 10-9
- 必要なロールバック・セグメント, 8-4
- ファイルをリストする, 8-14
- 複数を使う, 8-2
- 無制限の割当て制限, 20-13
- ユーザーからの取消し, 20-13
- ユーザーにデフォルトを割り当てる, 20-12
- ユーザーの割当て制限, 20-12
- ユーザー割当て制限を割り当てる, 8-3
- 読み込み専用, 8-10
- 割当て制限
 - 割り当てる, 8-3
- 割当て制限の表示, 20-23
- ヒント
 - オブジェクト監査のためのショートカット, 22-9
 - オブジェクト権限のショートカット, 21-10
 - 文監査ショートカット, 22-7

ふ

- ファイル
 - OS によるオープン数の制限 OS, 8-2
- ファンクション
 - 再コンパイル, 17-23
- 複合制限, 20-19
 - コスト, 20-19
 - サービス単位, 20-19
- プライベート
 - シノニム, 13-11
 - ロールバック・セグメント, 18-8
 - オフラインにする, 18-13
- ブランチ・ブロック
 - 必要な領域, A-9
- プログラム・グローバル領域 (PGA)
 - MAX_ENABLED_ROLES の効果, 21-15
- プロシージャ
 - 再コンパイル, 17-23
- プロセス
 - SNP バックグラウンド・プロセス, 7-2
- プロファイル, 20-17

- PUBLIC_DEFAULT, 20-17
- 管理, 20-17
- 削除, 20-20
- 削除するための権限, 20-20
- 作成, 20-17
- 制限を NULL に設定, 20-18
- デフォルト, 20-17
- 表示, 20-24
- 複合制限, 20-19
- 変更, 20-18
- 変更するための権限, 20-18
- ユーザーに割り当てる, 20-18
- リスト, 20-21
- リソース・コストを設定するための権限, 20-19
- リソース制限を使用可能にする, 20-20
- リソース制限を使用禁止にする, 20-20
- 分散、I/O, 2-15
- 分散処理
 - パラメータ・ファイルの位置, 3-13
- 分散データベース
 - ARCHIVELOG モードで稼働, 23-3
 - NOARCHIVELOG モードで稼働, 23-3
 - リモート・インスタンスの起動, 3-6
- 分析する、オブジェクトを
 - 権限, 17-3
 - 説明, 17-3

へ

- 変更
 - 記憶領域パラメータ, 12-7
 - クラスタ, 15-8
 - クラスタ化された表, 15-9
 - クラスタ索引, 15-9
 - 結合ビュー, 13-4
 - 索引, 14-8
 - 順序, 13-10
 - データベース状態, 3-6
 - ハッシュ・クラスタ, 16-8
 - パスワード, 20-15
 - パブリック・ロールバック・セグメント, 18-9
 - 表, 12-7, 12-8
 - 表領域に対する記憶領域, 8-5
 - ユーザー, 20-14
 - ロールバック・セグメント記憶領域パラメータ, 18-9
- 変更可能な結合ビュー

- 定義, 13-4

ま

- マークの付けられたユーザー・セッション, 4-19
- マルチスレッド・サーバー
 - OS ロール管理制限, 21-25
 - OS ロール認可での制限, 21-14
 - 起動, 4-5
 - 共有プールと~, 4-5
 - サービス名, 4-6
 - 使用可能および使用禁止にする, 4-8, 4-9
 - 専用サーバーと対比, 4-3
 - ディスパッチャの構成, 4-6
 - データベース起動, 3-2

み

- ミラー化
 - 制御ファイル, 2-10
- ミラー化された制御ファイル
 - 重要性, 6-2
- ミラー化した REDO ログ・ファイル
 - 位置, 5-3
 - サイズ, 5-3

め

- メモリー
 - ユーザーごとの表示, 20-25
- メンテナンス・リリース番号, 1-21

も

- モード
 - 制限, 3-4, 3-7
 - 排他, 3-5
 - パラレル, 3-5

ゆ

- ユーザー
 - PUBLIC グループ, 21-21
 - 新しく作ったデータベース, 2-13
 - 一般のユーザーのセキュリティ, 19-4
 - エンド・ユーザーのセキュリティ方針, 19-5
 - 数の制限, 2-13

- 管理, 20-10
- 権限の管理に関する方針, 19-5
- 削除, 20-16
- 削除後のオブジェクト, 20-16
- 削除するための権限, 20-16
- 識別, 20-7
- 情報の表示, 20-23
- セキュリティ, 19-2
- セッション、停止, 4-20
- 他と重複しないユーザー名, 2-13, 20-5
- 作するための権限, 20-10
- デフォルト・ロールの変更, 20-15
- デフォルト表領域, 20-12
- 登録, 1-20
- 認証
 - 説明, 19-2, 20-7
 - データベース認証, 20-7
- 認証方法の変更, 20-15
- パスワード・セキュリティ, 19-4
- パスワードの変更, 20-15
- パスワードを変更するための権限, 20-14
- 表領域割当て制限, 20-12
- 表領域割当て制限の表示, 20-23
- 表領域割当て制限を割り当てる, 8-3
- 複合制限, 20-19
- 付与される権限のリスト, 21-27
- 付与されるロールのリスト, 21-27
- プロファイルの削除, 20-20
- プロファイルを割り当てる, 20-18
- 変更, 20-14
- マルチバイト・キャラクタ
 - 名前, 20-11
 - パスワード, 20-11
- 無制限の割当て制限の割当て, 20-13
- メモリー使用の表示, 20-25
- ユーザー名を指定する, 20-11
- リスト, 20-21
- ロールの削除, 21-15
- ユーザー・セッション・ダイアログを停止する, 4-19
- ユーザー名
 - SYS と SYSTEM, 1-4
- ユーティリティ
 - SQL*Loader, 1-16
 - データベース管理者のための, 1-16

よ

読み専用表領域

- WORM デバイス上の~, 8-11
- 書込み可能にする, 8-11
- 作成, 8-10
- データ・ファイル, 9-7

ら

ライセンス

- 制限の参照, 20-6
- セッション制限を変更するための権限, 20-4
- セッションをベースとした, 20-2
- 同時使用, 20-2
- 同時実行セッションの数, 2-12
- 名前付きユーザー, 20-2, 20-5
- 名前付きユーザー制限を変更するための権限, 20-6
- ライセンス契約に従う, 2-12, 20-2

り

リスナー・プロセス

- MTS_LISTENER_ADDRESS の設定, 4-5
- 構成ファイル, 4-6

リソース

- プロファイル, 20-17

リソース制限

- NULL に設定, 20-18
- PUBLIC_DEFAULT プロファイル, 20-17
- コスト, 20-19
- コストを設定するための権限, 20-19
- サービス単位, 20-19
- 使用可能、使用禁止にするための権限, 20-20
- 使用可能にする, 20-20
- 使用禁止, 20-20
- 複合制限, 20-19
- プロファイル, 20-17
- プロファイルで変更する, 20-18
- プロファイルで割り当てる, 20-18
- プロファイルの作成, 20-17

リモート接続

- INTERNAL として接続, 1-13, 1-14, 1-15, 1-16, 1-18, 1-19, 1-20, 1-21, 1-22
- SYSOPER/SYSDBA として接続, 1-13
- パスワード・ファイル, 1-8

領域

- 索引による使用, 14-8
- データベースに追加, 8-3

領域管理

- PCTFREE, 10-2
- PCTUSED, 10-4

リリース

- Oracle8 Server の識別, 1-20
- 他の Oracle ソフトウェアのバージョン, 1-21
- パッチ・リリース番号, 1-21
- ポート固有のリリース番号, 1-21
- メンテナンス・リリース番号, 1-21
- リリース番号を調べる, 1-22

リレーショナル設計

- 計画, 1-18

履歴表

- 時間枠を移動する, 11-14

れ

例

- 索引の変更, 14-8
- 制約の作成, 17-18

例外

- 整合性制約, 17-20

列

- INSERT 権限, 21-18
- 権限, 21-18
- 権限の取消し, 21-19
- 権限の付与, 21-18
- 情報の表示, 17-30
- 選択した列についての権限の付与, 21-17
- 長さの拡張, 12-7
- 付与されるユーザーのリスト, 21-28

- 連鎖的な取消し, 21-20

ろ

ロール

- ADMIN OPTION, 21-16
- CONNECT ロール, 21-12
- DBA ロール, 1-5, 21-12
- EXP_FULL_DATABASE, 21-12
- GRANT OPTION, 21-18
- GRANT コマンド, 21-25
- IMP_FULL_DATABASE, 21-12
- OS 管理とマルチスレッド・サーバー, 21-25

- RESOURCE ロール, 21-12

- REVOKE コマンド, 21-25

- SET ROLE コマンド, 21-25

- アプリケーションの開発者, 19-9

- 一意の名前, 21-11

- オペレーティング・システムでの付与, 21-23, 21-25

- オペレーティング・システム認可, 21-14

- オペレーティング・システムを使った管理, 21-22

- 下位互換性, 21-12

- 管理, 21-11

権限

- 削除するための~, 21-15

- システム権限またはロールの付与, 21-16

- 作するための~, 21-11

- 認可方法の変更, 21-13

- パスワードの変更, 21-13

- 権限とロールのリスト, 21-29

- 権限付与のリスト, 21-27

- 削除, 21-15

- 使用可能にするためのパスワード, 21-13

- 事前定義済み, 1-5, 21-12

- セキュリティ, 19-5

- データベース認可, 21-13

- デフォルト, 20-15

- 取消し, 21-18

- 認可, 21-13

- 認可なし, 21-14

- 認可の変更, 21-15

- パスワードの変更, 21-15

- パスワードのマルチバイト・キャラクタ, 21-13

付与

- 説明, 21-16

- マルチスレッド・サーバー, 21-14

- マルチバイト・キャラクタ

- 名前, 21-11

- リスト, 21-29

- ロールで分類, 21-11

ロールバック・セグメント

- AVAILABLE, 18-11

- OFFLINE, 18-11

- PARTLY AVAILABLE, 18-11

- PENDING OFFLINE, 18-12

- SYSTEM ロールバック・セグメント, 18-3

- 位置, 18-7

- エクステンツのリスト表示, 17-31

- オフライン・ロールバック・セグメント, 18-10

オフラインかどうかの検査, 18-12
オフライン状態, 18-12
オフラインにする, 18-12
オンライン・ロールバック・セグメント, 18-10
オンライン状態, 18-12
数の選択, 2-14
監視, 18-6, 18-15
管理, 18-1
管理のガイドライン, 18-2
記憶領域パラメータ, 18-8
記憶領域パラメータと~, 18-8
記憶領域パラメータの変更, 18-9
起動時に獲得する, 2-11
権限
 削除するための~, 18-14
 作るために必要な, 18-8
 変更するために必要な, 18-10
サイズの設定, 18-4
サイズを選択, 2-14
サイズを小さくする, 18-10
削除, 18-14
削除の状態, 18-14
作成, 18-8
使用可能にする, 18-10
初期, 18-2
自動的に獲得, 18-3, 18-12
状態, 18-11
状態変更
 PARTLY AVAILABLE セグメントをオンライン
 にする, 18-11
 オンライン, 18-11
 新規作成時にオンラインにする, 18-8
 自動的にオンラインにする, 18-12
すべての名前の表示, 18-15
遅延, 18-16
データベースを作った後に作る, 18-3
等サイズのエクステント, 18-5
トランザクションと~, 18-13
パブリックとプライベートを作る, 18-3
パブリックの変更, 18-9
表示
 PENDING OFFLINE セグメント, 18-16
 情報, 18-15
 すべての遅延ロールバック・セグメント, 18-16
 遅延ロールバック・セグメント, 18-16
表領域をオフラインにすることと~, 8-10

複数を使用する, 18-2
無効な状態, 18-14
明示的にトランザクションを割り当てる, 18-13
割り当てる, 2-14
ログ・スイッチ
 強制的, 5-12
 権限, 5-12
 チェックポイントと~, 5-10
ログ・ファイルの消去, 5-13
ログ・ライター・プロセス (LGWR)
 トレース・ファイルの監視, 4-14
ログ順序番号, 5-2
ロック
 監視, 4-11
 ジョブ・キュー, 7-10

わ

ワイルドカード
 ビュー, 13-3
割当て
 一時領域, 12-6
 エクステント, 12-8
 クラスタのエクステント, 15-9
 マルチスレッド・サーバー, 4-5
 ロールバック・セグメントのエクステントの最小化
 , 18-13
割当て制限
 一時セグメントと~, 20-13
 ゼロに設定, 20-13
 表示, 20-23
 表領域, 20-12
 表領域割当て制限, 8-3
 無制限, 20-13
 ユーザーからの取消し, 20-13
 リスト, 20-21

「インスタンス」メニュー
 「オープン」オプション, 3-7
 接続オプションの抑止, 3-8
「インスタンス起動」ダイアログ・ボックス, 3-2
 「アンマウント」ラジオ・ボタン, 3-3
 「オープン」ラジオ・ボタン, 3-4
 「強制」チェック・ボックス, 3-5
 「データベースの制限」チェック・ボックス, 3-4

- 「マウント」ラジオ・ボタン, 3-4
- パラメータ・ファイルの指定, 3-3
- 「オンライン REDO ログ・メンバーの追加」ダイアログ, 5-6
- 「オンライン REDO ログ・グループの追加」ダイアログ・ボックス, 5-5
- 「オンライン REDO ログ・メンバー改名」ダイアログ・ボックス, 5-7
- 「オンライン REDO ログ・メンバーの削除」ダイアログ・ボックス, 5-10
- 「オンライン REDO ログの削除」ダイアログ, 5-8
- 「システム権限/ロールの取消し」ダイアログ, 21-18
- 「手動アーカイブ開始」ダイアログ・ボックス, 23-8
- 「自動アーカイブ開始」ダイアログ, 23-12
- 「自動アーカイブ停止」メニュー・オプション, 23-7
- 「チェックポイント強制」メニュー・オプション, 5-12
- 「停止」メニュー, 3-8
 - 「即時」オプション, 3-11
 - 「標準」オプション, 3-10
 - インスタンスの中断オプション, 3-12
- 「データ・ファイル改名」ダイアログ, 9-9
- 「表領域削除」ダイアログ・ボックス, 8-12
- 「表領域作成」ダイアログ, 8-4
- 「表領域に対する記憶領域の設定」ダイアログ・ボックス, 8-5
- 「表領域にデータ・ファイルを追加」ダイアログ, 9-5
- 「表領域をオンラインに設定」ダイアログ, 8-7
- 「プロファイルの削除」ダイアログ, 20-20
- 「プロファイルの作成」ダイアログ, 20-17
- 「プロファイルの変更」ダイアログ, 20-18
- 「ユーザーの削除」ダイアログ, 20-16
- 「ユーザー変更」ダイアログ, 20-14
- 「ロールの変更」ダイアログ, 21-15
- 「ロールバック・セグメント記憶領域の設定」ダイアログ, 18-8
- 「ロールバック・セグメント記憶領域の変更」ダイアログ, 18-9
- 「ロールバック・セグメント作成」ダイアログ, 18-8
- 「ロールバック・セグメントの削除」ダイアログ, 18-14
- 「ロールバック・セグメントをオフラインに設定」ダイアログ, 18-12
- 「ロールバック・セグメントをオンラインに設定」ダイアログ, 18-11
- 「ログ・スイッチの強制」メニュー・オプション, 5-12

