Oracle8*i*

レプリケーション・ガイド

リリース 8.1

2000 年 2 月

部品番号：J00936-01

ORACLE®

Oracle8*i* レプリケーション・ガイド , リリース 8.1

部品番号：J00936-01

原本名：Oracle8*i* Replication, Release 2 (8.1.6)

原本部品番号：A76959-01

原本著者：William Creekbaum and Randy Urbano

原本協力者 : Valarie Moore, Sukanya Balaraman, Ruth Baylis, Alan Downing, Al Demers, Ira Greenberg, Maria Pratt, Gordon Smith, Jim Stamos, Curt Elsbernd, Pat McElroy, Wayne Smith, Neerja Shodhan, Mahesh Subramaniam, Lik Wong

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

Printed in Japan.

制限付権利の説明

プログラム（ソフトウェアおよびドキュメントを含む）の使用、複製または開示は、オラクル社との契約に記された制約条件に従うものとします。著作権、特許権およびその他の知的財産権に関する法律により保護されています。

当プログラムのリバース・エンジニアリング等は禁止されております。

このドキュメントの情報は、予告なしに変更されることがあります。オラクル社は本ドキュメントの無謬性を保証しません。

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

危険な用途への使用について

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

Restricted Rights Notice

Programs delivered subject to the DOD FAR Supplement are "commercial computer software" and use, duplication, and disclosure of the Programs, including documentation, shall be subject to the licensing restrictions set forth in the applicable Oracle license agreement. Otherwise, Programs delivered subject to the Federal Acquisition Regulations are "restricted computer software" and use, duplication, and disclosure of the Programs shall be subject to the restrictions in FAR 52.227-19, Commercial Computer Software - Restricted Rights (June, 1987). Oracle Corporation, 500 Oracle Parkway, Redwood City, CA 94065.

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

目次

はじめに ix

『Oracle8*i* レプリケーション・ガイド リリース 8.1』の概要 x
対象読者 xi
前提知識 xi
このマニュアルの構成 xi
表記規則 xiii
特記事項 xiii
本文について xiii
コード例 xiv

1　レプリケーションの概要

レプリケーションの概要 1-2
レプリケーションを使用するアプリケーション 1-3
レプリケーション・オブジェクト、グループおよびサイト 1-4
レプリケーション環境のタイプ 1-5
マルチマスター・レプリケーション 1-5
スナップショット・レプリケーション 1-7
マルチマスターおよびスナップショットのハイブリッド構成 1-11
レプリケーション環境の管理ツール 1-12
Oracle Replication Manager 1-13
レプリケーション・マネージメント API 1-14
レプリケーション・カタログ 1-14
分散スキーマ管理 1-14
レプリケーションの競合 1-15
マルチマスター・レプリケーションのその他のオプション 1-16

同期レプリケーション 1-16
プロシージャ・レプリケーション 1-16

2 マスター・レプリケーションの概念およびアーキテクチャ

マスター・レプリケーションの概念 2-2
レプリケーションとは 2-2
マルチマスター・レプリケーションの使用目的 2-3
マスター・レプリケーションのタイプ 2-6
マルチマスター・レプリケーション・プロセス 2-7
競合解消の概念 2-10
マスター・レプリケーションのアーキテクチャ 2-11
マスター・サイトのメカニズム 2-11
管理メカニズム 2-20
編成メカニズム 2-26
伝播プロセス 2-29
パフォーマンス・メカニズム 2-36
競合解消メカニズム 2-41

3 スナップショットの概念およびアーキテクチャ

スナップショットの概念 3-2
スナップショットとは 3-2
スナップショットの使用目的 3-3
使用可能なスナップショット 3-4
スナップショットでのデータのサブセット化 3-8
マスター・サイトでのスナップショットの登録 3-16
スナップショットのアーキテクチャ 3-17
マスター・サイトのメカニズム 3-19
スナップショット・サイトのメカニズム 3-21
編成メカニズム 3-23
リフレッシュ・プロセス 3-28
スナップショットの準備 3-31
スナップショット・サイト・ユーザーの作成 3-31
マスター・サイト・ユーザーの作成 3-31
スキーマ 3-32

データベース・リンク 3-32
権限 3-33
マスター・サイトでのスケジュール・パージ 3-34
プッシュのスケジュール 3-34
SNP バックグラウンド・プロセスおよび間隔 3-34
スナップショット・ログの作成 3-37
フィルタ列の使用 3-37
スナップショット環境の作成 3-38
Replication Manager 3-38
レプリケーション・マネージメント API 3-39

4 配置テンプレートの概念およびアーキテクチャ

大量配布の課題 4-2
配置テンプレートおよび大量配布の目標 4-3
Oracle 配置テンプレートの概念 4-3
配置テンプレートの要素 4-4
配置テンプレートのパッケージ化およびインスタンシエーション 4-9
配置テンプレートのアーキテクチャ 4-14
システム表に格納されたテンプレート定義 4-14
パッケージ化およびインスタンシエーションのプロセス 4-16
インスタンシエーション後の作業 4-20
配置テンプレートの設計 4-22
垂直パーティション化 4-22
割当て表による水平パーティション化 4-24
データ・セット 4-26
設計に関するその他の考慮事項 4-28
スナップショット作成のローカル制御 4-28

5 競合解消の概念およびアーキテクチャ

競合解消の概念 5-2
データ要件およびアプリケーション要件の理解 5-2
レプリケーションの競合のタイプ 5-3
競合の検出 5-5
競合解消 5-6
競合の回避 5-7

競合解消のアーキテクチャ 5-10
サポート・メカニズム 5-10
更新競合の一般的な解消方法 5-11
更新競合のその他の解消方法 5-15
一意性競合の解消方法 5-24
削除による競合解消方法 5-26
パフォーマンス・メカニズム 5-26

6 高度な設定

プロシージャ・レプリケーションの使用方法 6-2
プロシージャ・レプリケーションの制限事項 6-2
トランザクションのシリアライズ化 6-3
レプリケート・プロシージャのサポートの生成 6-4
耐障害性を実現するための設計 6-7
Oracle Parallel Server とレプリケーション 6-8
耐障害性を実現するための設計 6-8
耐障害性の高いシステムの実現 6-9
動的所有権の競合回避の使用方法 6-11
ワークフロー 6-11
トークン渡し 6-12
行の所有権の検索 6-15
所有権の取得 6-15
変更の適用 6-16
変更をレプリケートしない表の変更 6-17
レプリケーション機能を使用禁止にする 6-18
レプリケーション機能を再び使用可能にする 6-19
トリガーおよびレプリケーション 6-19
スナップショットのレプリケーションを使用可能または使用禁止にする 6-19
レプリケーションの保護メカニズムについて 6-20
データ伝播の依存性の維持 6-21

7 レプリケーション環境の計画

レプリケート表に関する考慮事項 7-2
主キー 7-2
データ型に関する考慮事項 7-2

初期化パラメータ 7-3
マスター・サイトおよびスナップショット・サイト 7-8
マスター・サイトの利点 7-9
スナップショット・サイトの利点 7-9
スケジュール・リンクのガイドライン 7-10
定期的なプッシュのスケジューリング 7-10
連続的なプッシュのスケジューリング 7-11
遅延トランザクション・キューのスケジュール・パージのガイドライン 7-11
定期的なパージのスケジューリング 7-12
連続的なパージのスケジューリング 7-12
シリアル伝播およびパラレル伝播 7-12
配置テンプレート 7-13
配置テンプレートをインスタンシエートするスナップショット・サイトの準備 7-13
競合解消 7-16
セキュリティ 7-16

8 Replication Manager の概要

Replication Manager の用途 8-2
Replication Manager の起動 8-3
Oracle Enterprise Manager Console からの Replication Manager の起動 8-3
独立のアプリケーションとしての Replication Manager の起動 8-3
Replication Manager へのログイン 8-3
Replication Manager のインタフェース 8-4
ペイン 8-5
ツールバー 8-10
メニュー 8-13
Replication Manager のウィザード 8-17
Setup Wizard 8-18
Snapshot Group Wizard 8-20
Deployment Template Wizard 8-22
Offline Instantiation Wizard 8-24
Copy Template Wizard 8-26
レプリケート環境を作成するためのフローチャート 8-28

A 新機能

Oracle8*i* の新しいレプリケーション機能 A-2
パフォーマンスの改善 A-2
大量配布サポートの改善 A-2
セキュリティの改善 A-4
Replication Manager A-4
Oracle8*i* Lite の統合の改善 A-5
Oracle8 の新しいレプリケーション機能 A-5
パフォーマンスの強化 A-5
副問合せに基づくデータのサブセット化 A-6
ラージ・オブジェクト型（LOB）のサポート A-7
管理方法の改善と使用方法の簡易化 A-7
システムベースのセキュリティ・モデルの強化 A-8
Replication Manager の新機能 A-9

B レプリケーションに関する問題のトラブルシューティング

データベース・リンクに関する問題の診断方法 B-2
マスター・サイトに関する問題の診断方法 B-2
新しいマスター・サイトでレプリケート・オブジェクトが作成されない場合 B-2
DDL による変更がマスター・サイトに伝播されない場合 B-3
DML による変更が他のサイトに非同期で伝播されない場合 B-3
DML をレプリケート表に適用できない場合 B-4
バルク更新および制約違反 B-4
レプリケート・オブジェクトの再作成 B-4
表のレプリケーション・サポートを生成できない場合 B-4
レプリケート・プロシージャまたはレプリケート・トリガーに関する問題 B-5
ON DELETE CASCADE および整合性制約に関する問題 B-5
遅延トランザクション・キューに関する問題の診断方法 B-6
スケジュール・リンクのジョブのチェック B-6
分散トランザクションに関する問題 B-6
データベース・リンク指定の不備 B-6
不適切なレプリケーション・カタログ・ビュー B-6
スナップショットに関する問題の診断方法 B-7
スナップショット・サイトでのレプリケート・オブジェクトの作成に関する問題 B-7
リフレッシュに関する問題のトラブルシューティング B-7

リフレッシュに関する問題の高度なトラブルシューティング B-9

C RepAPI 用の Oracle8*i* Server の構成

RepAPI とは C-2
RepAPI の概念およびアーキテクチャ C-3
サーバー・コンポーネント C-4
コールアウト・クラス C-5
クライアント・エンジン C-5
Oracle8*i* Server で RepAPI を使用する前の作業 C-7
RepAPI サーバー・オブジェクトのロードおよび公開 C-9
オフライン・インスタンシエーション・ファイルの生成 C-14
RepAPI の管理 C-16
Replication Manager C-16
レプリケーション・マネージメント API C-16
レプリケーション・カタログ C-16
Oracle Client Replication Tool C-16
クライアントのサンプル・プログラム C-17
サンプル・プログラムのインストール C-17
サンプル・プログラムの使用方法 C-17

索引

はじめに

ここでは次のトピックについて説明します。

- 『Oracle8i レプリケーション・ガイド リリース 8.1』の概要
- 対象読者
- このマニュアルの構成
- 表記規則

このマニュアルには、Oracle8*i* および Oracle8*i* Enterprise Edition の機能に関する情報が含まれています。Oracle8*i* と Oracle8*i* Enterprise Edition の基本的な機能は同じです。ただし、Enterprise Edition でしか利用できない高度な機能もあります。そのような機能のいくつかはオプションとなっています。たとえば、パーティション化を行うには、Enterprise Edition とパーティション化オプションが必要です。

『Oracle8*i* レプリケーション・ガイド リリース 8.1』の概要

このマニュアルでは、Oracle8*i* Server のレプリケーション機能について説明します。Oracle のレプリケーション機能を使用するには、Oracle アドバンスト・レプリケーション・オプションをインストールする必要があります。基本レプリケーション（読取り専用スナップショットと更新可能スナップショット）は、Oracle の標準分散機能です。プロシージャ・レプリケーションを使用するには、PL/SQL とアドバンスト・レプリケーション・オプションが必要です。

このマニュアルの情報は、あらゆるオペレーティング・システム上で稼動している Oracle8*i* Server に適用されます。このマニュアルでは、次のトピックについて説明します。

- レプリケーション
  - 読取り専用スナップショット
  - 更新可能スナップショット
  - シングル・マスター・レプリケーション
  - マルチマスター・レプリケーション（Orcle8*i* Enterprise Edition のみ）
  - 配置テンプレート
- 競合解消
- レプリケート環境の管理
- 高度なテクニック
- トラブルシューティング
- ジョブ・キューの使用方法
- 遅延トランザクション

対象読者

このマニュアルは、アドバンスト Oracle8*i* 分散システムを開発およびメンテナンスするアプリケーションの開発者とデータベース管理者（DBA）を対象にしています。

前提知識

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、および分散データベースの管理、PL/SQL（プロシージャ・レプリケーションを使用する場合）、Oracle レプリケート環境を動作させるオペレーティング・システムをよく理解している必要があります。

また、このマニュアルの読者は、次に示すマニュアルを読み、その内容を理解しておく必要があります。

- 『Oracle8*i* 概要』
- 『Oracle8*i* 管理者ガイド』
- 『Oracle8*i* 分散システム』
- 『Oracle PL/SQL ユーザーズ・ガイドおよびリファレンス』（プロシージャ・レプリケーションを使用する場合）

このマニュアルの構成

このマニュアルは、次の章および付録から構成されています。

第 1 章「レプリケーションの概要」
Oracle レプリケーションの概念および用語について説明します。

第 2 章「マスター・レプリケーションの概念およびアーキテクチャ」
マルチマスター・レプリケーションの概念およびアーキテクチャについて説明します。

第 3 章「スナップショットの概念およびアーキテクチャ」
スナップショット・レプリケーションの概念およびアーキテクチャについて説明します。この章では、スナップショット環境を作成するための前提条件についても説明します。

第 4 章「配置テンプレートの概念およびアーキテクチャ」
配置テンプレートの概念およびアーキテクチャについて説明します。この章では、配置テンプレートの設計についても説明します。

第 5 章「競合解消の概念およびアーキテクチャ」
競合解消の概念およびアーキテクチャについて説明します。この章では、競合解消方法についても説明します。

第 6 章「高度な設定」
プロシージャ・レプリケーションの使用方法、耐障害性の設計方法、動的所有権の競合回避の使用方法および変更をレプリケートせずに表を変更する方法など、レプリケーションの高度な設定について説明します。

第 7 章「レプリケーション環境の計画」
初期化パラメータの設定やレプリケーション環境の準備など、レプリケーション環境の計画について説明します。

第 8 章「Replication Manager の概要」
Oracle Replication Manager の機能を紹介します。Oracle Replication Manager は、レプリケーション環境を作成、管理および監視するための Java ベースのツールです。

付録 A「新機能」
このリリースの新機能を概説し、このマニュアルおよびその他のマニュアルでこの新機能の詳細を記載している項を示します。

付録 B「レプリケーションに関する問題のトラブルシューティング」
レプリケーションに共通する問題の診断および解決方法を説明します。

付録 C「RepAPI 用の Oracle8i Server の構成」
Java RepAPI クライアントからアクセスする Oracle8*i* Server の構成方法を説明します。Java RepAPI は、Oracle8*i* Lite クライアントで Oracle Server のデータをレプリケートするためのランタイム・ライブラリです。

このリリースでの変更点

このリリースでは、主に次のような変更があります。

- 以前のリリースでは、Replication Manager の使用方法の詳細を記載していました。このリリースでは、Replication Manager の使用方法の詳細は、Replication Manager のオンライン・ヘルプのみに掲載しています。このリリースでは、第 7 章「レプリケーション環境の計画」と第 8 章「Replication Manager の概要」の 2 つの章が追加されました。
- 付録 B「レプリケーションに関する問題のトラブルシューティング」と付録 C「RepAPI 用の Oracle8i Server の構成」の 2 つの付録が追加されました。
- 移行および互換性に関する付録は、『Oracle8*i* 移行ガイド』マニュアルに移されました。

表記規則

このマニュアルで使用する表記上の規則について説明します。

特記事項

このマニュアルの本文では、特別な情報は次のように示します。

関連項目： 詳細な情報を得られる参照箇所を示します。

注意： 特別または予備の情報や、システムまたはデータに対して発生する可能性があるダメージについての重要な情報を示します。

本文について

次に、このマニュアルの本文で使用する規則について説明します。

大文字

大文字テキストは、文のキーワード、オブジェクト名、初期化パラメータおよびデータ・ディクショナリ・ビューを明示する場合に使用します。

たとえば、" プライベート・ロールバック・セグメントを作成する場合、ロールバック・セグメントの名前は ROLLBACK_SEGMENTS 初期化パラメータに含める必要があります。" のように使用します。

二重かぎカッコと二重引用符

二重かぎカッコと二重引用符は、マニュアルのタイトルまたは語句の定義を示します。

定義の例： "*database* は、1 つの単位として扱われるデータの集合です。データベースの一般的な目的は、関連する情報を保存および検索することです。"

他の資料を参照する例： 詳細は、『Oracle8*i* パフォーマンスのための設計およびチューニング』を参照してください。

コード例

SQL、Server Manager の行モードおよび SQL*Plus の文は、本文と区別するために、固定幅フォントで示されています。次に例を示します。

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

例文には、カンマや引用符などの句読点を含むことがあります。例文の句読点は、すべて必須です。例文は、すべてセミコロン（;）で終了します。セミコロンまたはその他の終了文字が文を終了するために必須であるかどうかは、アプリケーションによって異なります。

例文の大文字は、Oracle SQL のキーワードを示しています。ただし、文を発行するときには、キーワードの大文字と小文字は区別されません。

例文の小文字は、その単語が単なる例として使用されていることを示しています。たとえば、表、列、ファイルの名前などを表します。

1

レプリケーションの概要

この章では、Oracle レプリケーションに関する基本的な概念および用語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。

- レプリケーションの概要
- レプリケーションを使用するアプリケーション
- レプリケーション・オブジェクト、グループおよびサイト
- レプリケーション環境のタイプ
- レプリケーション環境の管理ツール
- レプリケーションの競合
- マルチマスター・レプリケーションのその他のオプション

注意： Trusted Oracle を使用している環境でレプリケーションを使用する方法は、Oracle セキュリティ関連製品のドキュメントを参照してください。

レプリケーションの概要

レプリケーションとは、分散データベース・システムを構成する複数のデータベースのデータベース・オブジェクト（表など）をコピーおよびメンテナンスするプロセスです。1 つのサイトに適用された変更は、ローカルで獲得および格納された後、リモートの各場所に転送され、適用されます。Oracle レプリケーションは、Oracle Server の完全に統合された機能で、別の独立したサーバーではありません。

レプリケーションは、分散データベース・テクノロジを使用して複数サイト間でデータを共有する機能です。ただし、レプリケート・データベースと分散データベースは同じものではありません。分散データベースの場合は、多くの場所からデータを使用できますが、特定の表が存在するのは 1 つの場所のみです。たとえば、DB1、DB2 および DB3 データベースで構成される分散データベース・システムでは、EMP 表は DB1 のみに存在しています。レプリケーションでは、複数の場所で同一のデータを使用できます。たとえば、DB1、DB2 および DB3 のどのデータベースでも EMP 表を使用できます。

レプリケーションを使用するのは、次のような理由によります。

可用性

レプリケーションを使用すると、様々な方法でデータにアクセスできるので、アプリケーションの可用性が向上します。1 つのサイトが使用できなくなった場合でも、それ以外の場所に対して問合せや更新を続行できます。つまり、レプリケーションを使用することで、フェイルオーバーによる完全な保護を確立できます。

パフォーマンス

レプリケーションを行うと、複数サイト間でアクティビティを均衡化できるので、共有データに高速にローカル・アクセスできます。各ユーザーは同時に別々のサーバーにアクセスでき、その結果全サーバーの負荷が軽減されます。また、ユーザーはアクセス・コストが最も低いレプリケーション・サイト（通常、ユーザーから地理的に最も近いサーバー）のデータを利用できます。

モバイル・コンピューティング

スナップショットは、ある一時点におけるターゲット・マスター表の完全または一部のコピー（レプリカ）です。スナップショットを使用すると、ユーザーは中央のデータベース・サーバーに接続せずにデータベースのサブセットを操作できます。その後、中央のデータベース・サーバーとの接続が確立された時点で、必要に応じてスナップショットを同期化（リフレッシュ）できます。ユーザーがスナップショットをリフレッシュすると、スナップショット・サイトのすべての変更が中央のデータベースに反映されます。また、ユーザーが接続されていない間に中央のデータベースで発生した変更が、スナップショットに反映されます。

ネットワーク負荷の軽減

レプリケーションを使用すると、複数の場所にデータを分散できます。したがって、アプリケーションは中央の 1 つのサーバーではなく様々な地域サーバーにアクセスできます。このように構成することで、ネットワーク負荷は大幅に軽減されます。

大量配布

組織では、データの使用や操作の能力が高いアプリケーションを多数配置する必要性が高まってきています。Oracle レプリケーションでは、配置テンプレートを使用することで、複数のスナップショット環境を迅速に作成できます。個々のニーズにあわせてスナップショット環境をカスタマイズするには、変数を使用します。たとえば、営業支援システムの配置テンプレートを使用する場合は、テンプレートに様々な販売地域や営業担当員用の変数を含めることができます。

レプリケーションの使用方法の詳細は、以降の章で説明します。

レプリケーションを使用するアプリケーション

レプリケーションは、要件が異なる様々なアプリケーションをサポートしています。アプリケーションの中には、比較的自律的な個別のスナップショット・サイトに適しているものもあります。たとえば、営業支援システム、フィールド・サービス、小売りなどの大量配布アプリケーションでは、通常、中央のデータベース・システムと、中央のデータベースと非接続状態にあることが多い多数の小規模リモート・サイトとの間で定期的にデータを同期化する必要があります。営業担当員は、中央のデータベースに接続されているかどうかに関係なく、トランザクションを完了できる必要があります。この場合、リモート・サイトは自律的である必要があります。

これに対し、コール・センターやインターネット・システムなどのアプリケーションでは、複数のサーバー上のデータを継続的かつほぼ即時に同期化し、常に同じ内容のサービスを利用できる必要があります。たとえば、インターネット上の小売り Web サイトでは、顧客が閲覧する情報がどのサイトのオンライン・カタログでも同じであることが必要です。この場合、サイトの自律性よりもデータの整合性が重要です。

Oracle レプリケーションは、どちらのタイプのアプリケーションでも使用できます。また、両方の側面を併せ持ったアプリケーションにも対応できます。実際、1 つの Oracle レプリケーション環境で、大量配布レプリケーションとサーバー / サーバー・レプリケーションの両方をサポートできるので、1 つの一貫した環境に統合することが可能です。このような環境では、たとえば、営業支援システムと顧客サービスのコールセンターの間でデータを共有できます。

レプリケーション・オブジェクト、グループおよびサイト

次の項では、レプリケーション・オブジェクト、レプリケーション・グループおよびレプリケーション・サイトなど、レプリケーション・システムの基本構成要素について説明します。

レプリケーション・オブジェクト

レプリケーション・オブジェクトは、分散データベース・システム内の複数のサーバーに存在するデータベース・オブジェクトです。レプリケーション環境では、1 つのサイトのレプリケーション・オブジェクトに加えられた更新がすべて、その他の各サイトにあるコピーに適用されます。Oracle レプリケーションを使用すると、次のタイプのオブジェクトをレプリケートできます。

- 表
- 索引
- ビュー
- パッケージおよびその本体
- プロシージャおよびファンクション
- トリガー
- 順序
- シノニム

レプリケーション・グループ

レプリケーション環境では、レプリケーション・グループを使用してレプリケーション・オブジェクトを管理します。レプリケーション・グループは、論理的に関連付けられたレプリケーション・オブジェクトの集合です。レプリケーション・グループ内のオブジェクトは、まとめて管理されます。

レプリケーション・グループ内に関連するデータベース・オブジェクトを編成すると、多数のオブジェクトをまとめて簡単に管理できます。通常、レプリケーション・グループを作成および使用して、特定のデータベース・アプリケーションのサポートに必要なスキーマ・オブジェクトを編成します。ただし、レプリケーション・グループおよびスキーマは、相互に対応している必要はありません。レプリケーション・グループには、複数のスキーマのオブジェクトを含めることができます。また 1 つのスキーマには、複数のレプリケーション・グループ内のオブジェクトを含めることができます。ただし、レプリケーション・オブジェクトは、1 つのレプリケーション・グループのメンバーにしかなれません。

レプリケーション・サイト

レプリケーション・グループは、複数のレプリケーション・サイトに存在できます。レプリケーション環境では、マスター・サイトおよびスナップショット・サイトの 2 つの基本的な

タイプのサイトをサポートしています。1 つのサイトが同時にマスター・サイトとスナップショット・サイトになることが可能です。

マスター・サイトとスナップショット・サイトには次のような違いがあります。

- マスター・サイトのレプリケーション・グループは、マスター・グループと呼ばれます。スナップショット・サイトのレプリケーション・グループは、スナップショット・グループと呼ばれます。また、すべてのマスター・グループには、マスター定義サイトが 1 つずつ存在します。レプリケーション・グループのマスター定義サイトは、レプリケーション・グループおよびグループ内のオブジェクトを管理する制御センターとして機能するマスター・サイトです。
- マスター・サイトでは、レプリケーション・グループ内のすべてのオブジェクトの完全なコピーがメンテナンスされます。一方、スナップショット・サイトのスナップショットには、マスター・グループ内のすべての表データまたは表データのサブセットを含めることができます。たとえば、SCOTT_MG マスター・グループに EMP 表と DEPT 表が含まれている場合、すべてのマスター・サイトに EMP 表および DEPT 表の完全なコピーがメンテナンスされている必要があります。これに対して、あるスナップショット・サイトには EMP 表のスナップショットのみを含め、別のスナップショット・サイトには EMP 表と DEPT 表の両方のスナップショットを含めることが可能です。
- マルチマスター・レプリケーション環境のすべてのマスター・サイトは、相互に直接通信を行って、レプリケーション・グループのデータおよびスキーマの変更を絶えず伝播します。スナップショット・サイトには、ある一時点における表データのイメージ、すなわちスナップショットが含まれています。通常、スナップショットは、定期的にリフレッシュされてマスター・サイトと同期化されます。スナップショットはリフレッシュ・グループに編成できます。リフレッシュ・グループ内のスナップショットは、1 つ以上のスナップショット・グループに含めることができます。各スナップショットは、リフレッシュ・グループ内のすべてのスナップショットのデータが、トランザクションの一貫性を保つ特定の時点と一致するように、同時にリフレッシュされます。

レプリケーション環境のタイプ

Oracle レプリケーションは、次のタイプのレプリケーション環境をサポートしています。

- マルチマスター・レプリケーション
- スナップショット・レプリケーション
- マルチマスターおよびスナップショットのハイブリッド構成

マルチマスター・レプリケーション

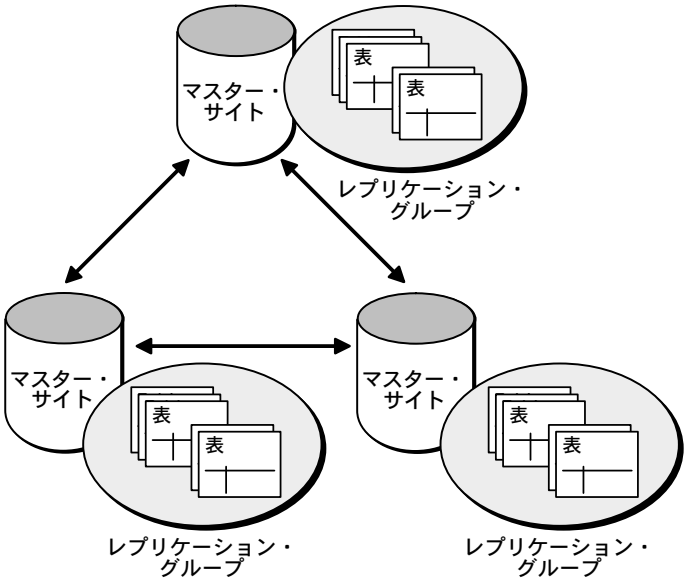
マルチマスター・レプリケーション（peer-to-peer レプリケーションまたは n-way レプリケーションとも呼ばれます）を使用すると、同等ピアとして機能する複数サイトでレプリケート・データベース・オブジェクトのグループを管理できます。マルチマスター・レプリケーション環境では、各サイトはマスター・サイトになります。

アプリケーションは、マルチマスター構成のサイトにある任意のレプリケート表を更新できます。マルチマスター環境でマスター・サイトとして機能している Oracle データベース・サーバーで、すべての表レプリカのデータが自動的に収集され、グローバルなトランザクション一貫性およびデータ整合性が保証されます。

マルチマスター・レプリケーションをインプリメントする最も一般的な方法は、非同期レプリケーションです。その他の方法としては、同期レプリケーションおよびプロシージャ・レプリケーションがあります。これらについてはこの章で後述します。非同期レプリケーションを使用する場合、表の更新は変更が発生したマスター・サイトの遅延トランザクションキューに格納されます。このような変更は、遅延トランザクションと呼ばれます。遅延トランザクションは、関係する他のマスター・サイトに一定の間隔でプッシュ（または伝播）されます。伝播間隔は、変更できます。

同じ行の値が 2 つの別のマスター・サイトでほぼ同時に更新される場合があるので、非同期レプリケーションを使用すると競合が発生する可能性があります。ただし、競合は技術によって回避することができます。競合が発生した場合は、Oracle が提供する組込みメカニズムにより問題を解決できます。

図 1-1 マルチマスター・レプリケーション



マスター・グループの停止

マスター・グループで管理作業を実行するために、マスター・グループのレプリケーション・アクティビティをすべて停止することが必要な場合があります。たとえば、グループ内の任意の表に対してデータ定義言語（DDL）文を発行する場合などです。マスター・グループのレプリケーション・アクティビティをすべて停止することを、グループの停止といいます。マスター・グループの停止中は、マスター・グループ内のオブジェクトに対してデータ操作言語（DML）文を実行できません。

スナップショット・レプリケーション

スナップショットには、ある一時点におけるターゲット・マスター表の完全または一部のコピーが含まれます。スナップショットには、読取り専用スナップショットと更新可能スナップショットがあります。

スナップショットには、次の利点があります。

- ローカル・アクセスが可能なため、応答時間を短縮し、可用性を向上できます。
- ローカル・スナップショットへの問合せが可能なので、マスター・サイトからの問合せをオフロードできます。
- ターゲット・マスター表のデータ・セットのうち、選択したサブセットのみをレプリケート可能にすることにより、データのセキュリティを高めることができます。

読取り専用スナップショット

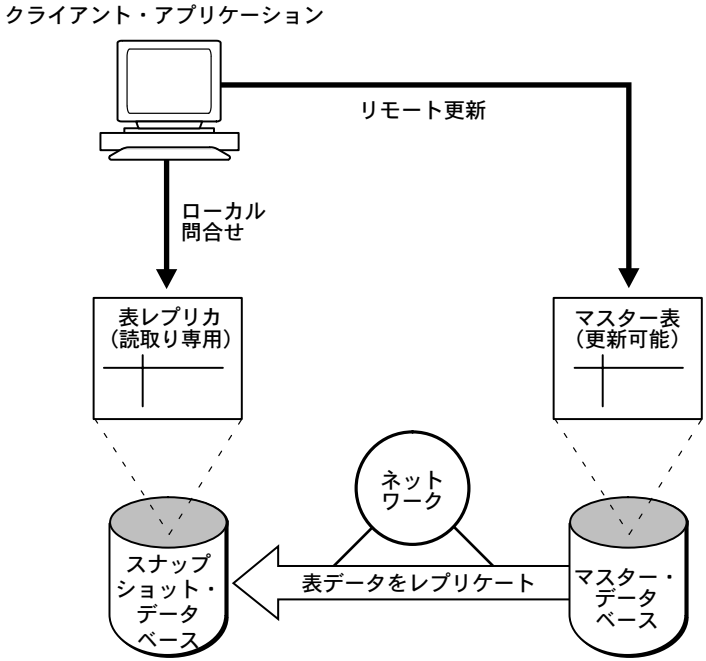
基本構成では、スナップショットを使用して、マスター・サイトのデータをベースにして作成された表データに読取り専用でアクセスできます。アプリケーションは、ネットワークが使用可能かどうかにかかわらず、ネットワークにアクセスせずに読取り専用スナップショットのデータを問い合せることができます。ただし、システム全体で使用するアプリケーションの場合、更新を行うにはマスター・サイトのデータにアクセスする必要があります。図 1-2 に、基本的な読取り専用レプリケーションを示します。読取り専用スナップショットのマスター表は、マスター・グループに属している必要はありません。

読取り専用スナップショットには、次の利点があります。

- 更新できないため、競合の可能性はありません。
- 複合スナップショットをサポートしています。複合スナップショットの例として、集合演算または CONNECT BY 句を含むスナップショットがあります。

> **関連項目：** 複合スナップショットの詳細は、3-4 ページの「使用可能なスナップショット」を参照してください。

図 1-2 読取り専用スナップショット・レプリケーション

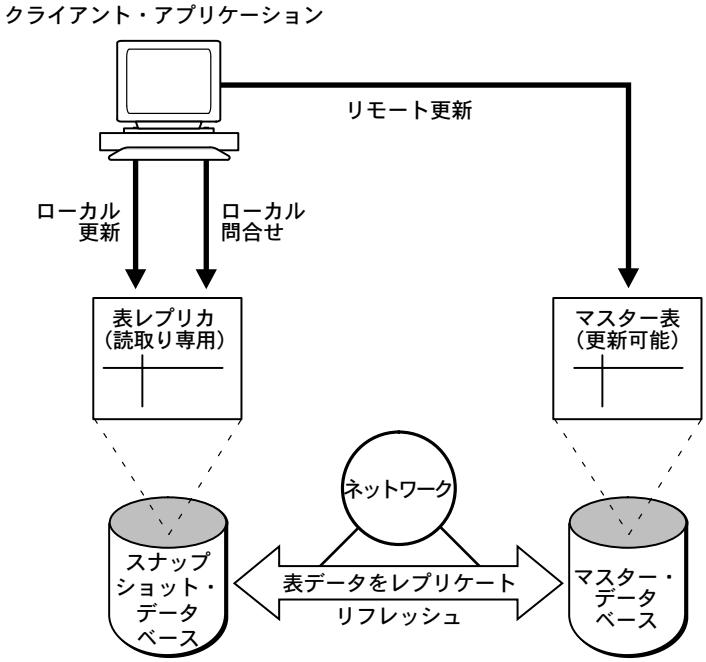


更新可能スナップショット

より高度な構成では、更新可能スナップショットを作成できます。更新可能スナップショットを使用すると、ユーザーはスナップショットに対して、挿入、更新および削除の操作を実行することで、ターゲット・マスター表の行を挿入、更新および削除できます。また、更新可能スナップショットには、ターゲット・マスター表のデータの 1 つのサブセットのみを含めることもできます。図 1-3 に、更新可能スナップショットを使用したレプリケーション環境を示します。

更新可能スナップショットは、レプリケーションをサポートするように設定された、マスター・サイトの表をベースにしています。実際、更新可能スナップショットは、マスター・サイトのマスター・グループをベースとするスナップショット・グループの一部であることが必要です。

図 1-3 更新可能スナップショット・レプリケーション



更新可能スナップショットには、次の特性があります。

- 常に 1 つの表をベースとしています。
- 増分リフレッシュ（または高速リフレッシュ）を実行できます。
- 更新可能スナップショットに対して加えられた変更は、スナップショットのリモート・マスター表に伝播されます。マスター表に加えられた更新は必要に応じて、その他のすべてのマスター・サイトに順に伝播されます。
- 読取り専用スナップショットをリフレッシュする場合と同様に、リフレッシュ・グループの一部としてリフレッシュできます。

更新可能スナップショットには、次の利点があります。

- マスター・サイトに接続されていない状態でも、ローカルのレプリケート・データ・セットに対して問合せおよび更新ができます。
- データの更新をサポートしていますが、リソースはマルチマスター・レプリケーションの場合よりも少なくて済みます。たとえば、スナップショットを Oracle8*i* Lite データベース上に常駐させることができるので、スナップショット・クライアントが必要とするディスク領域や必要なメモリー量は、Oracle8*i* Server の場合よりも大幅に少なくて済みます。

スナップショットのリフレッシュ

スナップショットとマスター表との整合性を保つには、スナップショットを定期的にリフレッシュする必要があります。スナップショットをリフレッシュするには、次の 3 つの方法があります。

- スナップショット・ログを使用して、最後のリフレッシュ以降に変更が加えられた行のみを更新する高速リフレッシュ。
- スナップショット全体を更新する完全リフレッシュ。
- 可能なときに高速リフレッシュを実行する強制リフレッシュ。高速リフレッシュが実行できない場合は、完全リフレッシュが実行されます。

スナップショット間のトランザクションの一貫性を保つことが重要なときは、スナップショットをリフレッシュ・グループに編成できます。リフレッシュ・グループをリフレッシュすることで、リフレッシュ・グループ内のすべてのスナップショットのデータが、トランザクションの一貫性を保つ特定の時点のデータと確実に一致します。リフレッシュ・グループ内のスナップショットを個々にリフレッシュすることも可能です。ただし、リフレッシュ・グループ内のその他のスナップショットはリフレッシュされないので、リフレッシュ・グループを編成する意味はなくなります。

スナップショット・ログ

スナップショット・ログは、DML によってマスター表に加えられたすべての変更を記録する表です。スナップショット・ログは 1 つのマスター表と対応付けられていて、各マスター表には、マスター表からリフレッシュされるスナップショットの数に関係なく、1 つのスナップショット・ログがあります。スナップショットの高速リフレッシュは、スナップショットのマスター表にスナップショット・ログがある場合にのみ可能です。スナップショットを高速リフレッシュすると、スナップショットと対応付けられたスナップショット・ログのエントリで、最後のリフレッシュ以降に追加されたエントリがスナップショットに適用されます。

配置テンプレート

配置テンプレートを使用すると、多くのリモート・スナップショット・サイトの配置およびメンテナンスの作業が簡単になります。また、配置テンプレートを使用すると、マスター・サイトでスナップショット定義の集合を作成でき、その定義中のパラメータを使用してスナップショットを個々のユーザーまたはユーザーのタイプに合わせてカスタマイズできます。

たとえば、営業用にテンプレートを作成し、技術員用に別のテンプレートを作成する場合などです。この場合、販売地域または顧客サポート・レベルといったパラメータを使用できます。リモート・ユーザーは、マスター・サイトに接続するときに、使用可能なテンプレートのリストを問い合せることができます。ユーザーがテンプレートをインスタンシエートすると、リモート・サイトで適切なスナップショットが作成され、データが移入されます。適切なパラメータ値は、リモート・ユーザーが指定するか、またはマスター・サイトに保持された表の値を利用します。

オンライン・インスタンシエーションおよびオフライン・インスタンシエーション ユーザーがスナップショット・サイトでテンプレートをインスタンシエートすると、オブジェクト DDL（たとえば、CREATE SNAPSHOT... や CREATE TABLE...）が実行されて、スナップショット・サイトに適切なスキーマ・オブジェクトが作成されます。そして、そのオブジェクトに適切なデータが移入されます。

テンプレートのインスタンシエーションは、マスター・サイトに接続した状態でネットワークを介して行うことも（オンライン・インスタンシエーション）、あるいはマスター・サイトから切断された状態で行うことも（オフライン・インスタンシエーション）可能です。

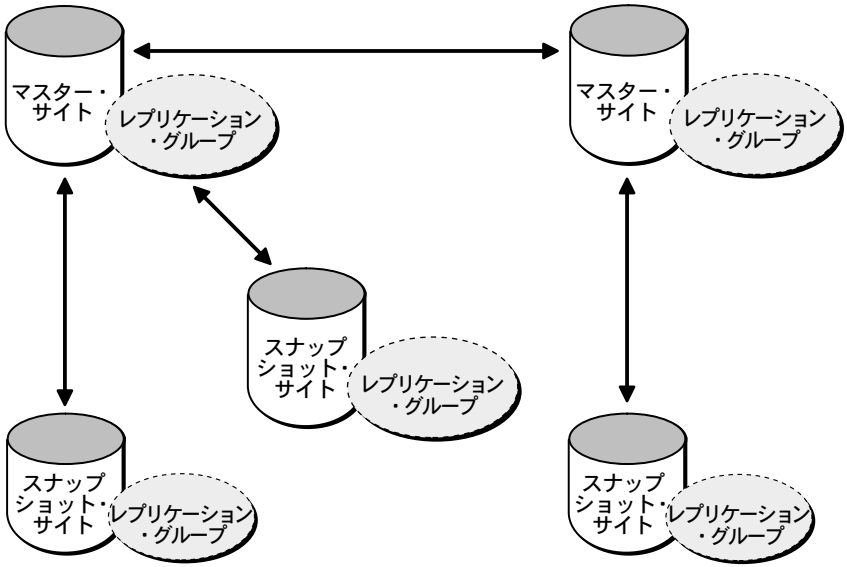
オフライン・インスタンシエーションは、ピーク期間中のサーバー負荷を軽減したり、リモート接続時間を短縮するために使用されます。テンプレートをオフラインでインスタンシエートするには、テンプレートおよび必要なデータをテープや CD-ROM などの記憶メディアにパッケージ化します。そして、マスター・サイトではなく、テンプレートとデータが入った記憶メディアからデータを取り出します。

マルチマスターおよびスナップショットのハイブリッド構成

マルチマスター・レプリケーションとスナップショットは、様々なアプリケーションの要件に合うようにハイブリッド（複合）構成で結合できます。複合された構成では、任意の数のマスター・サイトと、各マスター用の複数のスナップショット・サイトを作成できます。

たとえば、図 1-4 に示すように、2 つのマスター間のマルチマスター（または n-way）レプリケーションでは、2 つの地域をサポートするデータベース間の全表レプリケーションをサポートできます。スナップショットをマスター上に定義すると、全表または表のサブセットを各地域内のサイトにレプリケートできます。

図 1-4 ハイブリッド構成



スナップショットとレプリケート・マスターとの主な違いは次のとおりです。

- レプリケート・マスターにはレプリケートされている表全体のデータが含まれている必要がありますが、スナップショットはマスター表のデータのサブセットをレプリケートできます。
- マルチマスター・レプリケーションを使用すると、変更が発生するたびに各トランザクションの変更内容をレプリケートできます。スナップショット・リフレッシュを使用すると、一定の間隔で処理されるので、マルチ・マスター・レプリケーションより少ない頻度で複数のトランザクションの内容をより効率的に伝播できます。
- マスター・サイトでは、同じデータの複数のコピーに対して加えられた変更によって生じた競合を検出し、解消します。

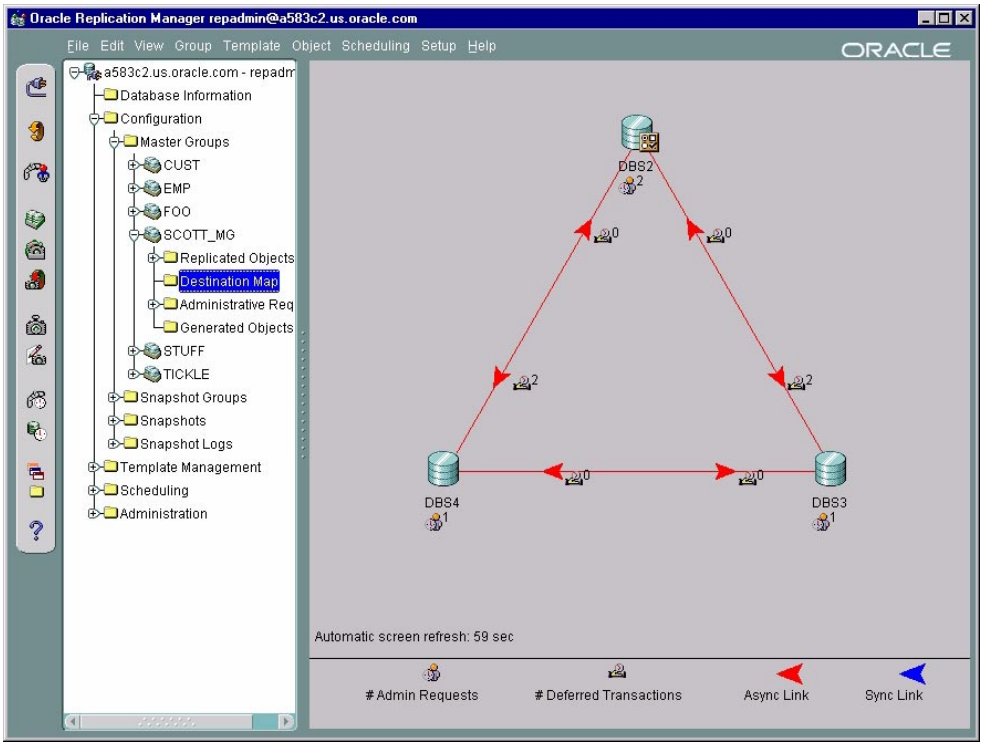
レプリケーション環境の管理ツール

レプリケーション環境を管理および監視するために、いくつかのツールを使用できます。Oracle Replication Manager では、レプリケーション環境の管理に役立つ強力な GUI インタフェースが提供されます。また、レプリケーション・マネージメント API では、使い慣れたアプリケーション・プログラミング・インタフェース（API）を使用してレプリケーション管理のためのカスタマイズ可能なスクリプトを作成できます。レプリケーション・カタログを使用すると、レプリケート環境に関する情報を管理することもできます。

Oracle Replication Manager

マルチマスター・レプリケーションおよびスナップショット・レプリケーションの両方をサポートしているレプリケーション環境では、構成や管理作業が複雑になります。このようなレプリケーション環境の管理に利用する高度な管理ツール、Oracle Replication Manager が用意されています。Replication Manager の使用方法および例は、このマニュアルの他の項で説明しています。Replication Manager の詳細は、Replication Manager のオンライン・ヘルプを参照してください。

図 1-5 Replication Manager



関連項目： Replication Manager の概説は、第 8 章「Replication Manager の概要」を参照してください。Replication Manager の使用方法の詳細は、Replication Manager のオンライン・ヘルプを参照してください。

レプリケーション・マネージメント API

レプリケーション・マネージメント・アプリケーション・プログラミング・インタフェース（API）は、Oracle レプリケーション環境を構成する際に使用するプロシージャおよびファンクションをカプセル化した PL/SQL パッケージのセットです。レプリケーション・マネージメント API は、Replication Manager の機能をコマンド・ラインで実行するものです。つまり、Replication Manager は、レプリケーション・マネージメント API のプロシージャとファンクションを使用して処理を実行します。たとえば、Replication Manager を使用して新しいマスター・グループを作成する場合、Replication Manager は、DBMS_REPCAT.CREATE_MASTER_REPGROUP プロシージャをコールして作業を完了させます。レプリケーション・マネージメント API を使用すると、レプリケーション環境を管理するカスタム・スクリプトを簡単に作成できます。

関連項目： レプリケーション・マネージメント API の使用方法の詳細は、『Oracle8*i* レプリケーション・マネージメント API リファレンス』を参照してください。

レプリケーション・カタログ

レプリケーション環境の各マスター・サイトおよびスナップショット・サイトには、レプリケーション・カタログがあります。サイトのレプリケーション・カタログは、そのサイトのレプリケーション・オブジェクトとレプリケーション・グループに関する管理情報をメンテナンスしている、データ・ディクショナリの表およびビューの固有のセットです。レプリケーション環境に関係する各サーバーでは、そのレプリケーション・カタログの情報を使用して、レプリケーション・グループのオブジェクトのレプリケーションを自動化できます。

関連項目： レプリケーション・カタログの詳細は、『Oracle8*i* レプリケーション・マネージメント API リファレンス』を参照してください。

分散スキーマ管理

レプリケートされたオブジェクトに対しては、Replication Manager または DBMS_REPCAT パッケージのいずれかを使用して DDL 文を発行する必要があります。いずれかのインタフェースを使用すると、レプリケーション環境に関係するすべてのサイトにすべての DDL 文がレプリケートされます。

注意： SQL*Plus などのツールを使用して直接 DDL 文を発行する場合は、その他のサイトにレプリケートされません。

レプリケーションの競合

非同期マルチマスター・レプリケーション環境および更新可能スナップショット・レプリケーション環境では、異なるサイトから発行された 2 つのトランザクションがほぼ同時に同じ行を更新した場合などに発生するレプリケーション競合に対処する必要があります。データ競合が発生した場合は、ビジネス・ルールに従って競合を解消し、すべてのサイトを適正に収束するためのメカニズムが必要です。

Oracle レプリケーションでは、レプリケート環境で発生したすべての競合がロギングされるのみでなく、様々な組込みの競合解消方法が用意されています。これらの方法を使用してデータベースの競合解消システムを定義すると、ビジネス・ルールに従って競合を解消できます。Oracle の組込みの競合解消方法によって競合を解消できない場合は、独自の競合ルーチンを作成し、それを使用することもできます。

関連項目：

- データ競合が発生しないようにデータベースを設計する方法、および競合が発生した場合に使用する競合解消ルーチンを作成する方法の詳細は、第 5 章「競合解消の概念およびアーキテクチャ」を参照してください。
- Replication Manager を使用して競合解消方法を作成する方法の詳細は、Replication Manager のオンライン・ヘルプを参照してください。
- レプリケーション・マネージメント API を使用して競合解消ルーチンを作成する方法の詳細は、『Oracle8*i* レプリケーション・マネージメント API リファレンス』の第 6 章「競合解消」を参照してください。

マルチマスター・レプリケーションのその他のオプション

マルチマスター・レプリケーションをインプリメントする最も一般的な方法は非同期レプリケーションです。ただし、他に同期レプリケーションおよびプロシージャ・レプリケーションという 2 つの方法もあります。

同期レプリケーション

マルチマスター・レプリケーション環境では、非同期レプリケーションまたは同期レプリケーションを使用してデータをコピーできます。非同期レプリケーションを使用する場合、1 つのマスター・サイトに加えられた変更は、一定時間後に関係するすべてのマスター・サイトに反映されます。同期レプリケーションの場合は、1 つのマスター・サイトに加えられた変更は即時に、関係するすべてのマスター・サイトに反映されます。

同期レプリケーションを使用して表を更新すると、関係するすべてのマスター・サイトで更新内容が即時にレプリケートされます。つまり、それぞれのトランザクションはすべてのマスター・サイトと関連があります。したがって、なんらかの理由から 1 つのマスター・サイトでトランザクションを処理できなかった場合、そのトランザクションはすべてのマスター・サイトでロールバックされます。

同期レプリケーションを使用して、競合の発生を回避できますが、処理を円滑に行うには非常に安定した環境が必要です。たとえば、ネットワーク上の問題により 1 つのマスター・サイトと通信できない場合、通信が再び確立されるまでどのトランザクションも完了しません。

プロシージャ・レプリケーション

バッチ処理アプリケーションでは、1 つのトランザクション内で大量のデータを変更できます。この場合、通常の行レベル・レプリケーションを実行すると、大量のデータ変更のためにネットワークの負荷が増大する可能性があります。このような問題を回避するために、レプリケーション環境で動作するバッチ処理アプリケーションでは、プロシージャ・レプリケーションを使用して、データ・レプリカを収集する簡単なストアド・プロシージャ・コールをレプリケートできます。プロシージャ・レプリケーションでは、アプリケーションが表の更新に使用するストアド・プロシージャのコールのみがレプリケートされます。データの変更自体はレプリケートされません。

プロシージャ・レプリケーションを使用するには、システムのデータを変更するパッケージをすべてのサイトにレプリケートする必要があります。パッケージをレプリケートした後に、各サイトでこのパッケージ用のラッパーを生成する必要があります。アプリケーションがデータを変更するパッケージ・プロシージャをローカル・サイトでコールすると、ラッパーは最終的にレプリケート環境内の他のすべてのサイトで同じパッケージ・プロシージャをコールします。プロシージャ・レプリケーションは、非同期または同期で実行できます。

競合の検出およびプロシージャ・レプリケーション

レプリケーション・システムでプロシージャ・レプリケーションを使用してデータをレプリケートするときは、データをレプリケートするプロシージャで、レプリケート・データの整合性を保証できる必要があります。つまり、このようなプロシージャは、レプリケーションの競合を回避または検出し、競合を適切に解消するように設計する必要があります。したがって、プロシージャ・レプリケーションは、主に大規模バッチ操作によってのみデータベースを変更する場合に使用されます。このような状況では、多数のトランザクションが同じデータで競合することはないため、レプリケーションの競合はほとんど発生しません。

関連項目： 詳細は、6-2 ページの「プロシージャ・レプリケーションの使用方法」を参照してください。

2

マスター・レプリケーションの概念およびアーキテクチャ

この章では、シングル・マスター・レプリケーション環境およびマルチマスター・レプリケーション環境での、Oracle のマスター・レプリケーション・サイトの概念およびアーキテクチャ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。

- マスター・レプリケーションの概念
- マスター・レプリケーションのアーキテクチャ

マスター・レプリケーションの概念

マスター・レプリケーションのアーキテクチャの詳細を理解するには、マスター・レプリケーションの概念について知っておく必要があります。マスター・レプリケーションの使用方法と目的がわかると、アーキテクチャの個々の要素がどのように相互に作用してマルチマスター・レプリケーション環境が作成されるのか理解が深まります。

レプリケーションとは

Oracle には、シングル・マスター・レプリケーションおよびマルチマスター・レプリケーションの 2 つの形式のマスター・レプリケーションがあります。

シングル・マスター・レプリケーションは、マスター・サイトおよびマスター・サイトがサポートする 1 つ以上のスナップショット・サイトから構成されます。接続されているすべてのスナップショットは、マスター・サイトに対して更新処理を行います。データ競合の検出および解消はマスター・サイトで実行されます。

> **関連項目：** スナップショットの詳細は、第 3 章「スナップショットの概念およびアーキテクチャ」を参照してください。

マルチマスター・レプリケーション（peer-to-peer レプリケーションまたは n-way レプリケーションとも呼ばれます）を使用すると、同等ピアとして機能する複数サイトでレプリケート・データベース・オブジェクトのグループを管理できます。アプリケーションは、マルチマスター構成のサイトにある任意のレプリケート表を更新できます。図 2-1 は、マルチマスター・レプリケーション・システムを示します。

マルチマスター・レプリケーション環境でマスター・サイトとして機能している Oracle データベース・サーバーで、すべての表レプリカのデータが自動的に収束され、グローバルなトランザクション一貫性とデータ整合性が保証されます。

マスター・サイト

マスター・サイトは、マルチマスター・レプリケーション環境のノードまたはシングル・マスター・レプリケーション環境の 1 つ以上のスナップショット・サイトのマスター（あるいはその両方）として設定できます。レプリケート・オブジェクトはマスター・サイトに格納され、ユーザーはアクセスできます。

マスター定義サイト マルチマスター・レプリケーション環境では、1 つのマスター・サイトがマスター・グループのマスター定義サイトとして機能します。この特定のサイトで、マルチマスター・レプリケーション環境の管理作業やメンテナンス作業の多くが実行されます。

マスター定義サイトはマルチマスター環境のどのマスター・サイトでもかまいませんが、マスター・グループ内に設定できるのは 1 つのみです。また、必要に応じて、マスター定義サイトを別のマスター・サイトに変更することもできます。

デフォルトでは、スナップショット・レプリケーションをサポートする 1 つのマスター・サイトがマスター定義サイトになります。

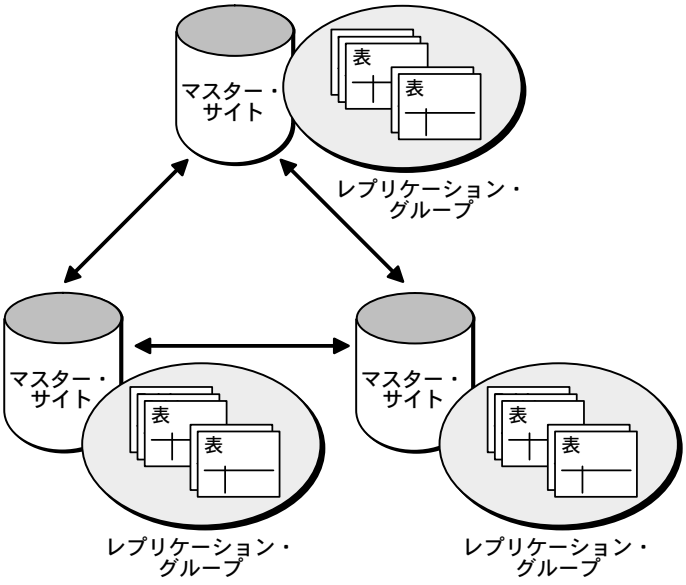
マルチマスター・レプリケーションの使用目的

基本的に、レプリケーションを使用するのは、必要なときに必要な場所で情報（データ）を確実に使用するためです。次の項では、情報の配布要件が異なるいくつかの環境について説明します。使用しているレプリケーション環境でも、次の要件を 1 つ以上備えています。

フェイルオーバー

マルチマスター・レプリケーションは、ミッション・クリティカルなデータベースの可用性を保護するのに役立ちます。たとえば、マルチマスター・レプリケーション環境では、システムまたはネットワーク障害が発生してプライマリ・サイトが使用できなくなった場合に備え、データベース内のデータ全体をレプリケートしてフェイルオーバー・サイトを確立できます。このフェイルオーバー・サイトは、Oracle のスタンバイ・データベース機能と異なり、プライマリ・サイトが同時に稼動しているときでも、アプリケーションからアクセスできる完全に機能したデータベースとして機能します。これに対して、スタンバイ・データベースが完全に機能するのは、プライマリ・サイトが使用できない場合のみです。

図 2-1 マルチマスター・レプリケーション



Net8 を使用すると、自動的に接続時フェイルオーバーを設定できます。この機能により、最初のマスター・サイトに障害が発生した場合、Net8 は別のマスター・サイトに接続を切り替えることができます。自動的に接続時フェイルオーバーを設定するには、`tnsnames.ora` ファイルで、FAILOVER オプションを ON に設定し、複数の接続記述子を指定します。

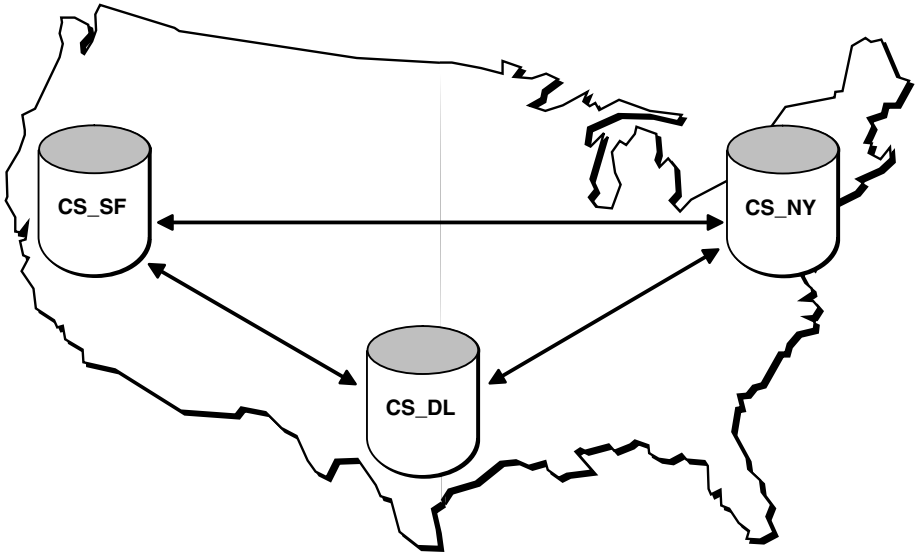
> **関連項目：** 接続時フェイルオーバーの設定の詳細は、『Oracle8*i* Net8 管理者ガイド』を参照してください。

ロード・バランス

マルチマスター・レプリケーションは、過度のアプリケーション負荷の分配、継続的な可用性の保証、またはよりローカライズされたデータ・アクセスの提供を目的として複数ポイントのデータベース情報にアクセスすることが必要なトランザクション処理アプリケーションに便利です。

アプリケーション負荷を分散することが必要なアプリケーションとして一般的なものに、サービス指向のアプリケーションがあります。

図 2-2 複数ポイントの更新アクセスをサポートするマルチマスター・レプリケーション



Parallel Server とレプリケーション

Oracle レプリケーションと Oracle Parallel Server のどちらを使用するかを考える場合は、ロード・バランスおよび耐障害性の 2 つの観点から検討します。

ロード・バランス： Oracle Parallel Server では、複数のインスタンスに対する読込みと書込みの負荷を均衡化します。Oracle レプリケーションでは、複数のデータベースに対する読込みロード・バランスを提供します。ただし、書込みロードバランス全体の効果は、それぞれの書込みが各マスター・サイトで実行されることにより提供されます。

耐障害性： レプリケーションの場合、各レプリケーション・サイトが地理的に異なる地域に置かれている可能性があるので、自然災害、停電またはサボタージュに対する耐障害性は高くなります。Oracle Parallel Server は、クラスタまたはその他の超並列システム上で動作し、同一の物理環境に存在しています。したがって、レプリケーションの使用で対処できる物理的問題に対する耐障害性はありません。

非接続スナップショット環境のサポート

スナップショット・レプリケーションを使用すると、ユーザーは、マスター・サイトに接続せずに、マスター・サイトからレプリケート・データの全部またはサブセットをリモートで格納できます。たとえば、営業支援システムでは、個人のラップトップ（非接続デバイス）に個々の営業担当員に関連するデータのサブセットを格納します。

マスター・サイトは、スナップショット環境のターゲットとして機能します。マスター・サイトのサポートには、次のものがあります。

- 全スナップショットをサポートする 1 つのマスター・サイト。競合はすべて 1 つのマスター・サイトで解消され、データの不一致は発生しません。
- マルチマスター・レプリケーションとスナップショット・レプリケーションの組合せ。この組合せでは、マルチマスター構成の様々なマスターがスナップショット・グループのターゲットになります。この構成を使用すると、ネットワーク負荷が複数のマスター・ノードに分散され、いずれかのマスター・ノードが使用できなくなった場合に備えてスケーラビリティと可用性を向上できます。

マスター・レプリケーションのタイプ

マスター・レプリケーションには、マルチマスター・レプリケーションとシングル・マスター・レプリケーションの 2 つのタイプのレプリケーションがあります。マルチマスター・レプリケーションは、複数のマスター・サイトを含んでおり、各マスター・サイトが同等ピアとして機能します。シングル・マスター・レプリケーションでは、スナップショット・レプリケーションをサポートする 1 つのマスター・サイトが、数百、数千のスナップショット・サイトをサポートします。また、1 つ以上のスナップショット・サイトをサポートする 1 つのマスター・サイトが、複数マスター・サイト環境の一部として、ハイブリッド・レプリケーション環境（マルチマスター・レプリケーションとスナップショット・レプリケーションの組合せ）を構成することも可能です。

マルチマスター・レプリケーション

マルチマスター・レプリケーションは同等な立場の複数のマスター・サイトから構成されており、どのマスター・サイトでも更新が可能です。個々のマスター・サイトの更新処理は、関係するすべてのマスター・サイトに伝播されます。

競合解消は、各マスター・サイトで独自に行われます。マルチマスター・レプリケーションでは、各マスター・サイトのレプリケート表の完全なレプリカが提供されます。

シングル・マスター・レプリケーション

1 つのマスター・サイトを、1 つ以上のスナップショット・サイトのターゲット・マスター・サイトとして機能させることもできます。マルチマスター・レプリケーションでは、1 つのサイトに対して行った更新処理はその他のすべてのマスター・サイトに伝播されましたが、スナップショットでは、更新はそのターゲット・マスター・サイトにのみ反映されます。ハイブリッド・レプリケーション環境（複数のマスター・サイトと各マスター・サイトがサポートする 1 つ以上のスナップショット・サイトから構成される環境）の場合、ターゲット・マスター・サイトから複数サイトのレプリケーション環境の各マスター・サイトにスナップショットのすべての更新内容が伝播されます。

競合解消は、マスター・サイトでのみ行います。スナップショット・レプリケーションには、レプリケート表の全部または一部のレプリカを含めることができます。

> **関連項目：** 1 つのマスター・サイトを使用するスナップショット・レプリケーションの詳細は、第 3 章「スナップショットの概念およびアーキテクチャ」を参照してください。

マルチマスター・レプリケーション・プロセス

マルチマスター・レプリケーションには、非同期および同期の 2 つのタイプがあります。

非同期レプリケーション（蓄積交換レプリケーションとも呼ばれます）では、ローカルで加えられたすべての変更を獲得してキューに格納し、一定の間隔でリモート・サイトに伝播して変更を適用します。この形式のレプリケーションでは、すべてのサイトでデータが収束するまでに一定の時間がかかります。

同期レプリケーション（リアルタイム・レプリケーションとも呼ばれます）では、1 つのトランザクションの一部として、レプリケーション環境に関係するすべてのサイトで変更を適用し、レプリケート・プロシージャを実行します。いずれかのサイトで DML またはプロシージャが失敗すると、トランザクション全体がロールバックされます。同期レプリケーションを使用すると、各サイトのデータ整合性がリアルタイムで保証されます。

マスター・サイトの伝播モードは、非同期から同期に、または同期から非同期に変更できます。マスター・グループ内のマスター・サイトの伝播モードを変更した場合、すべてのマスター・グループ・オブジェクトに対してレプリケーション・サポートを再度生成する必要があります。また、マルチマスター・レプリケーション環境では、同期レプリケーションおよび非同期レプリケーションの両方を混在させることも可能です。

関連項目： 詳細は、2-33 ページの「複合モードのマルチマスター・システムについて」を参照してください。

非同期レプリケーション

非同期レプリケーションでは、DML またはレプリケート・プロシージャを実行すると、マルチマスター・レプリケーション環境のすべてのマスター・サイトに個別に伝播（送信）されます。各トランザクションにおいてデータが伝播されるのは、DML またはレプリケーション・プロシージャをローカルで実行した後です。

非同期レプリケーションがデフォルトのレプリケーション・モードです。非同期レプリケーションでは、同期レプリケーションの場合よりもネットワーク・リソースやハードウェア・リソースの使用を抑えることができるため、可用性およびパフォーマンスが高くなります。

ただし、非同期レプリケーションを使用すると、変更内容が伝播されるまでの一定時間、レプリケーション環境にある別のマスター・サイトのデータ・セットと一致しません。また、非同期レプリケーションでは、データの競合が発生する可能性もあります。

非同期レプリケーション・プロセスは次のとおりです。

- **ユーザーは、DML を発行するか、またはレプリケート・プロシージャのラッパーを実行します。**

  レプリケーション用に設定した表に対してユーザーが発行した DML は、各マスター・サイトへのレプリケーション用にすべて獲得されます。

  挿入、更新または削除の対象となる各行に対して、内部トリガーによって遅延リモート・プロシージャ・コール（RPC）が作成されます。遅延トランザクション・キューには、すべての遅延 RPC が入っています。

  プロシージャがレプリケートされていて、そのラッパーがマスター・サイトで実行される場合、プロシージャ・コールも遅延トランザクション・キューに入れられます。

- **遅延トランザクション・キューに遅延 RPC が格納されます。**

  遅延トランザクション・キュー内の各トランザクションには接続先リストがあり、遅延トランザクションの伝播先が定義されています。このリストには、発行元サイト以外の全マスター・サイトが記載されています。

- **接続先へ遅延トランザクション・キューのエントリを送信します。**

  スケジュールした間隔で、または必要に応じて、遅延トランザクション・キュー内の遅延トランザクションがターゲットの接続先に伝播されます。

- **遅延トランザクション・キューのエントリがリモートの接続先で適用されます。**

  遅延トランザクションがターゲットの接続先へ伝播されると、接続先サイトで各遅延 RPC が適用されます。接続先サイトで遅延トランザクションの適用に失敗した場合、遅延トランザクションは再送されて、接続先サイトのエラー・キューに入れられます。DBA は、このエラー・キューを使用してエラー条件を修正し、遅延トランザクションを再び適用できます。

  遅延トランザクション・キューのエントリがリモートの接続先で適用されるときに、データ競合がないかがチェックされます。データ競合が検出された場合、その競合はロギングされ、オプションで競合解消方法が起動します。

  関連項目： 詳細は、第 5 章「競合解消の概念およびアーキテクチャ」を参照してください。

同期レプリケーション

同期レプリケーションでは、1 つのトランザクションが終了するまでの間に、ローカル・サイトで加えられた変更をレプリケーション環境内の同期リンクしたマスター・サイトに伝播します。いずれかのマスター・サイトで伝播に失敗した場合、ローカルのマスター・サイトで最初に加えられた変更を含め、トランザクション全体がロールバックされます。この厳しい規則により、レプリケーション環境全体のデータ整合性を保つことができます。非同期レプリケーションとは異なり、マスター・サイト間のデータが一致しない時間はありません。

関連項目： 1 つの環境で同期レプリケーションと非同期レプリケーションを混在させる方法の詳細は、2-33 ページの「複合モードのマルチマスター・システムについて」を参照してください。

同期レプリケーションでは、レプリケーション環境でデータ競合が発生しないことも保証されています。このようなメリットがある一方で、ダウンタイムに関する制限が多いハードウェア・リソースやネットワーク・リソースを必要とするというデメリットもあります。たとえば ノードが 6 つあるマルチマスター環境で、1 つのマスター・サイトが使用できなくなった場合、トランザクションは完了しません。これに対して非同期レプリケーションでは、ダウンしたサイトが使用可能になるまで遅延トランザクションが保持されます。

また、問合せはローカルで実行されるため高パフォーマンスを維持できるのに対して、更新処理には時間がかかります。これは、すべての更新内容が正しく伝播され、リモートの接続先サイトへ正しく適用できるように、2 フェーズ・コミット・プロトコルを使用しているためです。

同期レプリケーション・プロセスは次のとおりです。

- **ユーザーは、DML を発行するか、またはレプリケート・プロシージャのラッパーを実行します。**

  レプリケーション用に設定した表に対してユーザーが発行した DML は、各マスター・サイトへのレプリケーション用にすべて獲得されます。

  プロシージャがレプリケートされていて、そのラッパーがマスター・サイトで実行される場合、プロシージャ・コールもレプリケーション用に獲得されます。

- **DML またはラッパーを実行すると即時に接続先のサイトに伝播されます。**

  内部トリガーは、DML を獲得し、これらのアクションを即時にレプリケーション環境の各マスター・サイトに伝播します。そして、これらのアクションをプロパゲータのデータベース・リンクのセキュリティ・コンテキストで適用し、内部 RPC を使用して接続先サイトにこれらのアクションを適用します。

  内部トリガーと同様に、レプリケート・プロシージャのラッパーは、レプリケーション環境の各マスター・サイトにプロシージャ・コールを即時に伝播します。

  いずれかのマスター・レプリケーション・サイトでトランザクションが失敗した場合、すべてのマスター・サイトでトランザクションがロールバックされます。この方法により、すべてのマスター・サイト間でのデータ整合性が保証されます。トランザクションがいずれかのサイトで失敗するとロールバックを実行する必要があるので、同期レプリケーションは、可用性の高いネットワーク、データベースおよびそれに関連するハードウェアに大きく依存します。

競合解消の概念

表がレプリケートされるときに、任意のレプリケーション・サイト（マスター・サイトまたはスナップショット・サイトのいずれか）のレプリケート表に適用される DML のうち、データ競合を引き起こす DML は、マスター・サイトの Oracle Server によって自動的に検出されます。スナップショット・サイトでは、データ競合の検出も解消も行われません。スナップショット・サイトで発生したデータ競合は、スナップショットのターゲット・マスター・サイトで検出され、解消されます。

たとえば、次のマスター・グループが 1 時間に 1 回変更を伝播するようにスケジュールされていると仮定します。

表 2–1

時間	マスター・サイト A	マスター・サイト B	状態
8:00 AM	マスター・サイト B に変更を伝播	マスター・サイト A に変更を伝播	データの収束
8:15 AM	行 1 を更新		
8:30 AM		行 1 を更新	
9:00 AM	マスター・サイト B に変更を伝播	マスター・サイト A に変更を伝播	行 1 で競合検出

2 つのデータ伝播の間の時間を伝播間隔とみなした場合、1 つの間隔中に複数のサイトが同じ行を更新すると競合が発生します。

更新による競合のみでなく、挿入および削除時にも競合が発生します。次のようなタイプの競合について考えてみます。各競合は、同じ間隔内に競合するアクションが起こると発生します。

- 更新競合 : 前の更新内容が各サイトに伝播される前に、同じ行に対して別のサイトで更新が適用される場合。
- 一意性競合 : 複数のサイトで挿入を実行した結果、主キー（またはその他の一意の列のセット）に同じ値が存在する場合、またはあるサイトで更新を実行した結果、主キー（またはその他の一意の列のセット）が変更され、その主キーに別のサイトで挿入した値と同じ値が存在する場合。
- 削除競合 : あるサイトで行が削除され、別のサイトで同じ行に対して更新が行われる場合。この結果、存在しない行を更新しようとする可能性や、または複数のサイトで同じ間隔で同じ行を削除しようとする可能性があります。

> **関連項目：** その他のタイプのデータ競合の詳細は、第 5 章「競合解消の概念およびアーキテクチャ」を参照してください。

データ競合が検出されると、次のアクションが発生します。

- 競合解消方法によって、データ競合の解消が試みられます。
- 競合が解消されない場合は、接続先サイトのエラー・キューにデータ競合がロギングされます。

エラー・キューにデータ競合がロギングされた場合、データベース管理者はデータ競合を手動で解消する必要があります。

Oracle またはユーザー定義の競合解消方法を使用する場合、Oracle Server はデータ競合の自動的な解消を試みます。インプリメントされている競合解消方法は、レプリケート環境で定義されているビジネス・ルールに準拠し、データ収束を保証するものであることが必要です。

マスター・レプリケーションのアーキテクチャ

このマニュアル、Replication Manager のオンライン・ヘルプおよび『Oracle8*i* レプリケーション・マネージメント API リファレンス』で説明している手順や例に従えばレプリケート環境を作成できますが、レプリケーションのアーキテクチャを理解すると、レプリケーションをサポートするデータベース環境の設定、レプリケーション環境のチューニングやトラブルシューティングを、必要に応じて実施するために重要な情報が得られます。この項では、メカニズムとプロセスの観点からレプリケーションのアーキテクチャについて説明します。

マスター・サイトのメカニズム

Oracle では、レプリケート環境をサポートするために、マルチマスター・レプリケーション環境またはシングル・マスター・レプリケーション環境のいずれかに関係する各マスター・サイトで次のメカニズムを使用しています。これらのマスター・サイト・メカニズムの一部は、特定の環境でのみ必要となります。

マスター・サイトのロールおよびユーザー

セキュリティ要件によっては、管理、伝播および受信の 3 つのロールが 1 人のレプリケーション管理者に統合されている場合があります。実際、ほとんどのマルチマスター・レプリケーション環境では、1 人のユーザーがレプリケーションの管理、伝播および受信のロールを実行します。セキュリティ要件がより厳格な場合は、次のロールを別々のユーザーに割り当てることができます。

注意： このコンテキストのロールという用語は、SQL 用語のロールとは関連がありません。レプリケーションにおけるロールは、PL/SQL のストアド・プロシージャまたは個々の権限（あるいはその両方）を使用して付与されます。

レプリケーション管理者 レプリケーション管理者は、レプリケーション環境のマスター・サイトに関連するすべての管理機能を実行します。一般的に、1 つのレプリケーション環境に対して 1 人のレプリケーション管理者を置くことをお薦めします。レプリケーション管理者には、レプリケーションをサポートするデータベースの準備のみでなく、個々のマスター・レプリケーション・グループの作成とメンテナンス、各マスター・サイトの追加と削除、キューの管理およびレプリケーション環境の状態（通常および停止）の制御にも責任があります。レプリケーション管理者のデフォルトのユーザー名は REPADMIN ですが、別のユーザー名を使用することもできます。

プロパゲータ プロパゲータは、遅延トランザクション・キューに含まれる各トランザクションを接続先に伝播する作業を実行します。プロパゲータは、各データベースに 1 人存在します。つまり、異なるスキーマを管理するために複数のレプリケーション管理者を置くことはできますが、それぞれのデータベースには 1 人のプロパゲータしか配置できません。

受信者 受信者は、プロパゲータから遅延トランザクションを受信し、それを適用します。受信者に遅延トランザクション内のコールを適用する適切な権限が付与されていない場合、遅延トランザクション全体が接続先のエラー・キューに入れられます。

データベース・リンク

データベース・リンクは、マスター・サイトとスナップショット・サイト間でデータをレプリケートするための通信パスを指定します。マルチマスター環境では、各マスター・サイトからその他の各マスター・サイトに対してデータベース・リンクが存在します。つまり、各マスター・サイトにつき N - 1 個（N はマスター・サイトの合計数）のデータベース・リンクが存在します。

図 2-3 マスター・サイト間のデータベース・リンク

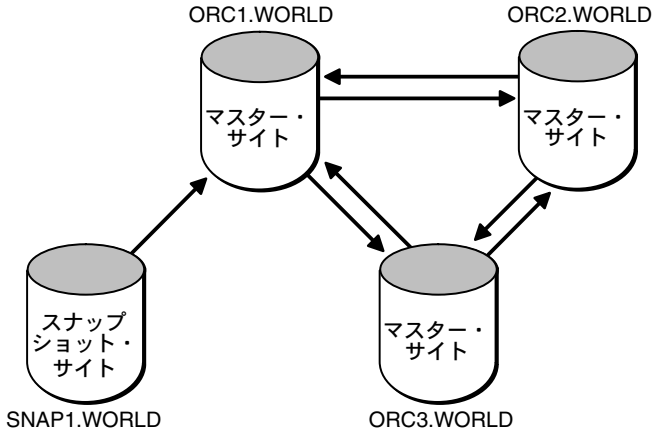


図 2-3 では、各マスター・サイトから他のマスター・サイトに対して 2 つ（N-1、すなわちこの例では 3-1=2）のデータベース・リンクが存在します。この構成では、マルチマスター・レプリケーションに必要な両方向通信チャネルがマスター・サイト間で保証されます。

最も基本的なデータベース・リンクは、個々のマスター・サイトのレプリケーション管理者と、関係する他のマスター・サイトのレプリケーション管理者とを結ぶデータベース・リンクです。

ただし、通常はレプリケーション環境にデータベース・リンクのセットを追加します。レプリケーション管理者のデータベース・リンクを作成する前に、CONNECT TO 句を指定せずに各マスター・サイト間にパブリック・データベース・リンクを作成します。パブリック・データベース・リンクでは、USING 句を使用して各データベース・リンクのターゲットを指定します。

パブリック・データベース・リンクを作成した後で、レプリケーション管理者のプライベート・データベース・リンクを作成できます。CONNECT TO 句の指定は必要ですが、パブリック・データベース・リンクを作成すると、USING 句を指定する必要はありません。

パブリック・データベース・リンクとプライベート・データベース・リンクの両方を使用すると、データベース・リンクの管理に必要な労力が軽減されます。次の利点もあります。

- データベース・リンクのターゲット・データベースが変更された場合に実行する必要があるのは、パブリック・データベース・リンクの USING 句の変更のみです。ターゲットが同じプライベート・データベース・リンクはすべて、パブリック・データベース・リンクの USING 句で定義した接続先にリンクされます。
- 複数のセットのプライベート・データベース・リンクで同じパブリック・データベース・リンクを共有でき、しかもデータベース・リンクの管理が簡単になります。

前述のように、レプリケーション管理者は、通常、マルチマスター環境の管理作業および伝播作業を実行します。これらの作業を実行するのは 1 人のユーザーのみであるため、レプリケーション管理者用に 1 つのセットのプライベート・データベース・リンクのみを作成する必要があります。

ただし、レプリケーション管理者以外のユーザーが伝播を実行するマルチマスター・レプリケーション環境では、これらのユーザー用に適切な数のプライベート・データベース・リンクを作成する必要があります。

接続修飾子 接続修飾子を使用すると、同じリモート・データベースにリンクされている複数のデータベース・リンクに異なるパスを指定して接続を確立できます。たとえば、データベース DBS1 では、DBS1.WORLD という名前の 2 つのパブリック・データベース・リンクに異なるパスを指定して、リモート・データベースに接続できます。

- DBS1.WORLD@ETHERNET - イーサネット・リンクを使用して DBS1 に接続するリンクです。
- DBS1.WORLD@MODEM - モデム・リンクを使用して DBS1 に接続する別のリンクです。

レプリケーションでは、接続修飾子の使用により、複数のマスター・グループの伝播特性をより厳密に制御できます。たとえば、各マスター・サイトに 3 つの別のマスター・グループがあり、接続修飾子を指定していないと想定します。この場合、遅延トランザクション・キューの伝播のスケジューリング特性はどのマスター・グループも同じです。この 3 つのマスター・グループのうち、1 つのマスター・グループが遅延トランザクションを 1 時間に 1 回伝播し、残りの 2 つのマスター・グループが 1 日に 1 回伝播したとすると、コストが高くなる可能性があります。

接続修飾子をマスター・グループと対応付けると、遅延トランザクション・キューの伝播のスケジューリング特性を、前述のようなデータベース・レベルではなくマスター・グループ・レベルで個別に定義できます。

> **関連項目：** データベース・リンクの接続修飾子の定義方法の詳細は、『Oracle8*i* 分散システム 』の第 2 章を参照してください。

新規マスター・グループを作成するときに、そのグループに対応するすべてのスケジュール・リンクに 1 つの接続修飾子を使用するよう指定できます。ただし、マスター・グループに接続修飾子を使用すると、各マスター・サイトで接続修飾子を使用してデータベース・リ

ンクを作成しないと情報が伝播されません。マスター・グループを作成した後は、グループの接続修飾子の削除、追加または変更はできません。

注意： 接続修飾されたリンクと複数のマスター・グループを使用するマルチマスター環境でのトランザクションの一貫性を保持するため、接続修飾子が異なる複数グループのレプリケーション・オブジェクトを 1 つのトランザクションで操作できません。

注意： 接続修飾子を使用する場合は、すべてのマスター・サイトの OPEN_LINKS 初期化パラメータの値を大きくすることが必要な場合があります。デフォルトでは、プロセスごとのオープン・リンク数は 4 つです。用途に応じて、必要な値を見積もってください。OPEN_LINKS の詳細は、7-3 ページの「初期化パラメータ」および『Oracle8*i* リファレンス・マニュアル』を参照してください。

レプリケーション・オブジェクト

レプリケーション環境の中で最も理解しやすいのは、レプリケート・オブジェクト自体です。レプリケート・オブジェクトの中では、レプリケート表がレプリケート環境の基盤です。次の項では、関連するデータベース・オブジェクトのレプリケーションについて説明します。特に、特定のタイプのデータベース・オブジェクトをレプリケートする場合の利点および制限事項を中心に説明します。

表 ほとんどの場合、レプリケート表がレプリケーション環境の基盤です。レプリケーション用の表を選択し、レプリケーション・サポートを生成すると、表に対して DML が適用されているかどうかを検出するために、内部トリガーによって監視が行われます（2-19 ページの「内部トリガー」を参照してください）。

表をレプリケートするときは、表構造と表データをリモート・データ・サイトにレプリケートすることもできますし、表構造のみをレプリケートすることもできます。また、同じ名前と構造の表がターゲットのレプリケーション・サイトにすでに存在する場合は、レプリケーション環境内の既存のオブジェクトを使用することもできます。

注意： 自己参照型整合性制約がある表では、正しい順序で削除が実行される保証はありません。このような表で削除を実行する場合は、プロシージャ・レプリケーションを使用します。詳細は、6-2 ページの「プロシージャ・レプリケーションの使用方法」を参照してください。

表レプリケーションの代替ユーザー

表のレプリケーションは、表データに加えられたすべての変更をレプリケーション環境に関係するすべてのサイトにレプリケートすることを目的としていますが、これ以外の用途もあります。

オブジェクトおよびデータの移送 : オブジェクトがターゲットの接続先サイトにレプリケートされると、それ以降レプリケーション・サポートは自動的には生成されません。オブジェクトとデータの移送により、オブジェクトとデータをリモートの接続先に簡単に配布できます。レプリケーション・オブジェクトの削除およびレプリケーション・サポートの生成を行っていない場合は、表（またはその他のオブジェクト）およびデータはリモートの接続先サイトに残り、リモートの接続先サイトの変更は一切レプリケートされません。この方法により、標準データベース環境とデータ・セットを新しいデータベース環境に配布できます。

オブジェクトの移送 : 同様に、データをコピーせずにターゲットの接続先サイトに表をレプリケートできます。この方法により、接続先サイトにオブジェクトが作成されますが、データは移入されません。したがって、空のデータベース環境を即時に配布できます。

索引 レプリケーション用の表が選択され、リモート接続先サイトで作成されると、表内での制約の施行に使用する索引がリモート・サイトで自動的に作成されます。ただし、パフォーマンス上の理由から索引を使用する場合は、レプリケーション用の索引を明示的に選択して、レプリケーション環境に関係する他のマスター・サイトに作成する必要があります。索引が別のサイトにレプリケートされると、ローカルで作成されたかのように動作します。変更を獲得するためのトリガーを監視するレプリケーション・メカニズムはありません。

パッケージおよびパッケージ本体 レプリケーション用のパッケージとパッケージ本体を選択し、必要なレプリケーション・サポートを生成すると、プロシージャ・レプリケーションを実行できます。プロシージャ・レプリケーションを使用すると、レプリケート環境でシリアルに実行されて大量の行を処理する、大規模なバッチ指向操作のパフォーマンスが向上します。

レプリケーション・サポートを使用するプロシージャのパラメータはすべて、IN パラメータであることが必要です。OUT モードおよび IN/OUT モードはサポートされていません。このパラメータにサポートされているデータ型は、NUMBER、DATE、VARCHAR2、CHAR、ROWID、RAW、BLOB、CLOB、NCHAR、NVARCHAR および NCLOB です。レプリケート・プロシージャは、パッケージ内で宣言する必要があります。スタンドアロンのプロシージャは、レプリケーション・サポートを使用できません。

関連項目： プロシージャ・レプリケーションの使用方法の詳細は、6-2 ページの「プロシージャ・レプリケーションの使用方法」を参照してください。

注意： 2-16 ページの「表レプリケーションの代替ユーザー」で説明した概念と同様に、レプリケーション・サポートを生成せずにレプリケーション用のパッケージおよびパッケージ本体を選択すると、このオブジェクトをリモート・サイトに簡単に配布できます。ただしその場合、パッケージに対するコールはレプリケートされません。

プロシージャおよびファンクション パッケージの一部として宣言されていないプロシージャおよびファンクションは、レプリケーション・サポートされません。スタンドアロンのプロシージャとファンクションを使用してプロシージャ・レプリケーション環境を作成することはできませんが、レプリケーションを使用してこれらのスタンドアロンのプロシージャとファンクションをレプリケーション環境の各サイトに配布することはできます。スタンドアロンのプロシージャまたはファンクションがレプリケーションを使用してリモート・サイトで作成されると、作成されたオブジェクトはレプリケーション・サポートされず、ローカルで作成されたかのように動作します。

トリガー 各マスター・サイトにアプリケーション・ロジックまたはデータベース・ロジックがあることを確認するために、レプリケーション用のトリガーを選択できます。トリガーのレプリケーションの重要な例として、DML が表に適用されたときに表に自動的にタイムスタンプを挿入するトリガーのレプリケーションがあります。

トリガーが再起動されないようにするには、API コールをトリガーに挿入して、トリガーがローカル・コールまたはリモート・コールによって起動されているかどうかを検出することが重要です。これにより、トリガーが行を更新して、その更新が別のマスターサイトに伝播されたときトリガー再起動の原因となる状況を回避できます。次のコード例の 5 行目を見てください。

```
1) CREATE TRIGGER scott.insert_time
2)    BEFORE
3)       INSERT OR UPDATE ON scott.emp FOR EACH ROW
4)    BEGIN
5)       IF DBMS_REPUTIL.FROM_REMOTE = FALSE THEN
6)          :NEW.TIMESTAMP := SYSDATE;
7)       END IF;
8)    END;
```

DBMS_REPUTIL.FROM_REMOTE ファンクションによって挿入または更新がローカルで開始されたと判断された場合、定義されているアクション（タイムスタンプ割当て）が発生します。挿入または更新がリモート・サイトから開始されたと判断された場合は、タイムスタンプ値は割り当てられません。

順序 異なるデータベースの 2 つの順序で同じ値が生成されることがあるため、順序のレプリケーションはサポートしていません。

順序のレプリケーションのかわりに、一意の値を生成し、一意性のデータ競合を確実に回避する方法が 3 つあります。第 1 の方法として、Oracle8*i* リリース 8.1.5 以降では、次の SELECT 文を実行して一意の識別子を取り出すことができます。

```
SELECT sys_guid() FROM dual;
```

SQL 演算子は、グローバルに一意な 16 バイトの識別子を戻します。この値は、タイムスタンプと日付スタンプおよびマシン識別子を使用してグローバルに一意な識別子を生成するアルゴリズムに基づいています。グローバルに一意な識別子は、次の形式で表示されます。

```
4595EF13AB785E73E03400400B40F58B
```

sys_guid() ファンクションを使用する第 2 の方法として、各マスター・サイトに順序を作成し、サイト名（またはその他のグローバルに一意な値）とローカルの順序を連結する方法があります。この方法は、順序の値の重複や、2-10 ページの「競合解消の概念」で説明している挿入競合の発生を回避するのに役立ちます。

さらに、レプリケーション環境内で一意の値を生成するように各マスター・サイトで順序を作成する方法があります。これを実現するには、開始値、増分値および最大値の組合せを使用します。たとえば、次のように設定すると想定します。

表 2–2

パラメータ	マスター・サイト A	マスター・サイト B	マスター・サイト C
start_with	1	3	5
increment_by	10	10	10
範囲の例	1、11、21、31、41、...	3、13、23、33、43、...	5、15、25、35、45、...

同様の方法を使用して、各サイトに対して一意の範囲を生成する START_WITH と MAXVALUE を指定することにより、各マスター・サイトごとに異なる範囲を定義できます。

ビューまたはシノニム ビューまたはシノニムをレプリケーション環境に関係する他のマスター・サイトに配布するためには、レプリケーションを使用します。ビューまたはシノニムは、他のサイトにレプリケートされると、それがローカルで作成されたかのように動作します。変更を獲得するためにオブジェクトを監視する内部トリガーまたはパッケージはありません。ただし、レプリケート・オブジェクトなので、Replication Manager またはレプリケーション・マネージメント API を使用して削除または変更できます。

内部トリガー

内部トリガーは、レプリケート・データの更新に関する情報の獲得および格納に使用します。内部トリガーによって、ローカル・サイトで加えられたデータ変更をリモート・サイトで再生成する RPC が作成されます。遅延 RPC は、遅延トランザクション・キューに格納され、レプリケーション環境に関係する他のマスター・サイトに伝播されます。データ・レプリケーションをサポートする内部トリガーは、Oracle Server 実行モジュール内の必須コンポーネントです。したがって、システム・リソースを最小限に使用して、レプリケート・データへの更新内容を高速に獲得し、格納できます。

遅延トランザクション

前述の内部トリガーによって生成される RPC を伝播（送信および実行）することにより、データ・レプリケーション情報が転送されます。この RPC は、遅延トランザクション・キューに格納されます。各 RPC には、接続先サイトでの内部プロシージャの実行コマンドだけでなく、ターゲット・サイトにレプリケートされるデータも含まれています。Oracle では分散トランザクション・プロトコルを使用しているので、グローバルなデータベース整合性が自動的に保護され、データの耐障害性が保証されます。

内部プロシージャ

内部トリガーによって作成された遅延 RPC がレプリケーション環境に関係する他のマスター・サイトに伝播されると、遅延 RPC は接続先サイトの内部プロシージャによってリモート・サイトで適用されます。この内部プロシージャは、表のレプリケーション・サポートを生成するときに自動的にアクティブになります。また、この内部プロシージャは、発行元サイトの遅延トランザクション・キューから受信される RPC に基づいて実行されます。

キュー

次のキューは、Oracle レプリケーションによって生成されるトランザクションを管理します。

遅延トランザクション・キュー このキューには、マスター・グループ内の別の接続先に伝播されるトランザクション（たとえば、DML）が格納されます。遅延伝播用に、サイトの遅延トランザクション・キューの内部トリガーによって生成される RPC が格納されます。また、トランザクションを構成するすべての RPC がトランザクションとして伝播されてリモートで適用されるように、開始トランザクションに関する情報も記録されます。Oracle のレプリケーション機能では、アドバンスト・キュー・メカニズムを使用して遅延トランザクション・キューがインプリメントされます。

エラー・キュー このキューには、ローカル・サイトでの適用に失敗した遅延トランザクションに関する情報が格納されます。エラー・キューにはレプリケーション環境のその他の各マスター・サイトのエラー情報は表示されません。エラー状態が解消されると、DBA は、トランザクションを再度実行するか、またはエラー・キューからトランザクションを削除できます。

ジョブ・キュー 伝播プロセスは、ジョブ・キュー・メカニズムおよび遅延トランザクションを使用して管理されます。各サーバーには、ローカル・ジョブ・キューがあります。サーバーのジョブ・キューは、ジョブの実行を要求する PL/SQL コールやジョブの実行時期など、ローカル・ジョブに関する情報を格納するデータベース表です。レプリケーション環境の一般的なジョブには、遅延トランザクションをリモートのマスター・サイトにプッシュするジョブ、遅延トランザクション・キューから適用済のトランザクションをパージするジョブおよびスナップショットのリフレッシュ・グループをリフレッシュするジョブがあります。

管理メカニズム

通常はレプリケート環境をサポートする管理作業を処理するために、いくつかのメカニズムが必要です。このメカニズムを使用すると、レプリケーション環境をオンまたはオフにしたり、レプリケート環境の作成時または変更時に生成される管理作業を監視できます。

レプリケーションの操作モード

レプリケーション環境の操作モードは 3 つあります。

通常 通常モードのレプリケート環境では、レプリケーションを使用できます。レプリケート・オブジェクトに対してトランザクションを実行でき、トランザクションは適切に伝播されます。

休止 休止モードは、レプリケート環境が通常モードから停止モードに移行する段階のモードです。レプリケーション環境の休止中は、ユーザーは、レプリケート・オブジェクトに対してトランザクションを実行できませんが、既存の遅延トランザクションは伝播されます。休止表に対する問合せはできます。各接続先へのすべての遅延トランザクションが正しく伝播終了すると、レプリケーション環境は停止モードに変わります。

停止 停止モードのレプリケーション環境では、レプリケーションの通常の使用が不可になっていて、主に管理目的（レプリケート・オブジェクトの追加、削除または変更）に使用されている状態です。停止状態では、レプリケーションをオフにしない限り、ユーザーは停止中のマスター・グループのレプリケート・オブジェクトに対してトランザクションを実行できません。レプリケーションをオフにすると、レプリケーションが再開されたときにデータの不一致が発生することがあります。トランザクションには、レプリケート表に対する DML またはレプリケート・プロシージャのラッパーの実行が含まれます。マスター表が停止中の場合は、そのマスター表をベースとするスナップショットからターゲットのマスター表に変更内容を伝播できませんが、スナップショットをローカルで変更することはできます。

レプリケーション環境は、マスター・グループ・レベルで停止します。ただし、レプリケート・マスター・グループに関係するすべてのマスター・サイトは影響を受けます。マスター・グループが停止状態になると、遅延トランザクション・キューのトランザクションが他のマスター・サイトに正常に伝播されたか、またはエラー・キューに入れられたことが確認できます。1 つのマスター・グループを停止しても、その他のマスター・グループには影響しません。通常モードのマスター・グループでは、その他のマスター・グループの停止中も更新処理を継続できます。

レプリケーション・モードの制御

レプリケーションの 3 つの操作モードは、中断および再開の 2 つのメカニズムで制御されます（休止モードは通常モードから停止モードへの移行段階のモードです）。

中断 中断メカニズムを実行すると、マスター・グループのレプリケーション操作モードが、通常から停止への移行モードである休止モードになります。遅延トランザクション・キューに、マスター・グループへ未伝播の遅延トランザクションがないときは、レプリケーション環境は停止モードに変わります。

中断メカニズムを実行できるのは、レプリケーション環境が通常モードの場合のみです。レプリケーション環境を変更する必要が場合は、中断メカニズムを実行します。

再開 再開メカニズムを実行すると、マスター・グループのレプリケーション操作モードが停止モードから通常モードに変わります。レプリケーション環境で管理作業（たとえば、レプリケート・オブジェクトの追加）を実行していた場合は、再開メカニズムを実行する前に管理キュー（DBA_REPCATLOG）が空であることを確認する必要があります。

管理要求

レプリケーション環境を構成および管理するために、関係する各サーバーは Oracle のレプリケーション・マネージメント API を使用します。サーバーのレプリケーション・マネージメント API は、レプリケーション機能の構成に管理者が使用できるプロシージャおよびファンクションをカプセル化した PL/SQL パッケージのセットです。また、Oracle Replication Manager では、各サイトのレプリケーション・マネージメント API のプロシージャおよびファンクションを使用して作業を実行します。

管理要求とは、Oracle のレプリケーション・マネージメント API のプロシージャまたはファンクションをコールすることです。たとえば、Replication Manager を使用して新しいマスター・グループを作成するときに、Replication Manager は、DBMS_REPCAT.CREATE_MASTER_REPGROUP プロシージャをコールして作業を完了させます。管理要求には、追加のレプリケーション・マネージメント API コールを生成して要求を処理するものもあります。

管理要求のメカニズム Replication Manager を使用するか、または DBMS_REPCAT パッケージのプロシージャをコールしてレプリケーション・システムを管理する場合、内部メカニズムによって要求が同期でブロードキャストされます。なんらかの理由で同期でのブロードキャストが失敗した場合は、エラー・メッセージが戻され、それを含むトランザクションがロールバックされます。

Oracle Server が管理要求を受信すると、DBA_REPCATLOG ビューに要求が記録され、対応する DDL 文が DBA_REPCATLOG ビューの子表に記録されます。マスター・サイトのマスター・グループの管理要求を表示すると、別のマスター・サイトからのコールバックを待機している要求が含まれる場合があります。Replication Manager を使用してレプリケーション・グループの管理要求を作成するたびに、そのグループに対してジョブが存在しない場合はローカル・ジョブ・キューにジョブが自動的に挿入されます。このジョブによって、DBMS_REPCAT.DO_DEFERRED_REPCAT_ADMIN プロシージャが定期的に実行されます。要求を同期でブロードキャストすると、レプリケートされた変更を各マスター・サイトに適用するためにこのジョブが即時に開始されます。

エラーがない場合、バックグラウンド・プロセスがジョブ実行用に使用可能なときは DO_DEFERRED_REPCAT_ADMIN が実行されます。初期化パラメータ JOB_QUEUE_INTERVAL によって、バックグラウンド・プロセスの起動頻度が決定されます。未処理ジョブを実行するために十分な数のバックグラウンド・プロセスを使用できない場合、処理が遅れることがあります。

注意： サイトで JOB_QUEUE_PROCESSES がゼロに設定されている場合は、そのサイトのすべてのグループに対して管理要求を手動で適用する必要があります。

関連項目： JOB_QUEUE_INTERVAL および JOB_QUEUE_PROCESSES の詳細は、7-3 ページ の「初期化パラメータ」および『Oracle8*i* リファレンス・マニュアル 』を参照してください。

マスター・サイトで DO_DEFERRED_REPCAT_ADMIN をコールするたびに、DBA_REPCATLOG ビューがチェックされ、実行が必要な要求があるかどうかが確認されます。管理要求が 1 つでも存在すると、要求が適用され、ローカル・ビューが適切に更新されます。このイベントは、各マスター・サイトで非同期に起こります。

DO_DEFERRED_REPCAT_ADMIN は、所定の順序でローカルの管理要求を実行します。DO_DEFERRED_REPCAT_ADMIN がマスター定義サイトではないマスター・サイトで実行される場合、可能な範囲で処理が実行されます。ただし、レプリケート表へのデータの移入などのいくつかの非同期アクティビティでは、マスター定義サイトとの通信が必要になります。通信が不可能な場合は、DO_DEFERRED_REPCAT_ADMIN によって管理要求の実行が停止され、管理要求が順序を無視して実行されるのを防ぎます。マスター定義サイトで管理要求を更新または削除する最終ステップなど、マスター定義サイトとの通信の中には遅延可能なものもありますが、そのような通信によって DO_DEFERRED_REPCAT_ADMIN による追加の要求の実行が妨げられることはありません。

各マスター・サイトで管理要求が成功または失敗したことが、各サイトの DBA_REPCATLOG ビューに記述されます。マスター・グループごとに、各管理要求に対応する状態が Replication Manager に表示されます。最後に、各マスター・サイトは、その管理要求の状態をマスター定義サイトに伝播します。マスター・サイトで要求が正常に完了した場合、マスター定義サイトの DBA_REPCATLOG ビューからサイトのコールバックが削除されます。すべてのサイトで要求が正常に完了した場合は、マスター定義サイトを含むすべてのサイトの DBA_REPCATLOG ビューのエントリがすべて削除されます。マスター定義サイト以外のサイトで要求が失敗した場合は、そのマスター・サイトの要求が削除され、マスター定義サイトの対応する AWAIT_CALLBACK 要求が失敗の理由とともに更新されます。

変更を同期でブロードキャストすることで、すべてのサイトで変更を確実に認識し、同期が取れた状態に保持できます。後の特定の時点でサイトに変更を適用でき、変更の適用に最も適した時点を自由に選択できます。

自動的に生成されるレプリケーション・サポートがオブジェクトに必要な場合、オブジェクトの変更後にレプリケーション・サポートを再生成する必要があります。Oracle では、内部

トリガーをアクティブにし、パッケージを再生成して、すべてのマスター・サイトにある変更後のオブジェクトのレプリケーションをサポートしています。

注意： エラーなしにこれらのプロシージャを完了させるには DDL をマスター定義サイトで正常に適用する必要がありますが、これによって、DDL が各マスター・サイトで正常に適用されることが保証されるわけではありません。Replication Manager により、すべての管理要求の状態が表示されます。また、DBA_REPCATLOG ビューには、一時的な状態および要求によって生成される非同期のエラー・メッセージが含まれます。

DDL 変更によって影響を受けるスナップショット・サイトがあれば、次にスナップショット・サイトのリフレッシュを実行するときに更新されます。すべてのマスター・サイトが他のマスター・サイトと通信できるのに対して、スナップショット・サイトが通信できるのは、対応付けられたマスター・サイトのみです。

スナップショットのマスターに変更を加えた結果、スナップショットの形式を変更する必要がある場合、スナップショットを削除し、再作成します。

管理キュー

DBA_REPCATLOG ビュー（通常、管理キューと呼ばれます）には、レプリケーション環境を管理および変更する管理要求が格納されます。一部の DBMS_REPCAT プロシージャは、実行すると管理キューにリストされます。たとえば、既存のマスター・グループにレプリケート表を追加すると、DBMS_REPCAT.CREATE_MASTER_REPOBJECT プロシージャという名前の要求が表示されます。

SQL*Plus などのツールを使用して DBA_REPCATLOG ビューを問い合せることによって管理キューを表示するか、または Replication Manager の接続先ノードごとに遅延トランザクションを表示できます。

各要求には、要求の状態を表す、次の 4 つの状態があります。

- **処理前** : 要求を実行する準備が整っている状態を示します。管理キューを監視したところ、ある要求がしばらく処理前の状態である場合、その処理前要求の前にコールバック待機中の要求が存在している可能性があります。
- **コールバック待機中** : ある要求が別のサイトで実行され、それが確認されるまで待機している状態を示します。要求がコールバックを受信すると、その要求は削除されるか、または状態が変更されます。
- **エラー** : 要求を正常に実行できない場合、エラー状態になります。エラー番号は、管理キューの Message 列（Replication Manager を使用している場合は Error 列）に表示されます。

注意： 要求がエラー状態にある場合は、エラー番号による説明に従ってエラー状態を解消し、要求を再送信してください。

- **コールバック中** : マスター・サイトの要求がコールバック中の状態にある場合、要求が成功したか失敗したかどうかをマスター・サイトからマスター定義サイトに知らせる必要があります。マスター定義サイトの要求がコールバック中の状態にある場合、他のすべてのマスター・サイトでカレント・フェーズが完了したときに要求の次のフェーズに進むように、マスター定義サイトからマスター・サイトに知らせる必要があります。

各マスター・サイトの管理キューには、そのマスター・サイトで実行される管理要求のみがリストされています。ただし、マスター・グループのマスター定義サイトには、各マスター・サイトで実行される管理要求がリストされています。DBA は、マスター定義サイトの管理キューを使用すると、レプリケーション環境のすべてのマスター・サイトの管理要求を監視できます。

編成メカニズム

Oracle では、編成メカニズムを使用して前述のマスター・サイトを編成し、管理メカニズムを使用して個別のレプリケーション・グループを作成します。編成メカニズムの中で最も重要なのは、マスター・グループです。追加の編成メカニズムは、レプリケート表の競合解消に使用する列をグループ化するのに役立ちます。

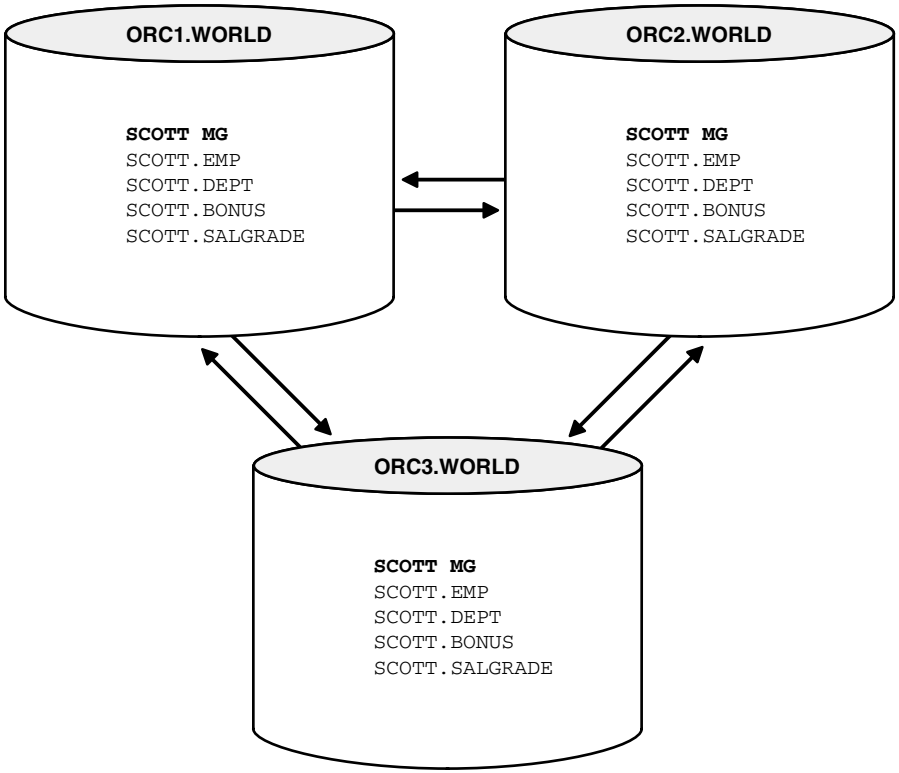
マスター・グループ

レプリケーション環境では、レプリケーション・グループを使用してレプリケーション・オブジェクトを管理します。レプリケーション・グループは、常にトランザクションの一貫性を保つように更新されるレプリケーション・オブジェクトの集合です。

レプリケーション・グループ内に関連するデータベース・オブジェクトを編成すると、多数のオブジェクトをまとめて簡単に管理できます。通常、レプリケーション・グループを作成および使用して、特定のデータベース・アプリケーションのサポートに必要なスキーマ・オブジェクトを編成します。ただし、レプリケーション・グループとスキーマが相互に対応している必要はありません。レプリケーション・グループのオブジェクトは、複数のデータベース・スキーマから作成でき、スキーマには異なるレプリケーション・グループのメンバーであるオブジェクトを入れることができます。制限は、レプリケーション・オブジェクトが 1 つのグループのメンバーにしかなれないことです。

マルチマスター・レプリケーション環境では、レプリケーション・グループはマスター・グループと呼ばれます。異なるサイトの対応するマスター・グループには、レプリケーション・オブジェクトの同一のセットが含まれている必要があります（2-15 ページの「レプリケーション・オブジェクト」を参照してください）。図 2-4 は、各マスター・サイトにレプリケート・オブジェクトの同一のレプリカが含まれているマスター・グループ "SCOTT_MG" を示します。

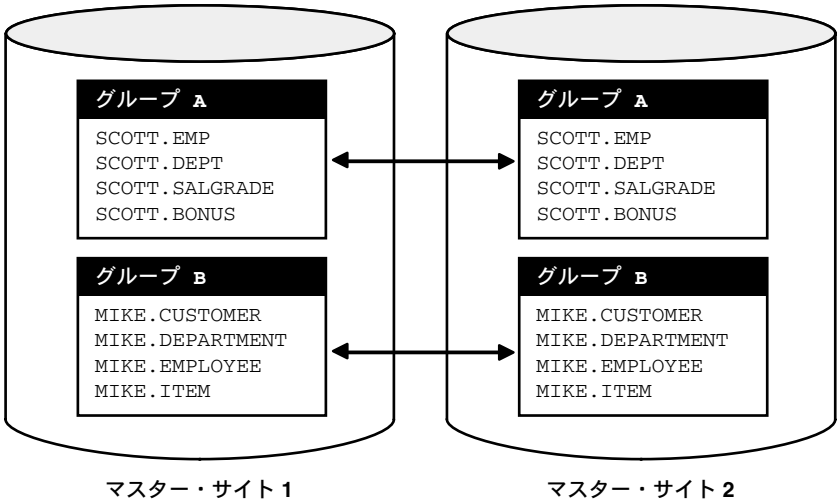
図 2-4 すべてのサイトに同じレプリケーション・オブジェクトを含むマスター・グループ SCOTT_MG



マスター・サイトのマスター・グループ編成は、スナップショット・サイトでのレプリケーション・オブジェクトの編成に不可欠です。スナップショット・サイトの編成メカニズムの詳細は、3-23 ページの「編成メカニズム」を参照してください。

また、各マスター・サイトに複数のレプリケーション・グループが含まれている例を図 2-5 に示します。各マスター・サイトの各グループに含まれているオブジェクトのセットは、同一であることが必要です。

図 2-5 各マスター・サイトで同一のマスター・グループ



列グループ

列グループは、競合解消ルーチンと関係するすべての列をグループ化する編成メカニズムを提供します。グループのいずれかの列で競合が発生した場合、残りの列を使用して競合を解消できます。たとえば、表の列グループに MIN_PRICE、LIST_PRICE、COST_PRICE および TIMESTAMP というフィールドが含まれていて、list_price フィールドで競合が発生した場合、タイムスタンプによる競合解消ルーチンを使用していたと想定すると、timestamp フィールドを使用して競合を解消できます。

表のすべての列を 1 つの列グループにまとめる必要はありません。すべての列を 1 つの列グループにまとめれば設定や管理は簡単になりますが、レプリケート表のパフォーマンスが低下し、データ競合が発生する可能性が高くなることがあります。「パフォーマンス・メカニズム」で説明しているように、最小限の通信機能によって、競合の発生時にその他の列グループから無関係なデータが送信されることはありません。すべての列を 1 つの列グループにまとめると、DBMS_REPCAT.SEND_OLD_VALUES プロシージャおよび DBMS_REPCAT.COMPARE_OLD_VALUES プロシージャを使用しない限り、最小限の通信機能の利点を失う可能性があります。

> **関連項目：** 列グループの詳細は、第 5 章「競合解消の概念およびアーキテクチャ」を参照してください。

伝播プロセス

伝播プロセスは、レプリケーション環境のすべてのマスター・サイトにすべてのアクションを送信または配布するメカニズムであり、レプリケーションにおいて不可欠な要素です。

伝播のタイプ

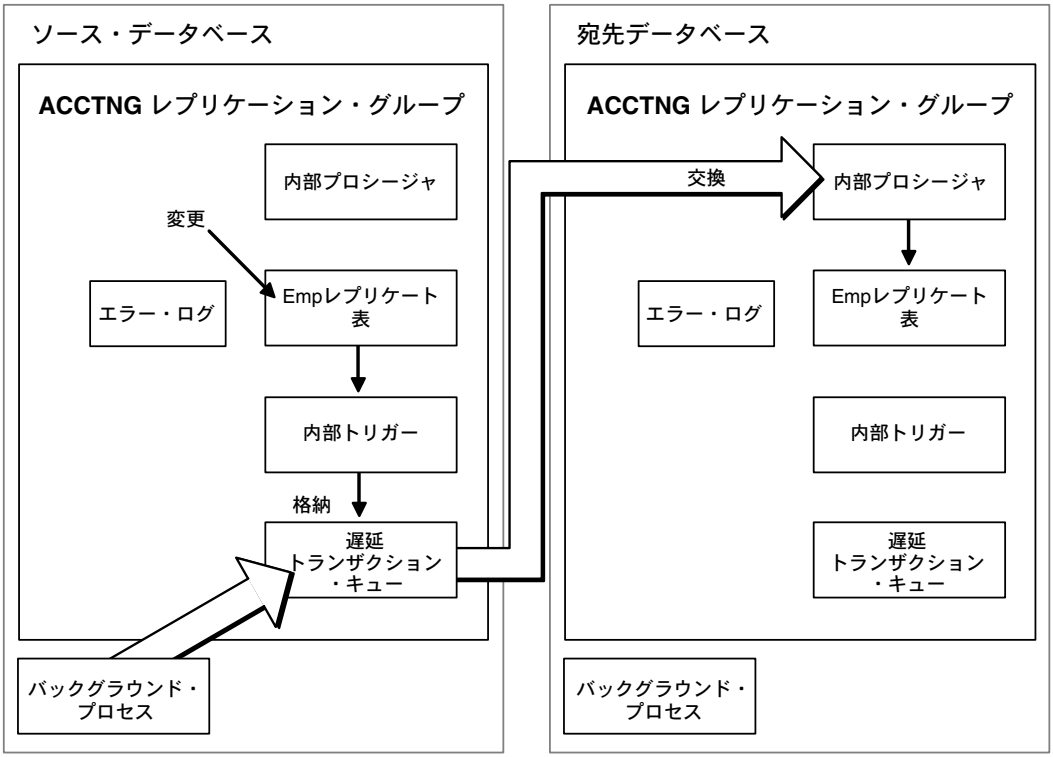
内部トリガーはレプリケート表に適用されるすべての DML を獲得するので、レプリケーション環境の他のマスター・サイトに DML を伝播または送信する必要があります。内部トリガーの詳細は、2-19 ページの「内部トリガー」を参照してください。

Oracle レプリケーションでは、非同期レプリケーションおよび同期レプリケーションの両方をサポートしています。

非同期 一般的なレプリケーション構成では、非同期データ・レプリケーションを使用します。非同期データ伝播は、アプリケーションで表のローカル・レプリカを更新し、レプリケーション情報をローカル・キューに格納し、その後で他のレプリケーション・サイトにレプリケーション情報を転送する場合に発生します。このため、非同期データ・レプリケーションは蓄積交換データ・レプリケーションとも呼ばれます。

図 2-6 に示すように Oracle では、内部トリガー、遅延トランザクション、遅延トランザクション・キューおよびジョブ・キューを使用して、更新可能スナップショットからマスター表に対してのみでなく、レプリケーション・システムのマスター・サイト間でも、データレベルの変更を非同期に伝播します。

図 2-6 非同期データ・レプリケーションのメカニズム



同期 Oracle では、特定の要件を持つアプリケーションに対しては同期データ伝播もサポートしています。同期データ・レプリケーションは、アプリケーションによって表のローカル・レプリカが更新され、同じトランザクション内で同じ表の少なくとも 1 つの他のレプリカも更新されたときに発生します。したがって、同期データ・レプリケーションはリアルタイム・データ・レプリケーションとも呼ばれます。アプリケーションでレプリケート・サイトを継続的に同期化する必要があるとき以外は、同期レプリケーションを使用しないでください。

図 2-7 同期データ・レプリケーションのメカニズム

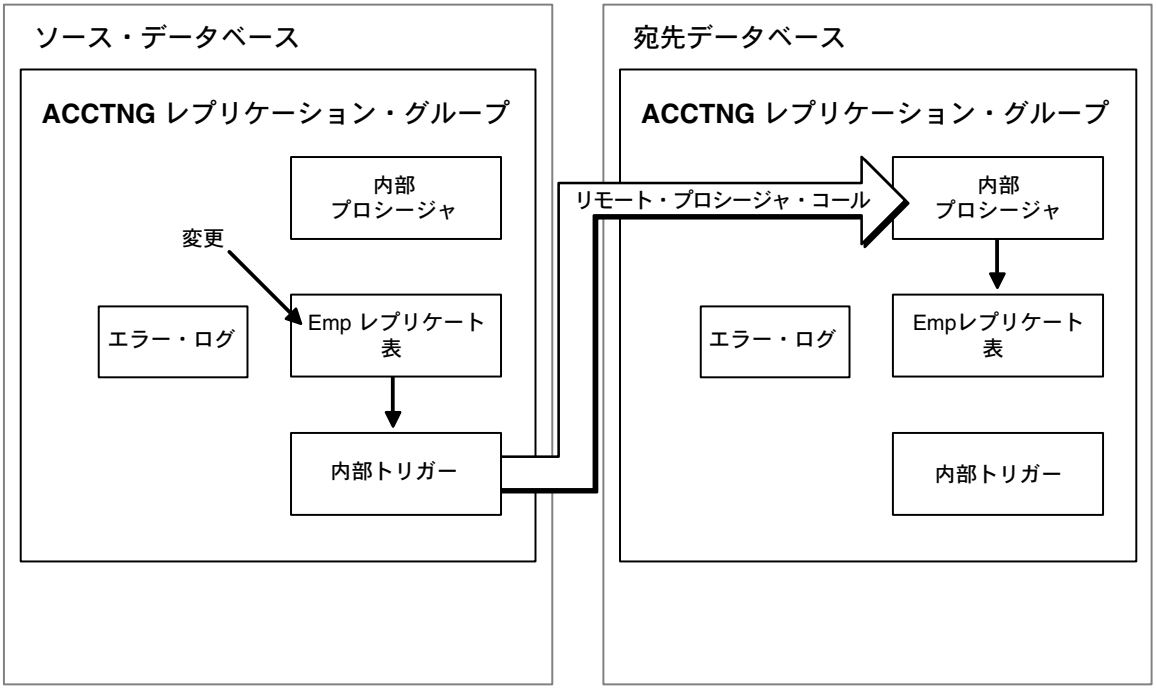
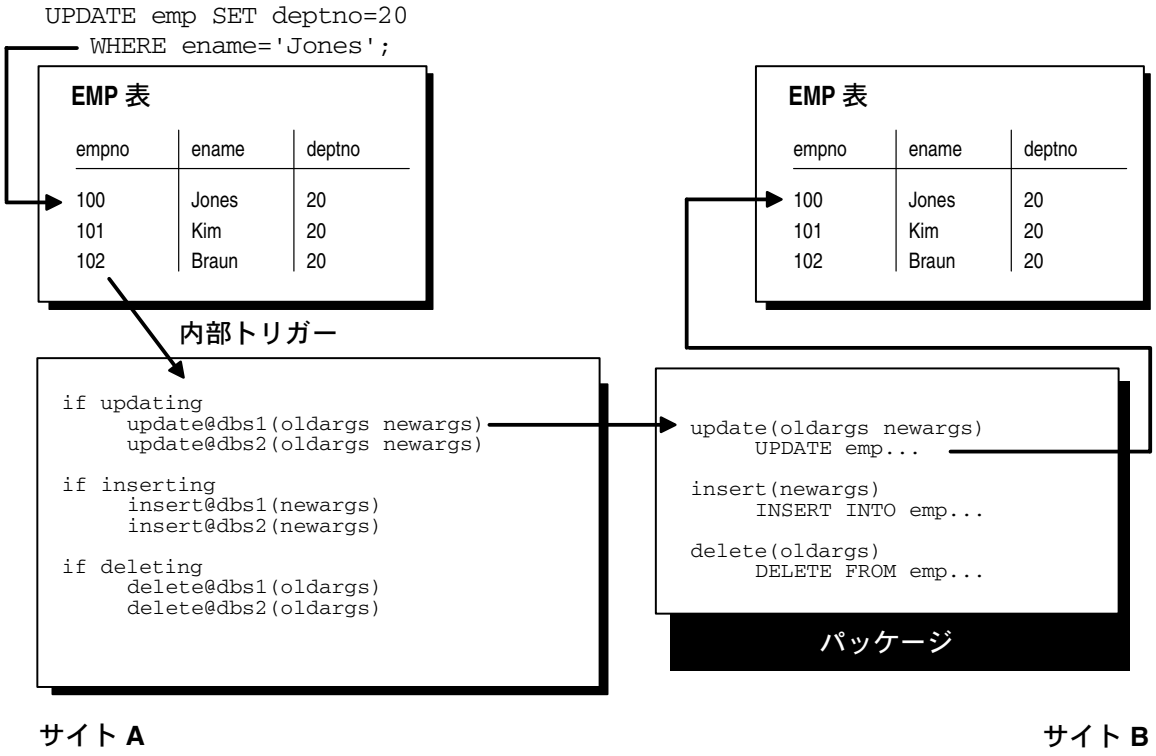


図 2-7 に示すように、同じ内部トリガーを使用して、データレベルの変更をその他のレプリケーション・サイトに非同期にレプリケートして、行レベルの同期データ・レプリケーションをサポートする RPC を生成します。ただし、この RPC の実行は延期されません。かわりに、データ・レプリケーション RPC は、ローカル・レプリカを変更する同一のトランザクションの境界内で実行されます。したがって、レプリケート表を管理する同期リンクされたすべてのサイトで、データレベルの変更が可能な必要があります。可能でない場合は、トランザクションはロールバックされます。

同期データ伝播について

図 2-8 に示すように、アプリケーションによってローカルのレプリケート表に DML による変更が加えられ、レプリケーション・グループで行レベルの同期レプリケーションを使用しているときは常に、変更内容は内部トリガーによってレプリケート環境の他のマスター・サイトに同期伝播されます。アプリケーションでローカルな変更が適用されると、内部トリガーからリモート・マスター・サイトで生成されたプロシージャに対して、レプリケーション・プロパゲータのセキュリティ・コンテキストでコールが発行されます。障害が発生した場合は、すべての分散トランザクションがコミットまたはロールバックされます。分散更新の詳細は、『Oracle8*i* 分散システム』を参照してください。

図 2-8 行レベルの同期レプリケーションを使用した変更の伝播



制限 同期レプリケーションで使用されるロック・メカニズムが原因で、デッドロックが発生することがあります。アプリケーションでレプリケート表への同期更新が実行されると、まずローカル行がロックされ、次に AFTER ROW トリガーを使用して対応するリモート行がロックされます。各サイトでトランザクションがコミットされると、ロックは解放されます。

注意： レプリケーション・データのリアルタイム伝播を使用するレプリケーション・システムは、システム内のすべてのサイトが同時に使用可能な場合にのみ機能するため、システムとネットワークの可用性に大きく依存します。

同期を取ってレプリケートされるトランザクションの接続先 同期レプリケーションのサポートに必要なリモート・プロシージャ・コールは、各オブジェクトの内部トリガーに組み込まれています。レプリケート・オブジェクトのレプリケーション・サポートを生成すると、すべてのマスター・サイトでトリガーがアクティブになり、新しいサイトに必要なリモート・プロシージャ・コールが追加されます。反対に、マスター・グループからマスター・サイトを削除すると、内部トリガーからコールが削除されます。

競合の検出 マスター・グループのすべてのサイトが互いに同期通信している場合は、アプリケーションでレプリケーションの競合が発生することはありません。ただし、1 つでも別のサイトに非同期で変更を送信するサイトがあると、レプリケート環境のいずれかのサイトで競合が発生する可能性があります。

変更が同期伝播されている場合は、エラーが発生し、ロールバックが必要になります。変更内容が非同期で伝播されている場合は、競合が自動的に検出され、ログに書き出され、適切な解消方法を指定している場合には、その競合が解消されます。

関連項目： 第 5 章「競合解消の概念およびアーキテクチャ」

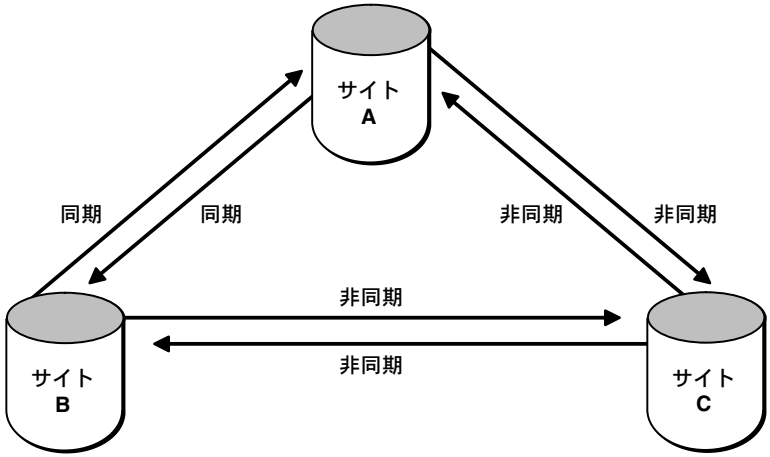
複合モードのマルチマスター・システムについて

状況によっては、一部のマスター・サイトでマスター・グループの変更内容を非同期で伝播し、他のマスター・サイトでは同期伝播するような複合モード環境の設定が必要な場合があります。新規マスター・サイトをデータ伝播モードが異なるグループに追加するときは、その順番が重要となることがあります。

たとえば、A、B および C の 3 つのマスター・サイトがあると想定します。まずマスター定義サイトとしてサイト A を作成し、次に同期伝播モードでサイト B を追加した場合、サイト A からサイト B へも、サイト B からサイト A へも変更は同期を取って送信されます。どのサイトでも遅延トランザクションは作成されないため、どちらかのサイトでリンクをスケジュールする必要はありません。

次に、非同期伝播モードでマスター・サイト C を作成するとします。伝播モードは、図 2-9 に示すようになります。

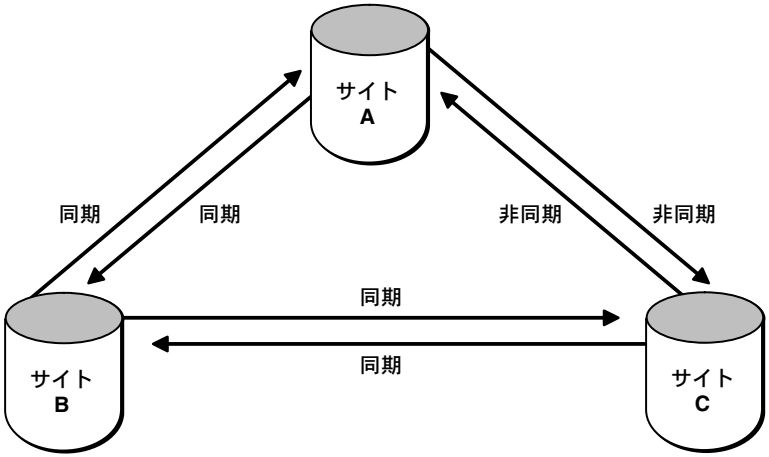
図 2-9 伝播モードの選択



次に、サイト A からサイト C、サイト B からサイト C、サイト C からサイト A およびサイト B への遅延トランザクション・キューの伝播をスケジュールする必要があります。

別の例として、サイト A をマスター定義サイトとして作成し、サイト C を非同期伝播モードで追加した後、サイト B を同期伝播モードで追加する場合を考えてみます。伝播モードは、図 2-10 に示すようになります。

図 2-10 順序に関する考慮事項



複合モードのマルチマスター・システムに新規マスター・サイトを追加するときは、その追加が既存のサイトとのデータ伝播モードに与える影響を考慮してください。

伝播の開始

同期伝播を使用する場合、DML が即時に伝播され、自動的に開始されます。非同期伝播を使用する場合は、2 つの方法で遅延トランザクションを伝播できます。通常、設定したスケジュール間隔に従って、ジョブを使用して遅延トランザクションを自動的に伝播します。また、ストアド・プロシージャを実行したり、Replication Manager を使用して、変更を手動で伝播することもできます。遅延トランザクションがジョブ・キューによって自動的に伝播されるまで待機しない場合は、遅延トランザクションを手動で伝播する必要があります。

パフォーマンス・メカニズム

他のエンタープライズ・データベース・ソリューションと同様に、データベース管理者にとってパフォーマンスは常に重要な課題です。Oracle レプリケーションには、レプリケート環境のパフォーマンスを高めるいくつかのメカニズムがあります。

パラレル伝播

パラレル伝播では、スループットを向上させるため、複数のパラレル伝送ストリームを使用してレプリケート・トランザクションを非同期で伝播します。必要に応じて依存トランザクションの実行が命令され、グローバルなデータベースの整合性が保証されます。

パラレル伝播では、使用可能なパラレル・サーバー・プロセスのプールが使用されます。これは、パラレル問合せ、パラレル・ロードおよびパラレル・リカバリなどの他のパラレル操作に使用されるプールと同じです。各サーバー・プロセスでは、シングル・ストリームを介してトランザクションが伝播されます。これらのサーバー・プロセスは、パラレル・コーディネータ・プロセスにより制御されます。コーディネータでは、トランザクション依存性を追跡しサーバー・プロセスに作業を割り当て、その進行状況を追跡します。

操作の実行中は、パラレル・プロセスがサーバーのパラレル操作に対応付けられた状態になります。操作が完了すると、サーバー・プロセスは他のパラレル操作の処理に使用できます。たとえば、接続先への遅延トランザクション・キューのパラレル・プッシュが実行されるときは、キューのプッシュに使用されるすべてのパラレル・サーバー・プロセスは、プッシュが完了するまで占有されます。

サーバー用のパラレル・サーバー・プロセスのプールを正しく構成するには、レプリケーション・システムの構成に関する問題を考慮する必要があります。

- シリアル伝播を使用するようにすべてのスケジュール・リンクを構成すると、レプリケーション・システムではパラレル・サーバー・プロセスは使用されません。したがって、レプリケーションを考慮して、パラレル・サーバーのプールのサイズを調整する必要はありません。
- 1 つ以上のスケジュール・リンクをパラレル伝播を使用するように構成する場合、それぞれのリンクが変更を接続先にプッシュするために使用するパラレル・サーバー・プロセスの数を考慮する必要があります。さらに、別の操作によって使用されないように、各プッシュでパラレル・サーバーを保持する時間を考慮する必要があります。たとえば、Delay Seconds 値を大きくして連続的な伝播用にスケジュール・リンクを構成すると、接続先にトランザクションをプッシュするために使用するパラレル・サーバー・プロセスが保持されます。したがって、サーバーの他のパラレル操作で使用するプロセスを十分に確保するために、対応するデータベース・サーバーのパラレル・サーバー・プロセスの数を増やす必要があります。

パラレル問合せプロセスのデータベース・サーバーのプールを構成するには、次の初期化パラメータを使用します。

- PARALLEL_MAX_SERVERS
- PARALLEL_MIN_SERVERS

関連項目：

- 7-3 ページ「初期化パラメータ」
- 『Oracle8*i* リファレンス・マニュアル』
- 『Oracle8*i* 概要 』

パラレル伝播の実現 ほとんどのユーザーの場合、パラレル伝播パラメータ値を 1 に設定すれば、レプリケーションで要求されるパフォーマンスを実現できます。ただし、一部のユーザーは、パラレル伝播値をさらにチューニングすることが必要な場合があります。

パラレル伝播値をさらにチューニングする場合、次の手順の実行をお薦めします。

1. パラレル伝播値を 1 に設定します。
2. 使用しているデータベース環境をテストし、伝播スループットを慎重に測定します。

   パラレル伝播値を 1 に設定することで目的のパフォーマンスを実現できた場合は、ステップ 5 に進みます。

   注意： パラレル伝播値を大きくすると、パラレル伝播のパフォーマンスが向上する一方で、追加のパラレル・スレーブのサポートに必要なリソースが増大します。

3. パラレル伝播値を 1 に設定した場合以上の伝播スループットを実現する場合は、パラレル伝播値を 2 に設定します。
4. 使用しているデータベース環境をテストし、伝播スループットを慎重に測定します。

   通常、伝播値を 2 に設定すると、伝播スループットは低下します。これは、依存トランザクションを使用可能なスレーブに割り当て、追加トランザクションを割り当てる前に必要なコミットを待機しているコーディネータに関連したラウンドトリップ・ディレイによるものです。

   パラレル伝播値を 4 に設定してステップ 3 と 4 を繰り返し、また、パラレル伝播値を 8 に設定して同じステップを繰り返します。スループットが改善されない場合は、使用している環境のトランザクションの相互依存性が非常に高いことが考えられます。パラレル伝播値を 1 に下げ、ステップ 5 に進んでください。

関連項目： トランザクション依存性を下げるテクニックの詳細は、2-38 ページの「パラレル伝播のチューニング」を参照してください。

パラレル伝播値を 2、4 または 8 のいずれかに設定することでパフォーマンスが向上した場合は、トランザクションの相互依存性は低いといえます。パラレル伝播値は 8 より大きい値に設定してもかまいません。必ず使用している環境を完全にテストしてください。パラレル伝播のパフォーマンスが向上すれば、追加のパラレル・スレーブのサポートに必要なリソースは増大することに注意してください。

5. 使用しているデータベース環境のパフォーマンスのニーズを最大限に満たすように、パラレル伝播値の調整を完了しました。

パラレル伝播のチューニング パラレル伝播のパフォーマンス上の利点を最大限に活用するには、作成する依存トランザクションの数を減らす必要があります。依存トランザクションは、依存関係にあるトランザクションがすべてがコミットされるまで開始できません。

依存トランザクション数を減らす場合、次のことを考慮してください。

- 可能であれば、トランザクションを小さくします（つまり、自律性を損なわない範囲で、コミットの回数を増やします）。
- 挿入を受け取る各表の空きリスト数を増やします。
- ホットスポット（頻繁に変更される行。同じ行に対して操作が行われる場合、そのトランザクションはシリアライズ化されます）を回避します。たとえば、行のカウンターを使用して手動で増分するのではなく、Oracle の順序を使用します。

最小限の通信

表のレプリケーション・パフォーマンスを向上させるには、レプリケート表のレプリケーション・サポートを生成するときに、必ず最小限の通信設定を使用可能にしておきます。

行の更新競合を検出および解消するには、伝播サイトから受信サイトに新旧の行に関するデータを送信する必要があります。使用している環境によって、更新競合の検出および解消をサポートするために伝播されるデータの量は異なります。

たとえば、関係するすべてのサイトが Oracle 8.0 以降のデータベースを使用している環境でレプリケート表を作成する場合は、レプリケート表内の変更された各行の競合を検出するために通信する必要があるデータの量を最少化できます。この場合は、次の値が伝播されます。

- 主キーの古い値および変更後の列グループの列の古い値（変更前の値）
- それぞれの更新された列の新しい値

注意： 挿入された行には古い値はありません。また、削除された行には新しい値はありません。

一般的に、Oracle リリース 8.0 以降のレプリケーション環境では、データ伝播を最少化して、ネットワーク上で送信されるデータの量を減らすことが可能です。それにより、システム全体のパフォーマンスが向上します。

これに対して、レプリケート環境で Oracle7 のサイトと Oracle リリース 8.0 以降のサイトが混在している場合は、更新競合を検出するために通信する行データの量を最少化することはできません。この場合、競合を検出するためには、変更された各行の新旧のデータをすべて伝播する必要があります。

関連項目： 競合解消、最小限の通信およびその他の競合解消テクニックの詳細は、5-26 ページの「パフォーマンス・メカニズム」を参照してください。

遅延秒数

パフォーマンスに直接かかわるメカニズムではありませんが、DELAY_SECONDS パラメータ値を適切に設定すると、遅延トランザクションの伝播のタイミングをより詳細に制御できます。

DELAY_SECONDS は、ジョブが遅延トランザクション・キューを認識し続ける時間を制御します。DELAY_SECONDS パラメータ値を最大限に活用した 2 つの例を次に示します。

DELAY_SECONDS = 0（デフォルト）

間隔を 60 分に設定したスケジュール・ジョブが午後 2 時 30 分に起動され、遅延トランザクション・キューをチェックする場合、既存の遅延トランザクションはすべて伝播されます。伝播には 2 分かかるので、ジョブが完了するのは午後 2 時 32 分です。

遅延トランザクションが午後 2 時 34 分にキューに入れられた場合、ジョブが完了しているためこの遅延トランザクションは伝播されません。このシナリオでは、遅延トランザクションは午後 3 時 30 分に伝播されます。

DELAY_SECONDS = 300

間隔を 60 分に設定したスケジュール・ジョブが午後 2 時 30 分に起動され、遅延トランザクション・キューをチェックする場合、既存の遅延トランザクションはすべて伝播されます。伝播には 2 分かかるので、ジョブが完了するのは午後 2 時 32 分です。

遅延トランザクションが午後 2 時 34 分にキューに入れられた場合、既存の遅延トランザクションの伝播完了後も 300 秒間（5 分間）はジョブによって遅延トランザクション・キューが認識されるので、この遅延トランザクションは伝播されます。このシナリオでは、この遅延トランザクションは午後 2 時 34 分に伝播されます。

さらに頻繁にジョブが実行されるように設定できないかという疑問が生じます。ジョブを開始および停止すると、ジョブを開始して一定時間認識し続けるよりもはるかに大きいオーバーヘッドが発生します。DELAY_SECONDS パラメータ値を使用すると、ジョブの開始および停止に伴うオーバーヘッドを削減できるのみでなく、スケジュール・ジョブのサポートに必要な REDO ロギングの量を減らすことができます。

パフォーマンスにかかわるほとんどの機能と同様に、デメリットもあります。いくつかの理由から、DELAY_SECONDS パラメータ値の長さを常にチェックする必要があります。

- **パラレル伝播**：遅延トランザクション・キューをプッシュするときに使用する各パラレル・プロセスは、その伝播ジョブが完了するまでその他のパラレル・アクティビティでは使用できません。長い DELAY_SECONDS 値を定義すると、他の操作がパラレル・プロセスを使用できなくなる可能性があります。
- **シリアル伝播**：（パラレル伝播ではなく）シリアル伝播しか使用しない場合、DELAY_SECONDS 値を設定するとオープン・セッションが遅延時間中スリープするため、この値を設定しても前述のようなメリットはありません。
- **精度パージ**：DBMS_DEFER_SYS.PURGE プロシージャを使用しているときに precise_purge 方法を指定し、DELAY_SECONDS 値に大きい値を定義した場合、指定したパージの実行時にパフォーマンスが低下する可能性があります。

一般的に、DELAY_SECONDS 値を 20 分（DELAY_SECONDS=1200）以上に設定するメリットはありません。

また、シリアル伝播しか使用しない場合（パラレル伝播 =0）は、DELAY_SECONDS 値を大きくするメリットも考慮する必要があります。パラレル伝播と異なりシリアル伝播では、遅延秒数値の時間が経過した後にキューのみがチェックされます。パラレル伝播値を 20 分に設定する場合、パラレル・プッシュによって 1 分に 1 回チェックされます。リソースを 20 分間ロックできる場合（その他のジョブは SNP プロセスを使用できません）、遅延秒数値を 1200 に設定できます。

リソースを 20 分間ロックできない場合は、DELAY_SECONDS 値を 10 秒または 20 秒に設定します。1200 秒に設定した場合よりもジョブの実行回数を多くする必要がありますが、遅延秒数機能を利用するメリットは残ります（0 秒に設定した場合と比較した場合）。

競合解消メカニズム

レプリケーション環境の受信マスター・サイトでは、次のような更新競合、一意性競合および削除競合が検出されます。

- 主キー列または更新された列グループの列のいずれかで、レプリケート行の古い値（変更前の値）と受信サイトの同じ行のカレント値が異なる場合、受信サイトで更新競合が検出されます。
- レプリケート行で INSERT または UPDATE を実行中に一意性の制約違反が発生した場合、受信サイトで一意性競合が検出されます。
- 行の主キーが存在しないために UPDATE または DELETE の対象の行が見つからない場合、受信サイトで削除競合が検出されます。

注意： 行の更新競合を検出および解消するには、伝播サイトから受信サイトに新旧の行に関する一定量のデータを送信する必要があります。パフォーマンスを最大にするには、更新競合の検出および解消サポートに使用するデータの量をチューニングします。詳細は、5-26 ページの「最小限の通信」を参照してください。

競合検出時の行の識別

レプリケーションの競合を正確に検出するには、データ・レプリケーション時に別々のサイトで対応する行を一意に識別して、一致させる必要があります。通常、Oracle のレプリケーション機能では、表の主キーを使用して表内の行を一意に識別します。表に主キーがない場合は、代替キーを指定する必要があります。代替キーとは、データ・レプリケーション時に表内の行の識別に使用する列または列のセットです。

注意： アプリケーションによって表の識別列を更新しないでください。この表により、行を識別し、レプリケート・データの整合性を保つことができます。

データ競合の解消

Oracle には、データ競合が検出されたときに競合解消方法を定義するメカニズムがあります。次の競合解消方法があります。

- 最も新しいタイムスタンプおよび最も古いタイムスタンプ
- 上書きおよび廃棄
- 最大値および最小値
- 加算および平均
- タイムスタンプ
- 優先グループ
- サイト優先順位

組込まれている競合解消方法ではレプリケート環境のニーズを満たさない場合、PL/SQL を使用して独自の競合解消方法を作成して、ユーザー定義の競合解消方法としてインプリメントできます。競合解消の仕組みの詳細は、第 5 章「競合解消の概念およびアーキテクチャ」を参照してください。

関連項目： Replication Manager を使用して競合解消方法をインプリメントする方法の詳細は、Replication Manager のオンライン・ヘルプを参照してください。レプリケーション・マネージメント API を使用して競合解消方法をインプリメントする方法の詳細は、『Oracle8*i* レプリケーション・マネージメント API リファレンス』の第 6 章「競合解消」を参照してください。

3

スナップショットの概念およびアーキテクチャ

この章では、スナップショットの概念およびアーキテクチャ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。

- スナップショットの概念
- スナップショットのアーキテクチャ
- スナップショットの準備
- スナップショット・ログの作成
- スナップショット環境の作成

スナップショットの概念

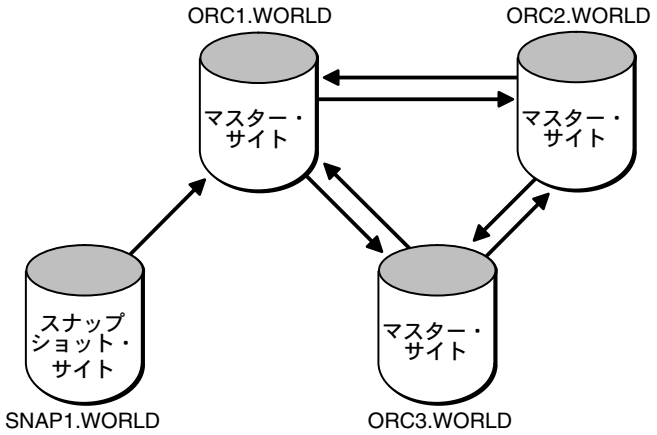
Oracle では、スナップショット（マテリアライズド・ビューとも呼ばれます）を使用して、レプリケート環境のマスター・サイト以外のサイトにデータをレプリケートし、データ・ウェアハウス環境のコストのかかる問合せをキャッシュします。この章およびこの『Oracle8*i* レプリケーション・ガイド』マニュアル全体では、レプリケート環境で使用するスナップショットについて説明します。

関連項目： データ・ウェアハウスのマテリアライズド・ビューの詳細は、『Oracle8*i* データ・ウェアハウス』を参照してください。

スナップショットとは

スナップショットとは、ある一時点におけるターゲット・マスター表のレプリカのことです。マルチマスター・レプリケーションでは、他のマスター・サイトによって継続的に表が更新されますが、スナップショット・レプリケーションでは、1 つのマスター・サイトからバッチ更新（リフレッシュとも呼ばれます）が実行されるたびに、1 つ以上のマスター表によって更新されます。図 3-1 を参照してください。

図 3-1 レプリケート環境の 1 つのマスター・サイトに接続されたスナップショット



スナップショットに WHERE 句を定義して、スナップショット・サイトにカスタマイズしたデータ・セットを含めることもできます。このようなスナップショットは、会社の全てのデータ・セットを必要としない地域事業所や営業担当員に役立ちます。

スナップショットをリフレッシュすると、マスター表に加えられた変更をすべて調査し、スナップショットに適用する変更があるかどうかの確認が行われます。したがって、スナップショットの最後のリフレッシュ以降にマスター表に変更がある場合は、スナップショットに

適用する変更がない場合も、リフレッシュには多少時間がかかります。ただし、スナップショットの最後のリフレッシュ以降、マスター表にまったく変更がない場合は、スナップショットのリフレッシュはすぐに完了します。

スナップショットの使用目的

スナップショットを使用する目的には、次のようなものがあります。

- ネットワーク負荷の軽減
- 大量配布
- データのサブセット化
- モバイル・コンピューティング

ネットワーク負荷の軽減

ネットワーク負荷の軽減が目的の一つである場合は、スナップショットを使用して会社のデータベースを地域サイトに分散できます。会社全体で 1 つのデータベース・サーバーにアクセスするのではなく、ユーザーの負荷を複数のデータベース・サーバーに分散します。また、スナップショットはマスター表のサブセットでもかまいません。これにより、レプリケートするデータの量が減ります。

マルチマスター・レプリケーションでも会社のデータベースが複数のサイトに分散されますが、マルチマスター・レプリケーションはトランザクションごとに処理を行うため、スナップショットを使用したレプリケーションよりもネットワーク要件が多くなります。また、マルチマスター・レプリケーションではリアルタイムまたはほぼリアルタイムに結果が提供されるため、ネットワーク・トラフィックが増大する結果となり、専用のネットワーク・リンクが必要となることもあります。

スナップショットは、1 つのマスター・サイトからバッチ処理によって効率的に更新されます。スナップショット・レプリケーションでは一定時間おきにデータが更新されるので、マルチマスター・レプリケーションの場合よりもネットワーク要件が少なく、依存性も高くありません。マルチマスター・レプリケーションではネットワーク上で常に通信が行われますが、スナップショット・レプリケーションでは定期的なリフレッシュのみが必要になります。スナップショットを使用したデータのレプリケーションでは、専用のネットワーク接続が不要なうえ、ターゲット・データへのローカル・アクセスを実行できるためデータの可用性が高くなります。これらのメリットに加え、大量配布およびデータのサブセット化（いずれもネットワーク負荷を軽減します）というメリットにより、レプリケート・データベースのパフォーマンスと信頼性が大幅に向上します。

大量配布

配置テンプレートを使用すると、スナップショット環境をローカルで事前作成できます。また、配置テンプレートを使用すれば、スナップショット環境を迅速かつ簡単に配置でき、営業支援システムやその他の大量配布環境をサポートできます。パラメータを使用すると、配置テンプレートを変更せずに個々のユーザー用のカスタム・データ・セットを作成できます。このテクノロジによって、データベースのインフラストラクチャを数百人または数千人のユーザーに提供できます。

データのサブセット化

マルチマスター・レプリケーションでは、表全体をレプリケートする必要がありますが、スナップショット・レプリケーションでは、列レベルまたは行レベル（あるいはその両方）でサブセットに基づいたデータをレプリケートできます。データのサブセットにより、特定のサイトのみに関係する情報をレプリケートできます。たとえば、各地の営業所ではその地域に必要なデータのみをレプリケートし、不要なネットワーク・トラフィックを減らすことができます。

モバイル・コンピューティング

マルチマスター・レプリケーションと異なり、スナップショットは専用のネットワーク・リンクを必要としません。ジョブをスケジュールしてリフレッシュ・プロセスを自動化することもできますが、必要なときに手動でスナップショットをリフレッシュできます。これは、ラップトップで実行する販売アプリケーションに理想的なソリューションです。たとえば、開発者は、必要なときにリフレッシュを実行するための Oracle レプリケーション・マネージメント API を販売アプリケーションに統合できます。1 日の受注業務を終えた営業担当員は、ネットワークにダイヤルアップ接続し、統合されたメカニズムを使用してデータベースをリフレッシュすることによって、本社に受注を転送できます。

使用可能なスナップショット

Oracle では、様々なレプリケーション環境（および非レプリケーション環境）で使用できるいくつかのタイプのスナップショットを用意しています。次の項では、それぞれのタイプのスナップショットと、各スナップショットに最適な環境について説明します。

次の項では、様々なタイプのスナップショットの作成例を示しています。どのタイプのスナップショットを作成する場合でも、スナップショットの問合せで表所有者のスキーマ名を指定します。たとえば、次の CREATE SNAPSHOT 文について考えます。

```
CREATE SNAPSHOT emp_snap AS SELECT * FROM scott.emp@db1.world;
```

この文では、スキーマ名に SCOTT を指定しています。

主キー

主キー・スナップショットは、デフォルトのスナップショット・タイプです。スナップショットをスナップショット・グループの一部として作成し、スナップショットを定義するときに "FOR UPDATE" と指定した場合は、主キー・スナップショットは更新可能になります。変更は、（ROWID ではなく）行の主キー値によって識別される、行レベルの変更に従って伝播されます。更新可能な主キー・スナップショットを作成するための SQL 文は、次のようになります。

```
CREATE SNAPSHOT sales.customer FOR UPDATE AS
 SELECT * FROM sales.customer@dbs1.acme.com;
```

リモート・スナップショット・サイトでデータを水平方向にパーティション化したサブセットを作成するために、主キー・スナップショットに副問合せを含めることができます。この副問合せには、基本的な WHERE 句のように単純なものと、マルチレベルの WHERE EXISTS 句のように複雑なものとがあります。選択されたクラスの副問合せを含む主キー・スナップショットに対しても、増分リフレッシュ（高速リフレッシュ）を実行できます。副問合せを含む WHERE 句を使用した副問合せスナップショットを次に示します。

```
CREATE SNAPSHOT sales.orders REFRESH FAST AS
 SELECT * FROM sales.orders@dbs1.acme.com o
 WHERE EXISTS
   (SELECT 1 FROM sales.customer@dbs1.acme.com c
    WHERE o.c_id = c.c_id AND zip = 19555);
```

関連項目： スナップショット・グループの詳細は、3-23 ページの「スナップショット・グループ」を参照してください。副問合せを使用するスナップショットの詳細は、3-9 ページの「副問合せを使用するスナップショット」を参照してください。

ROWID

Oracle では、下位互換性のために、デフォルトの主キー・スナップショットの他に、ROWID スナップショットもサポートしています。ROWID スナップショットは、マスター表の行の物理的な行識別子（ROWID）に基づいて作成されます。ROWID スナップショットを使用するのは、Oracle7 データベースのマスター表をベースとするスナップショットの場合のみです。Oracle リリース 8.0 以降のデータベースのマスター表をベースとする新しいスナップショットを作成するときは使用しないでください。

```
CREATE SNAPSHOT sales.customer REFRESH WITH ROWID AS
 SELECT * FROM sales.customer@dbs1.acme.com;
```

関連項目： ROWID スナップショットと主キー・スナップショットの相違点の詳細は、3-20 ページの「スナップショット・ログ」を参照してください。

複合スナップショット

高速リフレッシュを実行するには、スナップショットの定義問合せで一定の制限事項を守る必要があります。スナップショットの定義問合せが汎用的なものであり、制限事項を守ることができない場合は、そのスナップショットは複合スナップショットであり、高速リフレッシュを実行できません。

具体的には、スナップショットの定義問合せに次のものが含まれる場合、そのスナップショットは複合スナップショットとみなされます。

- CONNECT BY 句。
- 表 3-1「副問合せを使用するスナップショットに関する制限事項」に示した要件と適合しない句。
- UNION、INTERSECT または MINUS などの集合演算。
- 個別関数または集約関数のほとんど。ただし、定義問合せの中で個別関数または集約関数を使用しても、簡単なスナップショットにすることは可能です。

関連項目： 複合マテリアライズド・ビューの詳細は、『Oracle8*i* データ・ウェアハウス』を参照してください。Oracle マニュアルでは、スナップショットとマテリアライズド・ビューは同義で使用しています。マテリアライズド・ビューは、『Oracle8*i* データ・ウェアハウス』で使用しています。

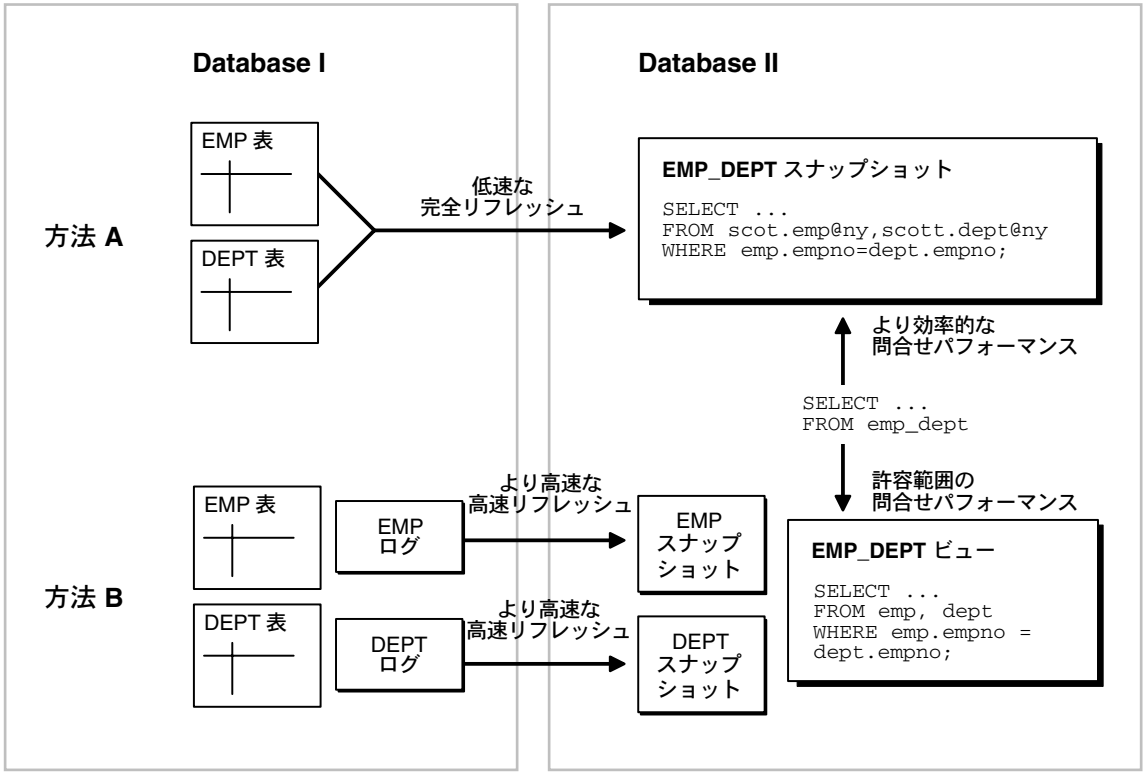
注意： 複合スナップショットでは高速リフレッシュができないため、ネットワークのパフォーマンスが低下することがあるので、できるだけ使用しないでください（詳細は、3-28 ページの「リフレッシュ・プロセス」を参照してください）。

複合スナップショットの CREATE 文の例を次に示します。

```
CREATE SNAPSHOT scott.snap_employees AS
 SELECT emp.empno, emp.ename FROM scott.emp@dbs1.acme.com
  UNION ALL
 SELECT new_emp.empno, new_emp.ename FROM scott.new_emp@dbs1.acme.com;
```

単純スナップショットと複合スナップショットの比較 アプリケーションによっては、複合スナップショットを使用することを考慮できます。図 3-2 および次の文章で、いくつかの考慮事項について説明します。

図 3-2 単純スナップショットと複合スナップショットの比較



- **複合スナップショット**: 図 3-2 の方法 A は、複合スナップショットを示しています。Database II 内のスナップショットでは、スナップショットのリフレッシュ中に結合操作が完了しているので、非常に効率的に問合せが行われます。ただし、複合スナップショットであるため、完全リフレッシュを実行する必要があります。多くの場合、完全リフレッシュは、高速リフレッシュに比べ実行速度が遅くなります。
- **結合ビューを含む単純スナップショット**: 図 3-2 の方法 B は、Database II 内の 2 つの単純スナップショットと、スナップショットのデータベース内で結合を実行するビューを示しています。ビューに対する問合せのパフォーマンスは、方法 A の複合スナップショットに対する問合せのパフォーマンスに比べ良くありません。ただし、単純スナップショットの方が、高速リフレッシュとスナップショット・ログを使用してより高速にリフレッシュできます。

要約すると、使用する方法は次のようにして判別できます。

- リフレッシュを実行する頻度が少なく、より高速な問合せのパフォーマンスを優先する場合は、方法 A を使用します。
- 頻繁にリフレッシュを実行し、問合せのパフォーマンスを優先する必要がない場合は、方法 B を使用します。

読取り専用スナップショット

これまでに説明したタイプのスナップショットはすべて、FOR UPDATE 句を省略するか、または Replication Manager の GUI インタフェースの該当するチェックボックスをオフにすることによって、読取り専用にできます。読取り専用スナップショットは、スナップショット・グループに属している必要がない場合を除き、更新可能スナップショットと同じメカニズムの多くを使用します。

> **関連項目：** 詳細は、3-23 ページの「スナップショット・グループ」を参照してください。

また、読取り専用スナップショットを使用すると、スナップショットが原因でマスター・サイトでデータ競合が発生するのを回避できますが、リモート・スナップショット・サイトでの更新はできなくなります。読取り専用スナップショットの定義例を次に示します。

```
CREATE SNAPSHOT sales.customer AS
  SELECT * FROM sales.customer@hq.acme.com
```

> **注意：** スナップショットの定義問合せでは、必ずマスター表のすべての主キー列を参照する必要があります。

スナップショットでのデータのサブセット化

場合によっては、マスター表のデータの水平方向または垂直方向のサブセットをスナップショットに反映させる場合があります。配置テンプレートを使用してスナップショットを作成する場合、垂直方向のデータ・サブセットを定義して列境界に沿ってデータをレプリケートできます。垂直パーティション化の詳細は、4-22 ページの「垂直パーティション化」を参照してください。データのパーティション化には、次のような目的があります。

- **ネットワーク・トラフィックの軽減**：スナップショットの定義問合せの WHERE 句を満たす変更のみがスナップショット・サイトに伝播されるため、転送されるデータの量が減り、ネットワーク・トラフィックが軽減します。
- **機密データの保護**：スナップショットの定義問合せを満たすデータ以外は表示できません。
- **リソース要件の削減**：スナップショットがラップトップに格納される場合は、一般にハード・ディスクは会社のサーバーよりもはるかに小さくなります。パーティション・

スナップショットを使用すれば、使用する記憶領域をかなり小さくすることができます。

- **リフレッシュ時間の短縮**：スナップショット・サイトに伝播されるデータが少なくなるので、リフレッシュ・プロセスは高速になります。このことは、ラップトップからダイヤルアップ・ネットワーク接続を使用してスナップショットをリフレッシュするユーザーには非常に有効です。

多くの場合、単純な WHERE 句を使用すれば前述の目的を達成できます。たとえば、次の文では、zip コード 19555 の顧客に関する情報を含むスナップショットを作成できます。

```
CREATE SNAPSHOT sales.customer AS
  SELECT * FROM sales.customer@dbs1.acme.com
  WHERE zip = 19555;
```

副問合せを使用するスナップショット

前述の例は、他のスナップショットへの参照制約がない個別のスナップショットでは非常にうまく機能します。しかし、単なる顧客情報よりも多くの情報が必要な場合は、これまでに説明したようなスナップショットではメンテナンスや定義が困難になります。

たとえば、CUSTOMER、ORDERS および ORDER_LINE という 3 つの表があり、これらのマスター表の参照整合性を保つ、対応する 3 つのスナップショットを作成するとします。営業担当員が zip コード 19555 のすべての顧客について顧客情報、未処理の受注およびその受注明細を取り出そうとする場合、スナップショットの定義問合せに 1 つ以上の副問合せを含むスナップショットを作成するのが最も効率のよい方法です。

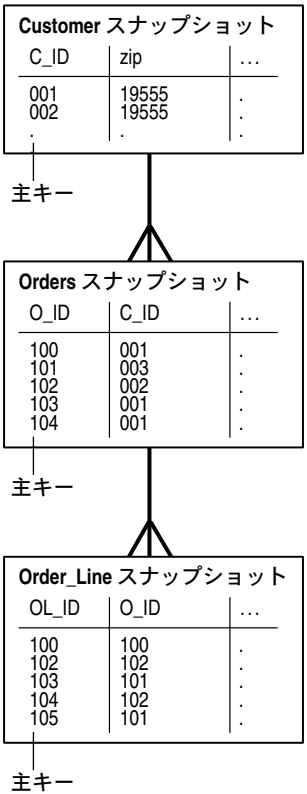
CUSTOMER マスター表は階層の最上位にあるため、CUSTOMER スナップショットに含まれるのは次のように非常に簡単な定義問合せです。

```
CREATE SNAPSHOT sales.customer AS
  SELECT * FROM sales.customer@dbs1.acme.com
  WHERE zip = 19555;
```

ORDERS スナップショットを作成するときは、zip コード 19555 の顧客のすべての受注を取り出す必要があります。図 3-3 では、CUSTOMER 表と ORDERS 表が C_ID 列で関連付けられています。次の文によって、適切なデータ・セットを含む ORDERS スナップショットを作成できます。

```
CREATE SNAPSHOT sales.orders AS
  SELECT * FROM sales.orders@dbs1.acme.com o
  WHERE EXISTS
    (SELECT c_id FROM sales.customer@dbs1.acme.com c
     WHERE o.c_id = c.c_id AND c.zip = 19555);
```

図 3-3 副問合せを使用する高度なサブセット化



ORDER_LINE スナップショットの作成方法は ORDERS スナップショットの場合と同様ですが、その場合はもう 1 つ別の副問合せを使用する必要があります。図 3-3 では、ORDER_LINE 表と ORDERS 表が O_ID 行で関連付けられています。次の文によって、適切なデータ・セットを含む ORDER_LINE スナップショットを作成できます。

```
CREATE SNAPSHOT sales.order_line AS
  SELECT * FROM sales.order_line@dbs1.acme.com ol
  WHERE EXISTS
    (SELECT o_id FROM sales.orders@dbs1.acme.com o
     WHERE ol.o_id = o.o_id AND EXISTS
      (SELECT c_id FROM sales.customer@dbs1.acme.com c
       WHERE o.c_id = c.c_id AND c.zip = 19555));
```

これらの 3 つの DDL 文で作成したスナップショットは、いずれも高速リフレッシュを実行できます。ターゲットの zip コードで新しい顧客が識別された場合は、後続のリフレッシュ・プロセスで新しいデータがスナップショット・サイトに伝播されます。同様に、ターゲットの zip コードから顧客が削除された場合は、後続のリフレッシュ・プロセスでそのデータもスナップショットから削除されます。

前述のスナップショットの例で示した副問合せは、子表から親表への多対 1 参照を走査します。各スナップショットの定義問合せを満たすデータのみがスナップショットに移入され、これらの定義問合せを満たすデータのみのリフレッシュが実行されます。

割当て表の使用

これまでに説明した副問合せの例ではスナップショットの柔軟性を示してきましたが、これらの例にはいくつかの制限事項もあります。たとえば、営業担当員の担当地域が変更になったか、あるいは既存の地域に別の zip コードが割り当てられた場合、前述のスナップショットの定義では zip コード 19555 がハードコードされているため、スナップショットの定義を変更または再作成する必要があります。

このことを念頭において、副問合せに基づくデータのサブセット化と割当て表を併用すれば、DBA はスナップショット環境の変更を容易に制御できます。たとえば、図 3-4 に示す顧客と営業担当員の関連付けを考えてみます。

この例では、営業担当員は ASSIGNMENT 表に基づいて顧客に割り当てられています。営業担当員の採用や退職があった場合は、割当て表の内容を変更すれば既存の顧客を新しい営業担当員に割り当てることができます。副問合せに基づくデータのサブセット化と割当て表を併用することによって、一元管理が可能になるのみでなく、安全性の高い管理が容易にできるようになります。たとえば、営業担当員 #1001 は他の営業担当員の顧客情報を参照できません。顧客情報に機密データが含まれる場合、このことは非常に重要です。

図 3-4 顧客と営業担当員の関連付け

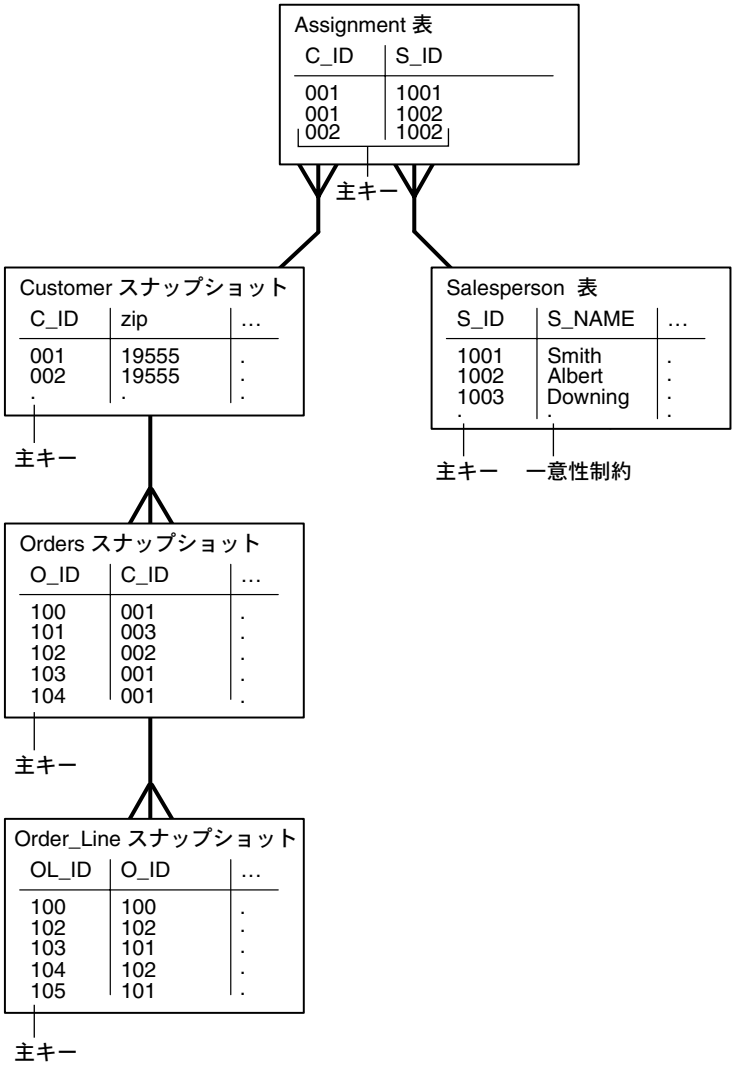


図 3-4 に示す関連付けにおいて、ORDRS スナップショットの定義副問合せが次のように指定されているとします。

```
CREATE SNAPSHOT sales.orders AS
 SELECT * FROM sales.orders@hq.acme.com o
    -- conditions for customers
  WHERE EXISTS
  ( SELECT c_id FROM sales.customer@hq.acme.com c
    WHERE o.c_id = c.c_id
      AND EXISTS
    ( SELECT * FROM sales.assignments@hq.acme.com a
     WHERE a.c_id = c.c_id
     AND EXISTS
     ( SELECT * FROM sales.salesperson@hq.acme.com s
       WHERE a.s_id = s.s_id AND s.s_id = 'gsmith')));
```

この場合、ORDERS スナップショットには営業担当員 'gsmith' に割り当てられた顧客の受注データが移入されます。CREATE SNAPSHOT 文の最後の行の 'gsmith' 値に注意してください。

このような柔軟性があるため、管理者は割当て表に簡単な変更を加えて、DBA に SQL の修正を依頼せずにスナップショットのデータ・セットを容易に制御できます。たとえば、ある営業担当員に 2 人の新しい顧客が割り当てられた場合、管理者は割当て表でその営業担当員に新しい顧客を割り当てるのみです。次回のスナップショットの高速リフレッシュ後、2 人の顧客のデータは営業担当員のラップトップなどのターゲット・スナップショット・サイトに伝播されます。逆に、ある営業担当員の顧客の 1 人を削除した場合は、次回リフレッシュ後にその顧客に関するすべてのデータがスナップショット・サイトから削除され、営業担当員はその情報にアクセスできなくなります。

関連項目： 詳細は、3-28 ページの「リフレッシュ・タイプ」を参照してください。

副問合せを使用するスナップショットに関する制限事項

副問合せを使用するスナップショットは、主キー・タイプのスナップショットである必要があります。この他にも、副問合せを使用するスナップショットの定義問合せには、スナップショットの高速リフレッシュ機能を保つためのいくつかの制限事項があります。

関連項目： 主キー・スナップショットの詳細は、3-5 ページの「主キー」を参照してください。

注意： スナップショットの副問合せが表 3-1 の多くの制限事項を満たしているかどうかを判別するには、高速リフレッシュを使用するスナップショットを作成します。単純副問合せスナップショットに関する制限事項に違反している場合は、エラーが戻されます。強制リフレッシュを指定した場合は、高速リフレッシュが実行できなければ完全リフレッシュが自動的に実行されるため、エラーは戻されません。

表 3-1 副問合せを使用するスナップショットに関する制限事項

- 副問合せでは、次の場合に子表から親表への多対 1 参照を走査できます。

  各親表で参照される結合列に、PRIMARY KEY 制約または UNIQUE 制約がある場合。

  結合式で、完全一致または等価比較（等価結合）を使用する場合。

- 副問合せでは、次の場合にも多対多参照を横断することができます。

  交差部（割当て）表の結合列の両方のセットを含む PRIMARY KEY 制約または UNIQUE 制約が存在する場合。

  交差部表が結合している表の結合列に PRIMARY KEY 制約または UNIQUE 制約があり、かつ、フィルタ列に別の UNIQUE 制約がある場合。交差部表しかない場合は、その交差部表の結合列とフィルタ列の両方を含む PRIMARY KEY 制約または UNIQUE 制約が必要です。

  副問合せで、完全一致または等価比較が指定される場合。

  注意 : これら 2 つのプロパティの組合せは、多対多参照で 1 行のみが戻されることを意味します。詳細は、3-11 ページの「割当て表の使用」を参照してください。ASSIGNMENT 表の PRIMARY KEY 制約には、C_ID 結合列と S_ID 結合列の両方が含まれます。SALESPERSON 表の S_ID 結合列には PRIMARY KEY 制約があり、S_NAME フィルタ列には UNIQUE 制約があります。副問合せでは、S_NAME 内の値が定数（この場合は 'gsmith'）と等しいことを指定しています。

- スナップショットは、主キー・スナップショットである必要があります。

- マスター表のスナップショット・ログには、副問合せで参照されるすべてのフィルタ列が含まれている必要があります。フィルタ列の詳細は、3-37 ページの「フィルタ列の使用」を参照してください。

- 副問合せは、肯定副問合せであることが必要です。たとえば、EXISTS は使用できますが、NOT EXISTS は使用できません。

- 副問合せは、EXISTS を使用して各ネスト・レベルを接続する必要があります（IN は使用できません）。

- 各ネスト・レベルの各表に、主キーが存在している必要があります。

- 各ネスト・レベルでは、上位のレベルの表のみを参照できます。

- 副問合せに AND 式を組込むことができますが、各 OR 式は 1 行に含まれている列のみを参照できます。1 つの副問合せ内の複数の OR 式は、AND 処理されます。

- 各表は、副問合せ内で 1 回に限り結合できます。各表は 1 つの EXISTS 式のみに存在できます。

- 副問合せで参照される各表は、同じマスター・データベースに存在している必要があります。

- 選択された列数と副問合せで使用される各表の主キー列の数の合計は、表で認められている最大列数（1000 列）より少ないことが必要です。

マスター・サイトでのスナップショットの登録

マスター・サイトでは、マスター表をベースとするスナップショットに関する情報が Oracle データベースに自動的に登録されます。次の項では、スナップショットの登録メカニズムについて説明します。

登録されたスナップショットに関する情報の表示

DBA_REGISTERED_SNAPSHOTS データ・ディクショナリ・ビューに問い合せて、リモート・スナップショットに関する次の情報を表示することができます。

- 所有者、名前およびスナップショットを含むデータベース
- スナップショットの定義問合せ
- リフレッシュ方法（高速または完全）など、その他のスナップショットの特性

DBA_SNAPSHOT_REFRESH_TIMES ビューをマスター・サイトで問い合せて、各スナップショットの最後のリフレッシュ時刻を取得することもできます。管理者は、この情報を使用してマスター・サイトからスナップショット・アクティビティを監視し、マスター表の削除、変更または再配置が必要な場合にスナップショット・サイトへの変更を調整できます。

内部メカニズム

スナップショットを作成するとそのマスター・データベースにスナップショットが自動的に登録され、スナップショットを削除すると登録が解除されます。

注意： スナップショットの作成時または削除時は、マスター・サイトでのスナップショットの登録または登録解除は保証されません。スナップショットの作成時にスナップショットが正常に登録されなかった場合は、次回のスナップショットのリフレッシュ時にその登録が実行されます。スナップショットの削除時にスナップショットが正常に登録解除されなかった場合は、手動で登録解除するまで登録情報はマスター・データベースに残ります。複合スナップショットは、登録できない場合があります。

注意： Oracle7 のマスター・サイトでは、スナップショットは登録できません。

手動によるスナップショットの登録

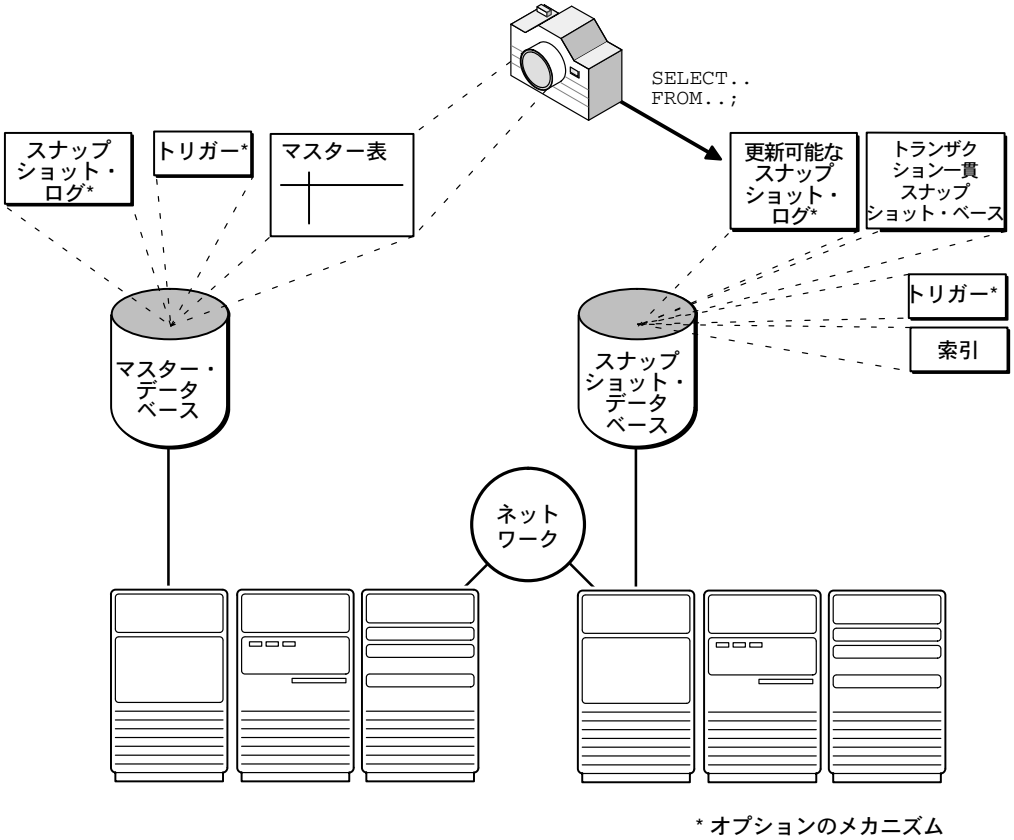
必要な場合は、登録を手動でメンテナンスできます。マスター・サイトで DBMS_SNAPSHOT パッケージの REGISTER_SNAPSHOT プロシージャおよび UNREGISTER_SNAPSHOT プロシージャを使用して、スナップショットの登録情報を追加、変更または削除します。

> **関連項目：** REGISTER_SNAPSHOT プロシージャおよび UNREGISTER_SNAPSHOT プロシージャの詳細は、『Oracle8*i* レプリケーション・マネージメント API リファレンス』を参照してください。

スナップショットのアーキテクチャ

図 3-5 は、スナップショット・レプリケーションに使用されるメカニズムを示します。これらのメカニズムの一部はオプションで、作成したスナップショット環境のサポートに必要な場合にのみ使用します。たとえば、読取り専用スナップショットを使用する場合は、リモート・サイトの更新可能スナップショット・ログや内部トリガーは使用できません。また、高速リフレッシュを実行できない複合スナップショットを使用する場合は、マスター・サイトのスナップショット・ログは使用できません。

図 3-5 スナップショット・レプリケーションのメカニズム



マスター・サイトのメカニズム

図 3-6 に示す 3 つのメカニズムは、マスター・サイトでスナップショットの高速リフレッシュをサポートするために必要なものです。

図 3-6 マスター・サイトのメカニズム

マスター表

マスター表はスナップショットの基礎となるもので、ターゲット・マスター・サイトにあります。この表は、スナップショット・レプリケーションとマルチマスター・レプリケーションの両方に必要です（前述のように、1 つのスナップショットは 1 つのマスター・サイトにリンクされています）。

スナップショット・ログに記録されたマスター表への変更は、リフレッシュ・プロセスの実行時にスナップショットに伝播されます。

注意： スナップショットは、表をベースにして作成します。シノニムまたはビューをベースにして作成することはできません。

内部トリガー

DML を使用してマスター表が変更されると、内部トリガーによって、影響を受けた行に関する情報がスナップショット・ログに記録されます。記録される情報は、主キー値または ROWID（あるいはその両方）、およびフィルタ列の値です。これは、ターゲット・マスター表のスナップショット・ログを作成するときに自動的にアクティブになる内部トリガーです。

注意： スナップショットに副問合せが含まれているときは、フィルタ列が必要です。副問合せスナップショットの詳細は、3-8 ページの「スナップショットでのデータのサブセット化」を参照してください。フィルタ列の詳細は、3-37 ページの「フィルタ列の使用」を参照してください。

スナップショット・ログ

マスター表のスナップショット・ログを作成すると、スナップショット・ログの基礎となる表が作成されます。スナップショット・ログには、主キーまたはマスター表で更新された行の ROWID（あるいはその両方）が含まれます。スナップショット・ログには、副問合せを使用するスナップショットの高速リフレッシュをサポートするために、フィルタ列を含めることもできます。スナップショット・ログの表の名前は、*MLOG$_master_table_name* です。スナップショット・ログは、ターゲット・マスター表と同じスキーマで作成されます。1 つのスナップショット・ログで、マスター表の複数のスナップショットをサポートできます。

前の項で説明したように、ターゲット・マスター表で DML トランザクションが開始されたときは常に、内部トリガーによってスナップショット・ログに変更情報が追加されます。

スナップショット・ログには次の 3 つのタイプがあります。

- **主キー** : マスター表が変更されると、影響を受けた行の主キーに基づいて変更内容が記録されます。
- **ROWID**: マスター表が変更されると、影響を受けた行の ROWID に基づいて変更内容が記録されます。
- **組合せ** : マスター表が変更されると、影響を受けた行の主キーおよび ROWID の両方に基づいて変更内容が記録されます。このスナップショット・ログは、主キー・スナップショットおよび ROWID スナップショットの両方をサポートしています。これは、混在環境の場合に便利です。

組合せスナップショット・ログは主キー・スナップショット・ログおよび ROWID スナップショット・ログと同様に機能しますが、変更された行の主キーと ROWID の両方が記録される点が異なります。

主キーまたは ROWID に基づくこれらのスナップショット・ログの間に大きな違いはありませんが（影響を受けた行を記録するのに、主キーと物理 ROWID のどちらを使用するかという点のみが異なります）、実際に与える影響は非常に大きくなります。ROWID スナップショットとスナップショット・ログを使用すると、マスター表の再編成および切捨てが難しくなります。これは、ROWID スナップショットを高速リフレッシュできなくなるためです。マスター表の再編成または切捨てを実行した場合は、マスター表の ROWID が変更されるため、ROWID スナップショットを完全リフレッシュする必要があります。

異なるスキーマへのスナップショットおよびスナップショット・ログのインポート スナップショットとスナップショット・ログは、DDL 文で明示的に指定されたスキーマ名でエクスポートされます。したがって、スナップショットとスナップショット・ログをエクスポート元のスキーマと異なるスキーマにインポートすることはできません。
FROMUSER/TOUSER インポート・オプションを使用してスナップショットまたはスナップショット・ログを含むエクスポート・ダンプ・ファイルをインポートしようとすると、インポート・ログ・ファイルにエラーが書き込まれ、スナップショットとスナップショット・ログはインポートされません。

スナップショット・サイトのメカニズム

スナップショットを作成すると、スナップショットをサポートするために、いくつかのメカニズムがスナップショット・サイトで作成されます。具体的には、1 つのベース表、1 つ以上の索引、および場合によっては 1 つのビューが作成されます。更新可能なスナップショットを作成する場合は、スナップショット・サイトに内部トリガーとローカル・ログ（更新可能なスナップショット・ログ）も作成されます。

ベース表

Oracle8*i* リリース 8.1.5 から、ベース表は実スナップショットとなりました（ビューは必要ありません）。ベース表には、スナップショットの作成時に指定した名前が付いています。

スナップショット・サイトの互換性設定がリリース 8.1.0 以前の場合、ベース表がビューの基礎サポート・メカニズムとなります。互換性設定は、初期化パラメータ・ファイルの COMPATIBLE 初期化パラメータによって定義されます。ベース表がビューのサポート・メカニズムとなっている場合は、ベース表の名前は SNAP$_*Snapshot_Name* です。スナップショットの作成時に生成される索引は、ベース表に作成されます。

注意： SNAP$_*Snapshot_Name* が 32 文字を超える場合、表名が切り捨てられ、順序番号が追加されます。

関連項目： COMPATIBLE の詳細は、7-3 ページの「初期化パラメータ」および『 Oracle8*i* 移行ガイド 』を参照してください。8.1.0 より前のバージョンと互換性を持つスナップショットをサポートして作成されるビューの詳細は、3-22 ページの「ビュー」を参照してください。

スナップショットのデータ型に関する考慮事項

Oracle では、NUMBER 型、DATE 型、VARCHAR2 型、CHAR 型、NVARCHAR2 型、NCHAR 型、RAW 型および ROWID 型を使用するマスター表列のスナップショットをサポートしています。

また、バイナリ型 LOB（BLOB）、文字型 LOB（CLOB）および各国文字型 LOB（NCLOB）というラージ・オブジェクト型を使用するマスター表列のスナップショットもサポートしています。ただし、スナップショットの定義問合せの WHERE 句で LOB 列は参照できません。部分単位の更新と追加がこれらの LOB 列に適用されると、レプリケーションに使用する遅延および同期のリモート・プロシージャ・コール・メカニズムによって、サポートされている LOB データ型に部分単位の変更のみが伝播されます。

注意： Oracle8*i* は、一部のサイトで Oracle7 リリース 7.3 が稼動しているレプリケーション環境では、LOB データ型のレプリケーションをサポートしていません。

Oracle では、LONG データ型を使用する列のレプリケーションはサポートしていません。LONG データ型を含む列のデータはスナップショットから単純に削除されます。

また、ユーザー定義のオブジェクト型、外部 LOB やファイル・ベース LOB（BFILE）もサポートしていません。これらのデータ型の列を含むスナップショットを構成しようとすると、エラー・メッセージが戻されます。

注意： マスター表の列にデフォルト値が定義されていても、マスター表をベースとする更新可能スナップショットがそのデフォルト値を自動的に使用するわけではありません。

注意： 更新可能スナップショットは、DELETE CASCADE 制約をサポートしていません。

ビュー

ビューは、Oracle のリリースが 8.0 以前の場合、またはリリース 8.1 のスナップショット・サイトの互換性設定が 8.1.0 より前の場合に、スナップショット・レプリケーションをサポートするために作成されます。ビューが作成されると、そのビューには CREATE SNAPSHOT 文で指定されているのと同じ名前が付きます。たとえば、CREATE SNAPSHOT SALES.SNAP_CUSTOMER AS .... 文によって作成されるビューの名前は SNAP_CUSTOMER となります。

索引

リモート・スナップショット・サイトでは、各主キー・スナップショットに少なくとも 1 つの索引が作成されます。この索引は、ターゲット・マスター表の主キーと対応しており、名前は I_SNAP$_*Snapshot_Name* になります。副問合せを使用するスナップショットの高速リフレッシュをサポートするために、リモート・スナップショット・サイトで追加の索引が作成されることもあります。

注意： I_SNAP$_*Snapshot_Name* が 32 文字を超える場合、表名が切り捨てられ、順序番号が追加されます。

更新可能スナップショット・ログ

更新可能スナップショット・ログ（USLOG$_*Snapshot_Name*）は、ターゲット・マスター表からの取り出しが必要なデータを判断するために使用します。読取り専用スナップショットでは、このログは作成されません。

内部トリガー

マスター・サイトの内部トリガーと同様、スナップショット・サイトの内部トリガーによって、更新可能なスナップショットに加えられた DML 変更が USLOG$_*Snapshot_Name* ログに記録されます。

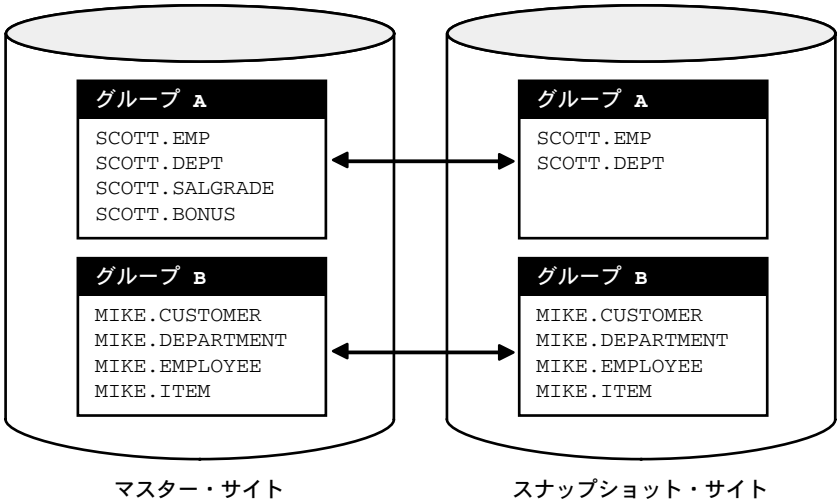
編成メカニズム

前の項で説明したスナップショットのメカニズムの他にも、スナップショット・サイトにはスナップショットを編成するいくつかのメカニズムがあります。このメカニズムによって、スナップショット・サイトとマスター・サイト間の編成上の整合性、およびターゲット・マスター・グループとのトランザクション（読込み）一貫性が保たれます。そのメカニズムは、スナップショット・グループとリフレッシュ・グループです。

スナップショット・グループ

レプリケーション・システム内のスナップショット・グループでは、ターゲット・マスター・グループにあるオブジェクトの一部または完全なコピーがメンテナンスされます。スナップショット・グループは、マスター・グループの境界をまたがって作成することはできません。図 3-7 は、マスター・サイトのグループ A および B と、スナップショット・サイトのグループ A および B との間の相関関係を示しています。

図 3-7 スナップショット・グループとマスター・グループの対応



スナップショット・サイトのグループ A（図 3-7 を参照）には、マスター・サイトの対応するグループ A にあるオブジェクトの一部しか含まれていません。スナップショット・サイトのグループ B には、マスター・サイトのグループ B のオブジェクトがすべて含まれています。しかし、スナップショット・サイトのグループ B にマスター・サイトのグループ A のオブジェクトが含まれることはありません。図 3-7 に示すように、スナップショット・グループの名前は、ベースとなっているマスター・グループの名前と同じです。たとえば、"PERSONNEL" マスター・グループをベースとするスナップショット・グループの名前も "PERSONNEL" になります。

スナップショット・グループは、スナップショット・サイトとマスター・サイト間の編成上の整合性を保つため、および更新可能スナップショットをサポートするために必須です。スナップショットがスナップショット・グループに属していない場合、そのスナップショットは必ず読取り専用スナップショットであることが必要です。

グループ所有者の使用 スナップショット・サイトの同じデータベース内で複数のユーザーをサポートする必要がある場合は、1 つのターゲット・マスター・グループに複数のスナップショット・グループを作成できます。複数のスナップショット・グループを作成すれば、各スナップショット・グループのスナップショット定義に異なる副問合せを定義でき、各ユーザーが自分用のデータのサブセットのみにアクセスできます。

別々のスナップショット・グループに複数のデータ・セットを定義する方法は、別々のユーザーをサポートする複数のビューに別々の WHERE 句を定義する方法よりも安全です。別々のスナップショット・グループに複数のデータ・セットを定義すると、ユーザーに対して個々のスナップショット・オブジェクトへのアクセス権限を付与でき、ユーザーによる表

示、削除および挿入を制御できます。ビューで WHERE 句を使用すると、ユーザーによる表示は制御できますが、削除または挿入は制御できません。

複数のスナップショット・グループを定義すると、グループ・レベルでデータ・セットを制御できます。たとえば、HR、PERSONNEL および MANUFACTURING の各部門に別々のスナップショット・グループを作成した場合、各部門を個々のオブジェクトではなくグループとして管理できます。それによって、スナップショットを部門グループとしてリフレッシュしたり、オブジェクトをグループとして削除したりできます。

複数のスナップショット・グループを 1 つのマスター・グループに基づく同一のスナップショット・サイトに置くには、スナップショット・グループの定義時にグループ所有者を追加の識別子として指定します。

グループ所有者を指定してスナップショット・グループを定義した後、同じグループ所有者を定義してスナップショット・オブジェクトをターゲット・スナップショット・グループに追加します。グループ所有者を使用する場合は、各スナップショット・オブジェクトが一意の名前を持つ必要があります。1 つのスナップショット・サイトに同じマスター・グループをベースとする複数のスナップショット・グループがある場合、スナップショット・グループのオブジェクトの名前を別のスナップショット・グループのスナップショット・オブジェクトと同じ名前にすることはできません。名前の競合を避けるには、オブジェクト名の最後にグループ所有者を追加します。たとえば、グループ所有者が "HR" と "PERSONNEL" ならば、"EMP" スナップショット・オブジェクトの名前をそれぞれ "EMP_HR" と "EMP_PERSONNEL" にします。

また、1 つのスナップショット・サイト内の同じマスター・グループをベースとするスナップショット・グループはすべて、同じマスター・サイトにリンクされている必要があります。たとえば、所有者が HR の SCOTT_MG スナップショット・グループが ORC1.WORLD マスター・サイトの対応するマスター・グループをベースとしており、所有者が HR のグループと所有者が PERSONNEL のグループが同じスナップショット・サイトにある場合は、所有者が PERSONNEL の SCOTT_MG スナップショット・グループも ORC1.WORLD の対応するマスター・グループをベースとしている必要があります。

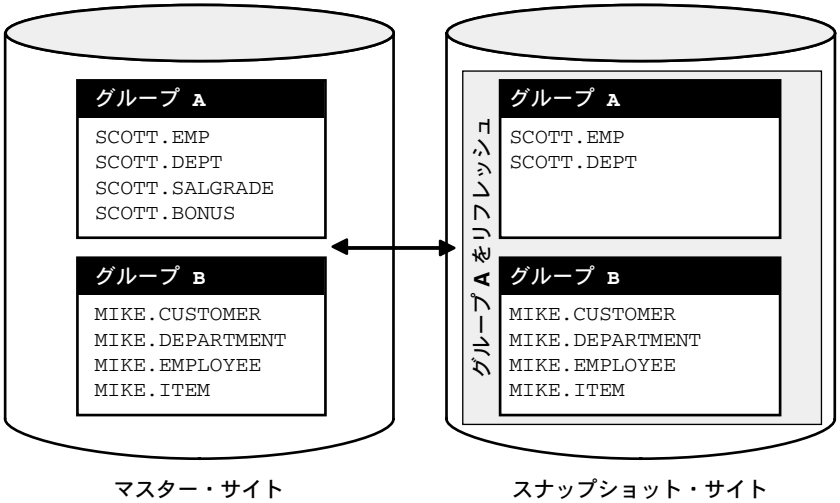
> **関連項目：** レプリケーション・マネージメント API を使用してグループ所有者を定義する方法の詳細は、『Oracle8*i* レプリケーション・マネージメント API リファレンス』の第 7 章の「グループの所有者の使用」の項を参照してください。

リフレッシュ・グループ

複数のスナップショット間の参照整合性およびトランザクション（読込み）一貫性を保つために、Oracle では、個々のスナップショットをリフレッシュ・グループの一部分としてリフレッシュできます。リフレッシュ・グループ内のすべてのスナップショットのリフレッシュが終了すると、グループ内のすべてのスナップショットのデータはトランザクション一の貫性を保つ特定の時点のデータに対応します。

図 3-8 に示すように、リフレッシュ・グループには複数のスナップショット・グループのスナップショットを含めることができるので、マスター・グループ境界にまたがってトランザクション（読込み）一貫性を保つことができます。

図 3-8 複数のスナップショット・グループのオブジェクトを含むリフレッシュグループ



スナップショット・グループごとに 1 つのリフレッシュ・グループを定義するよりも、複数のスナップショット・グループのオブジェクトを含む 1 つの大きなリフレッシュ・グループを使用するほうが効率的な場合があります。このように構成すると、スナップショットのリフレッシュに必要なオーバーヘッドの量が削減されます。リフレッシュ・グループには、最大で 400 のスナップショットを含めることができます。この数は、Oracle Server の前のバージョンよりも多くなっています。

1 つのスナップショット・グループの内容をリフレッシュするために複数のリフレッシュ・グループを使用する構成は避けてください。複数のリフレッシュ・グループを使用して 1 つのスナップショット・グループの内容をリフレッシュすると、スナップショット・データの整合性が失われる原因となり、スナップショット・サイトで参照整合性の問題が発生することがあります。このような構成は、データベース環境について十分な知識があり、参照整合性の問題の発生を回避できる場合にのみ使用してください。

リフレッシュ・グループのサイズ

リフレッシュ・グループのサイズを決定するときは、サイズの違いによるメリットとデメリットの両方を考慮する必要があります。Oracle は、大規模リフレッシュ・グループに対応できるように最適化されています。したがって、大規模リフレッシュ・グループと小規模リフレッシュ・グループを比較した場合、グループ内のスナプショットが類似していると想定すると、大規模リフレッシュ・グループのほうが同じ数のスナップショットを高速にリフレッシュできます。たとえば、スナップショットを 20 ずつ含む 5 つのリフレッシュ・グループよりも、100 のスナップショットを含む 1 つのリフレッシュ・グループのほうが高速にリフレッシュできます。また、大規模リフレッシュ・グループを使用すると、1 回のレプリケーション・マネージメント API コールで、より多くのスナップショットをリフレッシュできます。

リフレッシュ・グループのリフレッシュ時に、グループ内の各スナップショットは、スナップショット・サイトでグループ内のすべてのスナップショットのリフレッシュが終了するまでロックされます。リフレッシュ操作中にユーザーがスナップショットを更新するとデータの整合性が損なわれる可能性があるので、これを回避するためにスナップショットのロックは必須です。したがって、リフレッシュ・グループの規模が小さいということは、リフレッシュ実行時のスナップショットのロック時間が短くなることを意味します。

リフレッシュの実行時は、ネットワーク接続を維持する必要があります。リフレッシュ実行時にネットワーク接続が切断または妨害されると、データベースの整合性を保つためにすべての変更がロールバックされます。したがって、ネットワーク接続を維持することが困難な場合は、リフレッシュ・グループの規模を小さくします。

リリース 8.1 では、NULL リフレッシュの最適化はサポートしていません。つまり、特定のスナップショットの最後のリフレッシュ以降にマスター表が変更されなかった場合は、スナップショット・グループのリフレッシュ時にスナップショットがロックされることはほとんどありません。ただし、リリース 8.1 より前のバージョンでは、更新頻度が少ないマスター表のスナップショットを別の独立したリフレッシュ・グループに分けることを検討してください。それによって、更新頻度が高いマスター表のスナップショットを含む他のスナップショット・グループのリフレッシュ時間を短縮できます。

表 3-2 に、大規模リフレッシュ・グループと小規模リフレッシュグループのメリットをまとめます。

表 3-2 大規模リフレッシュ・グループおよび小規模リフレッシュ・グループ

大規模リフレッシュ・グループのメリット	小規模リフレッシュ・グループのメリット
■ 同数のスナップショットを複数のリフレッシュ・グループに分けてリフレッシュするよりも高速にリフレッシュできます。	■ スナップショットのロック時間が短くなります。
■ 1 回のレプリケーション・マネージメント API コールでリフレッシュできます。	■ ネットワーク接続が失われたことが原因でリフレッシュ実行時に変更がロールバックされる可能性が低くなります。

リフレッシュ・プロセス

スナップショットのデータは、そのマスター表のカレント・データと常に一致している必要はありません。スナップショットは、ある一時点（作成時またはリフレッシュ時）に存在していたデータとして、トランザクション（読込み）一貫性を保ってマスター表を反映したものです。スナップショットのデータを比較的新しいマスター表のデータと一致させるために、スナップショットを定期的にリフレッシュする必要があります。スナップショット・リフレッシュは、より新しい状態のマスター表をスナップショットに反映させる効果的なバッチ操作です。

より新しいマスター・データを反映するために、各スナップショットをいつ、どのようにリフレッシュするかを決定する必要があります。たとえば、アプリケーションで更新されることが多いマスター表のスナップショットは、頻繁にリフレッシュする必要があります。これに対して、比較的変化の少ないマスター表のスナップショットは、あまり頻繁にリフレッシュする必要はありません。スナップショットの適切なリフレッシュ間隔を判断するためには、アプリケーションの特性および要件を分析する必要があります。

スナップショットのリフレッシュでは、リフレッシュの開始方法が異なるいくつかのタイプがサポートされています。

リフレッシュ・タイプ

スナップショットをリフレッシュするには、高速リフレッシュ、完全リフレッシュまたは強制リフレッシュのいずれかを使用できます。

完全リフレッシュ スナップショットの完全リフレッシュを実行するには、スナップショットを管理するサーバーで、スナップショットの定義問合せを実行します。問合せの結果セットで既存のスナップショット・データが置き換えられ、スナップショットがリフレッシュされます。完全リフレッシュはあらゆるスナップショットで実行できます。定義問合せを満たすデータの量によっては、完全リフレッシュの実行に高速リフレッシュの場合よりもかなり長い時間がかかる場合があります。

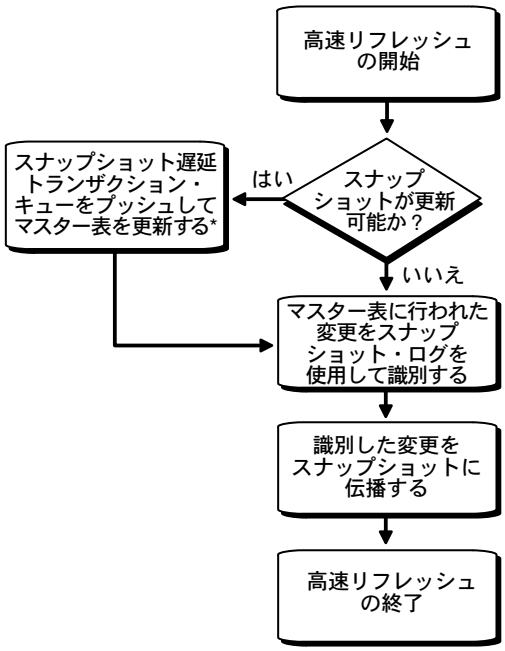
注意： スナップショットを完全リフレッシュする場合は、効率を最大にするために、PCTFREE を 0 に、PCTUSED を 100 に設定してください。

高速リフレッシュ 高速リフレッシュの実行時は、スナップショットを管理しているサーバーで、スナップショットの最新リフレッシュ後にマスター表に加えられた変更が識別され、その変更内容がスナップショットに適用されます。高速リフレッシュは、関係する各サーバーおよびネットワークでレプリケートするデータが少なくて済むため、マスターに加えられた変更が少ないときは完全リフレッシュよりも効率的です。スナップショットの高速リフレッシュは、マスター表にスナップショット・ログがある場合にのみ実行できます。

SQL*Loader を使用してマスター表でダイレクト・パス・ロードを行うと、それ以後は高速リフレッシュを実行してもダイレクト・パス・ロード中に発生した変更は適用されません。また、マスター表でその他のバルクロード操作を行った結果加えられた変更も適用されませ

ん。たとえば、APPEND ヒントを含む INSERT 文や、INSERT ... SELECT * FROM 文がこれにあたります。

図 3-9 スナップショットの高速リフレッシュ



* このステップは、DBMS_DEFER_SYS.PUSH ファンクションを使って別個に実行することもできます。

強制リフレッシュ スナップショットの強制リフレッシュを実行するためには、スナップショットを管理するサーバーは高速リフレッシュの実行を試みます。高速リフレッシュが実行できない場合は、完全リフレッシュが実行されます。高速リフレッシュが実行できない場合にスナップショットのリフレッシュを実行するには、強制リフレッシュの設定を使用します。

リフレッシュの開始

管理者は、リフレッシュ・グループを作成するときに、スケジュールした間隔でスナップショットを自動的にリフレッシュするようにグループを構成できます。逆に、スケジューリング情報を設定せずに、リフレッシュ・グループを手動でまたは必要に応じてリフレッシュするようにグループを構成することもできます。ダイヤルアップ・ネットワーク接続を使用してリフレッシュを実行する場合は、手動リフレッシュは理想的なソリューションです。

スケジュールしたリフレッシュ スケジュールに従ってリフレッシュするリフレッシュ・グループを作成するには、作成プロセスで、スケジュールしたリフレッシュ間隔をグループに指定する必要があります。グループのリフレッシュ間隔を設定するときは、次の特性を考慮してください。

- リフレッシュ間隔を指定する日付または日付式の値は、将来の特定の時点である必要があります。
- リフレッシュ間隔は、リフレッシュの実行に必要な時間よりも長くする必要があります。
- 相対日付式の値は、最新のリフレッシュ日付から計算される特定の時点である必要があります。ネットワークまたはシステムの障害により、グループのリフレッシュがスケジュールどおりに実行されない場合、相対日付式の結果は状況に応じて変更されることがあります。
- 明示日付式の値は、最新のリフレッシュ日付に関係なく特定の時点である必要があります。
- 古いデータに対する環境の許容度を考慮してください。許容度が低い場合はリフレッシュ間隔を小さくし、許容度が高い場合はリフレッシュ間隔を大きくします。

必要に応じたリフレッシュ 環境や状況によっては、スケジュールしたスナップショット・リフレッシュの実行が適切でない場合もあります。たとえば、マスター表にデータをバルクロードした直後は、依存スナップショットにマスター表のデータは反映されていません。次に実行予定の自動グループ・リフレッシュを待たずに、対応付けられたスナップショットにマスター表の新規行をすぐに伝播するために、依存スナップショット・グループを手動でリフレッシュすることもできます。

スナップショットが非接続環境のラップトップの営業支援システムと統合されている場合は、スナップショットを必要に応じてリフレッシュすることもできます。営業支援ソフトウェアを設計する開発者がアプリケーション・コントロール（ボタンなど）を作成します。営業担当員は、1 日の受注をサーバーに転送するときに、ダイヤルアップ・ネットワーク接続を確立し、アプリケーション・コントロールを使用してスナップショットをリフレッシュできます。

スナップショットの準備

スナップショット・レプリケーションに関する問題の多くは、環境の準備が正しく行われていないことによるものです。スナップショット環境を作成するには、事前に次の 4 つのことを行う必要があります。

- 必要なスキーマの作成
- 必要なデータベース・リンクの作成
- 適切な権限の割当て
- 十分なジョブ・プロセスの割当て

Replication Manager の「Setup Wizard」では、前述の作業が自動的に実行されます。以降では、DBA がレプリケーション環境について理解するのに役立つ情報、およびユーザーがレプリケーション・マネージメント API を使用する際に役立つ情報について説明します。「Setup Wizard」を実行した後で、必要なスナップショット・ログを作成します。Replication Manager を使用してスナップショット・サイトを設定する方法の詳細は、Replication Manager のオンライン・ヘルプを参照してください。できる限り、Replication Manager を使用することをお薦めします。

関連項目： 3-37 ページの「スナップショット・ログの作成」

Replication Manager を使用できない場合は、『Oracle8*i* レプリケーション・マネージメント API リファレンス』の第 2 章の「スナップショット・サイトの設定」の項の、レプリケーション・マネージメント API を使用したスナップショット・サイトの設定方法の詳細を参照してください。

次の項では、Replication Manager の「Setup Wizard」または『Oracle8*i* レプリケーション・マネージメント API リファレンス』に記述されているスクリプトを使用してスナップショット・サイトを設定する方法について説明します。

スナップショット・サイト・ユーザーの作成

各スナップショットには、スナップショット・サイトで管理アクティビティやリフレッシュ・アクティビティを実行する何人かのユーザーが必要です。スナップショットの管理者とリフレッシャには、必要な権限を作成して付与する必要があります。

マスター・サイト・ユーザーの作成

スナップショット・サイトのユーザーのかわりに作業を実行するには、ターゲット・マスター・サイトにも同等のプロキシ・ユーザーが必要です。通常は、プロキシ・スナップショット管理者とプロキシ・リフレッシャを作成します。

スキーマ

リモート・データベースのスナップショットを含むスキーマは、マスター・データベースのマスター表を含むスキーマに対応している必要があります。したがって、スナップショットを使用してレプリケートするマスター表を含むスキーマを識別してください。マスター・データベースのターゲット・スキーマを識別した後、リモート・データベースで、対応するアカウントを同じ名前で作成します。たとえば、すべてのマスター表が DB1 データベースの SALES スキーマ内にある場合、対応する SALES スキーマをスナップショット・データベース DB2 に作成します。

> **関連項目：**『Oracle8*i* レプリケーション・マネージメント API リファレンス』では、第 5 章「スナップショット・グループの作成」で説明しているスクリプトの一部として、必要なスキーマを作成しています。

データベース・リンク

スナップショットの定義問合せでは、1 つ以上のデータベース・リンクを使用してリモート表のデータを参照します。スナップショットを作成するには、使用する予定のデータベース・リンクが使用可能になっている必要があります。さらに、リモート・データベースにアクセスするためにデータベース・リンクで使用するアカウントには、セキュリティ・コンテキストが定義されます。そのコンテキストに基づいてスナップショットが作成され、リフレッシュが実行されます。

正常な動作を保証するために、スナップショットの定義問合せでは、ユーザー名とパスワードが定義に埋め込まれているデータベース・リンクを使用する必要があります。スナップショットを作成するときは、パブリック・データベース・リンクを使用できません。ユーザー名とパスワードが埋め込まれているデータベース・リンクは常に、指定されたアカウントを使用してリモート・データベースとの接続を確立します。また、リンクで使用するリモート・アカウントには、スナップショットの定義問合せの中で参照されるデータにアクセスするために必要な SELECT 権限が必要です。

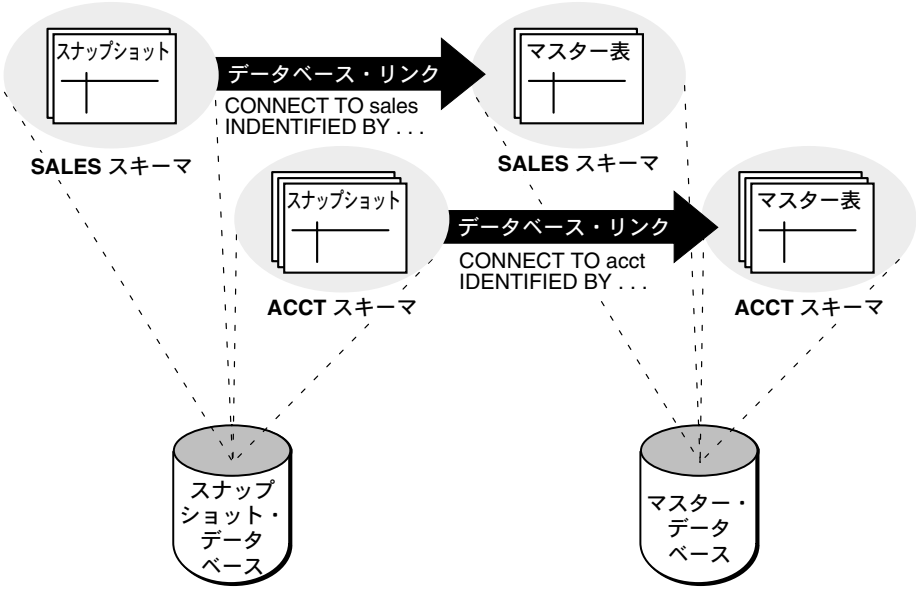
スナップショットを作成するには、いくつかの管理データベース・リンクを作成する必要があります。具体的には、スナップショット・サイトからマスター・サイトへの PUBLIC データベース・リンクを作成する必要があります。各リンクには USING 句を含める必要がないため、これによってプライベート・データベース・リンクの定義が容易になります。また、スナップショット管理者からプロキシ管理者へのプライベート・リンクおよびプロパゲータから受信者へのプライベート・リンクも作成する必要があります。ただし、これらのデータベース・リンクは、Replication Manager の「Setup Wizard」を使用すると自動的に作成されます。

> **関連項目：** 詳細は、『Oracle8*i* レプリケーション・マネージメント API リファレンス』の付録 A「セキュリティ・オプション」を参照してください。

管理データベース・リンクを作成した後、スナップショット・データベースの各レプリケート・スナップショット・スキーマをマスター・データベースの対応するスキーマに接続する

プライベート・データベース・リンクを作成する必要があります。スナップショット・データベースの各プライベート・データベース・リンクには、対応付けられたマスター・データベースのアカウント情報を埋め込むことが必要です。たとえば、スナップショット・データベース DB2 の SALES スキーマには、SALES というユーザー名とパスワードを使用して接続するプライベート・データベース・リンク DB1 が必要です。

図 3-10 お薦めするスキーマおよびデータベース・リンク構成



権限

スナップショットの作成者と所有者は、ともにスナップショットの定義 SELECT 文を発行できる必要があります。スナップショットの所有者とは、スナップショットが含まれているスキーマのことです。レプリケーション管理者またはスナップショット管理者以外のユーザーがスナップショットを作成する場合は、そのユーザーに CREATE SNAPSHOT 権限および定義 SELECT 文を実行する適切な SELECT 権限が付与されている必要があります。

> **関連項目：** 『Oracle8*i* レプリケーション・マネージメント API リファレンス』では、第 5 章「スナップショット・グループの作成」で説明しているスクリプトの一部として、必要な権限を付与しています。

マスター・サイトでのスケジュール・パージ

遅延トランザクション・キューのサイズを適正に保つために、パージ操作をスケジュールして、正常に送信されたすべての遅延トランザクションを遅延トランザクション・キューから削除する必要があります。この操作は、マスター・サイトで実行済みの場合もあります。パージ操作を再スケジュールしてもマスター・サイトに影響はありませんが、パージのスケジューリング特性が変更される可能性があります。

プッシュのスケジュール

スナップショット・サイトでプッシュをスケジュールすると、スナップショット・サイトの遅延トランザクションが対応付けられたターゲット・マスター・サイトへ自動的に伝播されます。これは、スケジュール・リンクとも呼ばれます。スナップショット・グループは 1 つのターゲット・マスター・サイトのみを持つので、通常、スナップショット・サイトの各スナップショット・グループには 1 つのスケジュール・リンクのみが存在します。

SNP バックグラウンド・プロセスおよび間隔

スナップショットの自動リフレッシュを処理するには、十分な数の SNP（またはジョブ・キュー）バックグラウンド・プロセスを割り当てることが重要です。通常、スナップショット・サイトからターゲット・マスター・サイトにはスケジュール・リンクのみが存在するため、スナップショット・サイトでは 1 つの SNP プロセスしか必要ありません。ただし、スケジュール・ジョブなどの追加のアクティビティを処理するには、スナップショット・サイトに少なくとも 2 つの SNP プロセスを割り当てる必要があります。また、それぞれのパラレル化の程度に合わせて少なくとも 1 つの SNP プロセスが必要です。

SNP プロセスは、データベースの初期化パラメータ・ファイルにある JOB_QUEUE_PROCESSES 初期化パラメータを使用して定義されます。SNP プロセスを設定するには、Instance Manager（Oracle Enterprise Manager のコンポーネント）を使用するか、または初期化パラメータ・ファイルを手動で編集します。

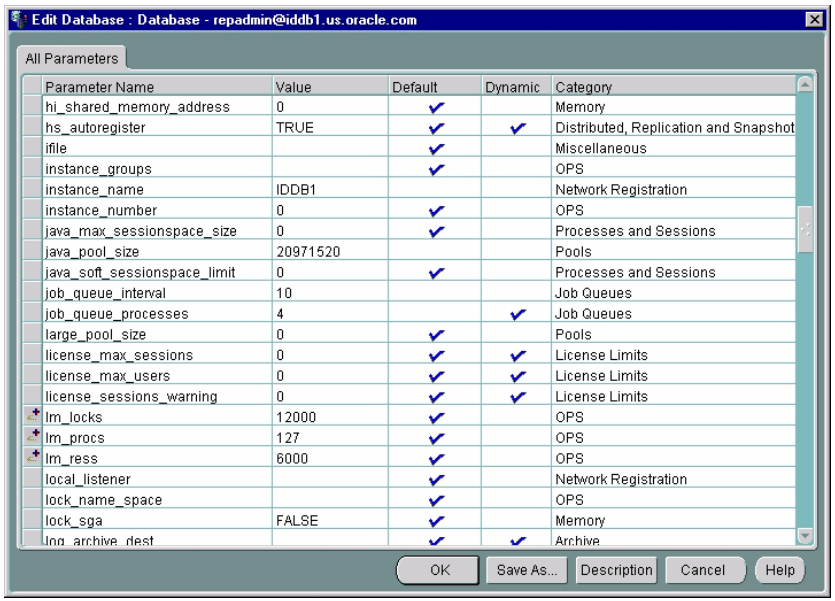
SNP ジョブ間隔によって、保留中の操作（キューのプッシュなど）を実行するために SNP プロセスを起動する間隔が決まります。一般的なレプリケート環境ではデフォルト値の 60 秒が適当ですが、要件によってはパフォーマンスを最大にするためにこの値を調整する必要があります。たとえば、ターゲット・マスター・サイトに 20 秒間隔で変更を伝播する場合は、60 秒のジョブ間隔では長すぎます。これに対し、変更を 1 日に 1 回伝播する場合は、SNP プロセスが保留中の操作を検査するのは 1 時間に 1 回のみで十分ということもあります。

関連項目： JOB_QUEUE_PROCESSES の詳細は、7-3 ページの「初期化パラメータ」および『Oracle8*i* リファレンス・マニュアル』を参照してください。

Instance Manager

スナップショット・サイトへの専用ネットワーク・リンクがある場合や、ネットワーク・リンクをスケジュールできる場合は、通常、Instance Manager の「Edit Database」ダイアログ・ボックスを使用してスナップショット・サイトの SNP プロセスおよび SNP ジョブ間隔を設定します。Instance Manager はスナップショット・サイトにインストールされることはほとんどなく、マスター・サイトからリモートで設定する必要があるため、この設定は必須です。リモートで設定できない場合は、次の項を参照してください。

図 3-11 Instance Manager を使用したジョブ・プロセスの値の設定



Parameter Name	Value	Default	Dynamic	Category
hi_shared_memory_address	0	✔		Memory
hs_autoregister	TRUE	✔	✔	Distributed, Replication and Snapshot
ifile		✔		Miscellaneous
instance_groups		✔		OPS
instance_name	IDDB1			Network Registration
instance_number	0	✔		OPS
java_max_sessionspace_size	0	✔		Processes and Sessions
java_pool_size	20971520			Pools
java_soft_sessionspace_limit	0	✔		Processes and Sessions
job_queue_interval	10			Job Queues
job_queue_processes	4		✔	Job Queues
large_pool_size	0	✔		Pools
license_max_sessions	0	✔	✔	License Limits
license_max_users	0	✔	✔	License Limits
license_sessions_warning	0	✔	✔	License Limits
lm_locks	12000	✔		OPS
lm_procs	127	✔		OPS
lm_ress	6000	✔		OPS
local_listener		✔		Network Registration
lock_name_space		✔		OPS
lock_sga	FALSE	✔		Memory
log_archive_dest		✔	✔	Archive

Instance Manager を使用してジョブ・プロセスを設定するには、次の操作を実行します。

1. Instance Manager を起動し、ターゲット・データベースに接続します。
2. ナビゲータ・シートで「Database」を選択します。
3. 右ペインの「General」タブで、「All Initialization Parameters」ボタンをクリックします。「Edit Database」ダイアログ・ボックスが表示されます。
4. JOB_QUEUE_PROCESS の横の「Value」フィールドに 2（または 2 より大きい値）を入力します。
5. 必要に応じて、JOB_QUEUE_INTERVAL を変更します（入力値の単位は秒です）。
6. 「OK」ボタンをクリックします。

この設定は保存できます。設定を保存すると、Instance Manager を使用してデータベースを管理する場合に便利です。

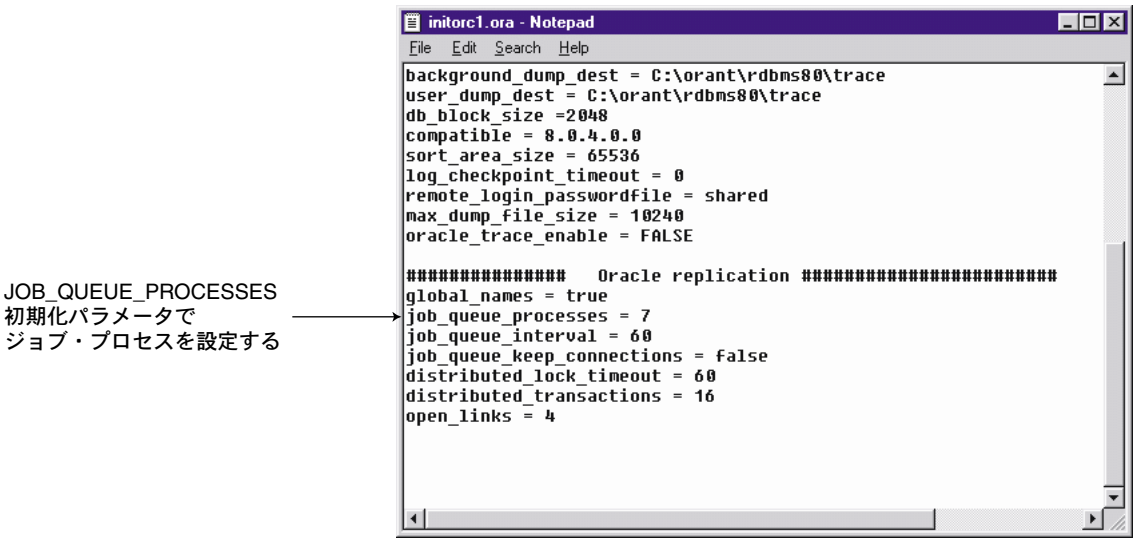
関連項目： Instance Manager の使用方法の詳細は、『Oracle Enterprise Manager 管理者ガイド リリース 2.1 』および Instance Manager のオンライン・ヘルプを参照してください。

初期化パラメータ・ファイルの手動による編集

Instance Manager を使用できない場合は、初期化パラメータを手動で編集する方法もあります。初期化パラメータ・ファイルの内容を変更するには、テキスト・エディタを使用します。

通常、レプリケーションで使用される初期化パラメータはすべて、初期化パラメータ・ファイルの "Oracle replication" という項目の下にまとめられています。

図 3-12 テキスト・エディタを使用した初期化パラメータ・ファイルの編集



初期化パラメータ・ファイルの内容を変更した後で、データベースを再起動して新しい設定を有効にします。

関連項目： Oracle レプリケーションの初期化パラメータの詳細は、『Oracle8*i* リファレンス・マニュアル』の第 1 章の「初期化パラメータ」を参照してください。データベースの再起動の詳細は、『Oracle8*i* 管理者ガイド』を参照してください。

スナップショット・ログの作成

リモート・スナップショット・サイトのスナップショット・グループおよびスナップショットを作成する前に、必要なスナップショット・ログをマスター・サイトに作成しておく必要があります。スナップショット・ログは、スナップショットを 1 つでも高速リフレッシュでサポートしているすべてのマスター表に必要です。

スナップショット・ログを作成するには、次の権限が必要です。

- CREATE ANY TABLE
- CREATE ANY TRIGGER
- SELECT（スナップショット・ログのマスター表に対して必要です）
- COMMENT ANY TABLE

関連項目： Replication Manager を使用してマスター・サイトでスナップショット・ログを作成する方法の詳細は、Replication Manager のオンライン・ヘルプの「Creating Snapshot Logs」のトピックを参照してください。このトピックを参照するには、「Help」の「Contents」で「Snapshot Replication」を開きます。

フィルタ列の使用

スナップショット・ログを作成するときは、フィルタ列を使用するかどうかを指定できます。フィルタ列は、副問合せスナップショットを使用しているときは必須のコンポーネントです。フィルタ列は、WHERE 句内にあり、かつ等価結合列の一部ではない列を参照するスナップショットをサポートしているスナップショット・ログで定義する必要があります。

たとえば、次のような DDL を考えます。

```
1)  CREATE SNAPSHOT sales.orders AS
2)    SELECT * FROM sales.orders@dbs1.acme.com o
3)    WHERE EXISTS
4)      (SELECT c_id FROM sales.customer@dbs1.acme.com c
5)       WHERE o.c_id = c.c_id AND zip = 19555);
```

この DDL の 5 行目では、WHERE 句内で 3 つの列が参照されています。O.C_ID と C.C_ID の各列は等価結合句の一部として参照されていますが、ZIP 列は追加のフィルタ列です。したがって、SALES.CUSTOMER 表の ZIP 列のためのフィルタ列をスナップショット・ログに作成します。

作成するスナップショットの定義問合せを分析し、スナップショット・ログに作成する必要があるフィルタ列を識別することをお薦めします。スナップショット・ログにフィルタ列を作成せずに、フィルタ列を必要とするスナップショットを作成またはリフレッシュしようとした場合、スナップショットの作成またはリフレッシュは失敗します。

関連項目：

- 副問合せを使用するスナップショットの詳細は、3-8 ページの「スナップショットでのデータのサブセット化」を参照してください。
- スナップショット・ログの作成方法の詳細は、3-37 ページの「スナップショット・ログの作成」を参照してください。
- 副問合せを使用するスナップショットの詳細は、3-14 ページの「副問合せを使用するスナップショットに関する制限事項」を参照してください。

スナップショット環境の作成

スナップショット環境は、様々な場所から様々な方法で作成できます。一般的には、マスター・サイトの配置テンプレートを使用してスナップショット環境をローカルで事前作成し、それをターゲット・スナップショット・サイトに個別に配置します。

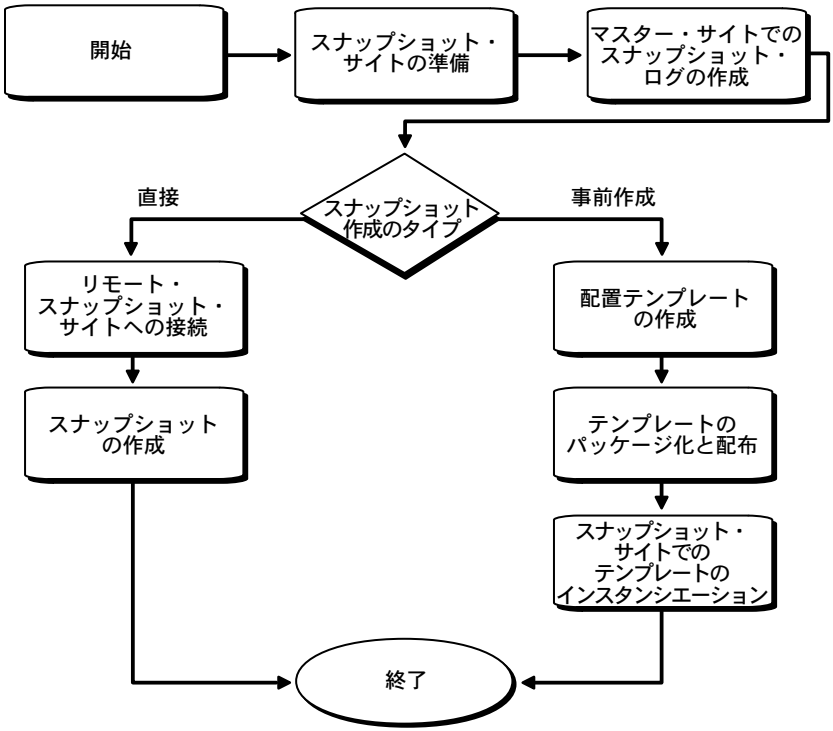
また、スナップショット・サイトへの接続を確立してスナップショット環境を直接作成する方法もあります。

Replication Manager

配置テンプレートを使用して Replication Manager でスナップショット環境をまとめて作成する方法の詳細は、Replication Manager のオンライン・ヘルプを参照してください。

また、リモート・スナップショット・サイトに直接接続し、Replication Manager を使用してスナップショット・サイトを個別に作成する方法の詳細は、Replication Manager のオンライン・ヘルプを参照してください。

図 3-13 スナップショット作成のフローチャート



レプリケーション・マネージメント API

配置テンプレートを使用してレプリケーション・マネージメント API でスナップショット環境をまとめて事前作成する方法の詳細は、『Oracle8*i* レプリケーション・マネージメント API リファレンス』の第 4 章「配置テンプレートの作成」を参照してください。

リモート・スナップショット・サイトに直接接続し、レプリケーション・マネージメント API を使用してスナップショット環境を個別に作成する方法の詳細は、『Oracle8*i* レプリケーション・マネージメント API リファレンス』の第 5 章「スナップショット・グループの作成」を参照してください。

4

配置テンプレートの概念およびアーキテクチャ

この章では、配置テンプレートの概要と、配置テンプレートを使用して簡単かつ効率的にスナップショット・サイトを配布する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。

- 大量配布の課題
- Oracle 配置テンプレートの概念
- 配置テンプレートのアーキテクチャ
- 配置テンプレートの設計
- スナップショット作成のローカル制御

注意： 配置テンプレートを作成する前に、第 3 章「スナップショットの概念およびアーキテクチャ」を読んでください。スナップショットについての理解は、配置テンプレートを作成する際に役に立ちます。

大量配布の課題

Oracle 配置テンプレートは、広範囲に分散したスナップショット環境を効率的に配置し、管理するためのツールです。配置テンプレートの概念、アーキテクチャおよび使用方法について説明する前に、大量配布環境の課題について検討します。

時と場所を問わず正確な情報を入手する必要性は、ますます強くなっています。また、情報は分散し、ユーザーもネットワークに接続されていない環境にいることが多く、実際に利用する場所に情報を配布する必要があります。

たとえば、モバイル端末を使用する営業担当員の場合を考えます。多くの営業担当員が、単純な操作でラップトップ上に正確な顧客情報を入手することを必要としています。これに応えるため、DBA は、データおよびデータベース・インフラストラクチャ（表、索引、制約、トリガーなど）を適切なタイミングで効率よくすべてのサイトに提供する必要があります。

これまで DBA は、独自の配置方法を開発する必要がありました。つまり、リモート・スナップショット・サイトにスナップショット環境を作成するための非常に複雑なスクリプトを作成する必要がありました。さらに、それに加えて、DBA はデータ・セットをスナップショット・サイトでカスタマイズすることが必要な場合もありました。スクリプトを作成した後も、スクリプトを配置するために手動でパッケージ化し、インプリメントする必要があり、多くの場合その両方に多大なトラブルシューティングが必要でした。

このような問題点をもとに、効率的な大量配布を行うための新しいテクノロジとリソースが生まれました。大量配布とは、データベース・インフラストラクチャ、データおよびフロントエンド・アプリケーションを多数のユーザーに配布するプロセスのことです。Oracle レプリケーションでは、大量配布という用語はデータおよびデータ・インフラストラクチャの配布のみを指します。

配置テンプレートおよび大量配布の目標

大量配布のツールおよびテクノロジは、データおよびデータベース・インフラストラクチャを配布する DBA を支援する必要があります。配置テンプレートの目標は、一度定義した環境をもとに必要な数の配置テンプレートのインスタンスを作成し、個々のサイトでカスタマイズを行えるようにすることです。

この目標を実現するために、Oracle の配置テンプレートでは次のことをサポートしています。

1 回のスナップショット環境定義
配置テンプレートを使用してスナップショット環境の構造を 1 回定義することにより、各ユーザー（サイト）がフロントエンド・アプリケーションをサポートするデータベース・インフラストラクチャを利用できるようにします。

スナップショット・サイトの個別のカスタマイズ
配置テンプレートのパラメータを使用して各スナップショット環境をカスタマイズし、各ユーザーが必要な特定のデータ・サブセットを受信できるようにします。

大量配布には、モバイル端末を使用する営業担当員やサービス技術員、小売店およびリモートの在庫保管サイトなどへの情報の配布といった様々な用途があります。このような環境では、配置テンプレートを使用してリモート・サイトにデータベース・インフラストラクチャを作成します。これは主に、配置テンプレートがデータのサブセット化、非接続環境でのレプリケーション、および少ないリソース要件をサポートしているからです（これはラップトップ・ユーザーに最適です）。

Oracle 配置テンプレートの概念

DBA は、Oracle に提供されている配置テンプレートを使用して、スナップショット環境をパッケージ化し、安全性の高いカスタム配置を簡単に実行できます。配置テンプレートには、固定データ・セットを使用する 1 つのスナップショットのみを含む単純なものも、1 つ以上の変数に基づく動的データ・セットを使用する数百のスナップショットを含む複雑なものもあります。

配置テンプレートには、次の機能があります。

- 集中制御
- 多数のスナップショット環境の配置
- リモート・サイトでのデータのサブセット化およびカスタマイズを可能にする、テンプレート・パラメータ
- テンプレートのインスタンシエーションおよびデータ・アクセスを制御する、認証ユーザー・リスト

スナップショット環境を配置する前に、マスター・サイトで配置テンプレートを作成します。このテンプレートには、リモート・サイトのオブジェクトやターゲット・リフレッ

シュ・グループを作成するデータ定義言語（DDL）など、スナップショット環境を配置するために必要なすべての情報を格納します。また、このテンプレートには、ユーザーのセキュリティ情報へのリンクや、カスタム・スナップショット作成のためのテンプレート・パラメータも格納します。

配置テンプレートの要素

各配置テンプレートには、スナップショット・サイトに必要なスナップショットおよびそれに関連するオブジェクトを作成するための設計図を含めます。具体的には、スナップショット環境を作成するために必要なスナップショット、トリガー、ビューなどをテンプレートに追加します。また、オプションでテンプレート・パラメータや認証ユーザーを定義して、インスタンシエーション・プロセス実行時の柔軟性や安全性を大幅に向上させることもできます。

配置テンプレートの要素は、次の 4 つのカテゴリに分類できます。

- テンプレート一般情報
- オブジェクト定義
- テンプレート・パラメータ
- ユーザー認証

テンプレート一般情報

テンプレート名、ターゲット・リフレッシュ・グループ、およびプライベートまたはパブリックの状態から構成されるテンプレート一般情報は、Oracle 配置テンプレートの基礎になる情報です。図 4-1 に示すように、REFRESH_TEMPLATE_NAME は、配置テンプレートのデータ・ディクショナリ・ビューのどの場面でも使用します。DBA は、スナップショット環境のオブジェクトをテンプレートに追加し、その後、指定されたテンプレートの識別に応じて配布するテンプレートをリリースします（図 4-2 を参照してください）。

配置テンプレートは、1 つのマスター・サイトで定義します。マスター・サイトに同じ名前の配置テンプレートを 2 つ定義することはできませんが、配置テンプレートを同じ名前で他のサイトにコピーすることはできます。

図 4-1 リフレッシュ・グループのテンプレートのビューの関係

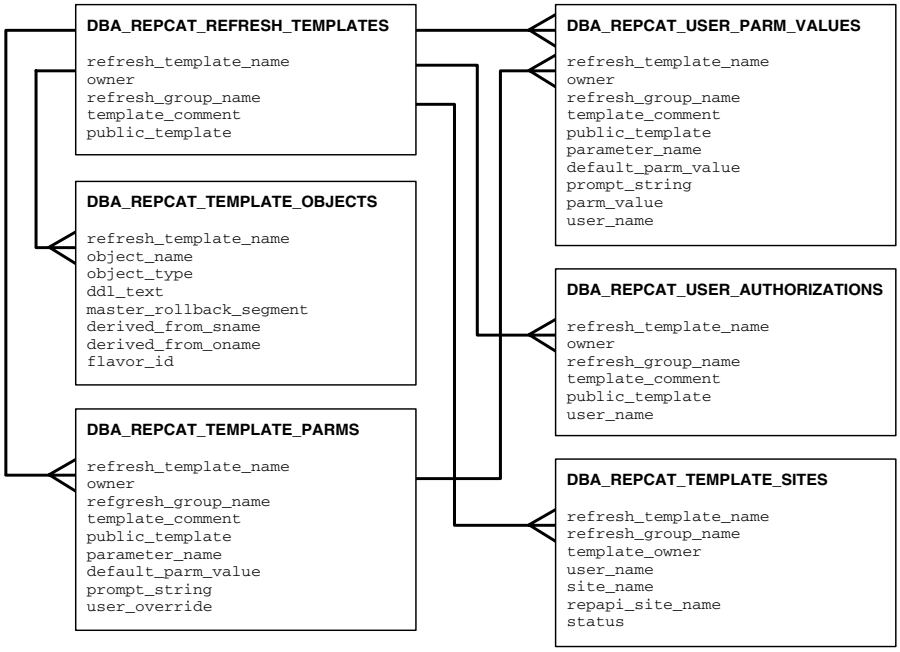
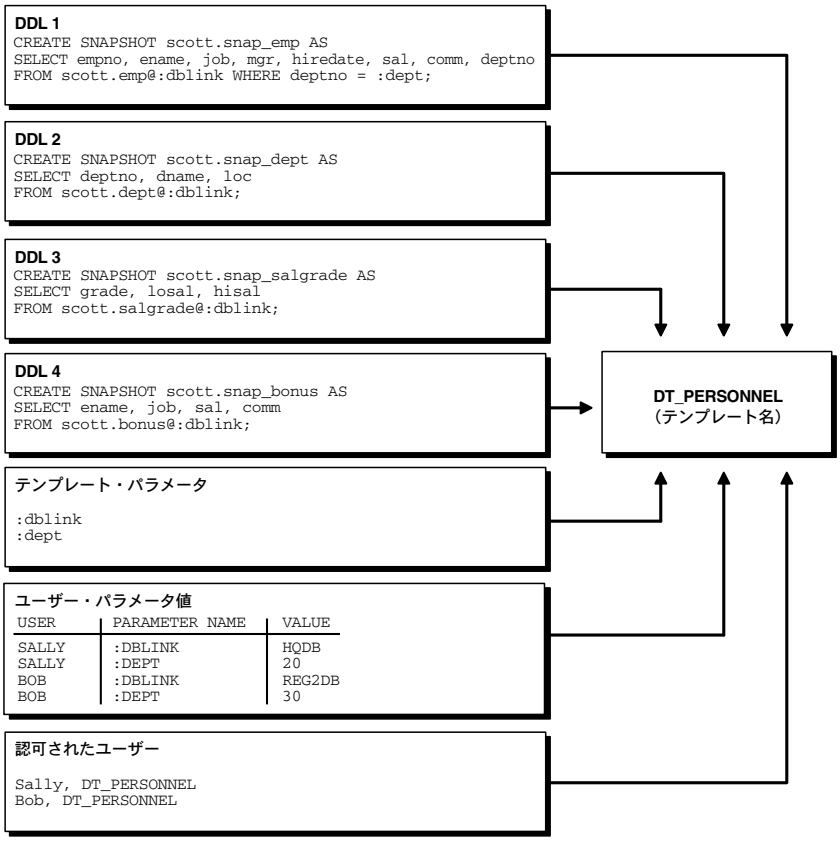


図 4-2 テンプレートに追加する配置テンプレートの要素



オブジェクト定義

配置テンプレートを定義した後で、テンプレートにオブジェクトを追加します。テンプレートがスナップショット・サイトでインスタンシエートされると、オブジェクト DDL（CREATE SNAPSHOT、CREATE TABLE など）が実行され、スナップショット・サイトに適切なオブジェクトが作成されます。

配置テンプレートに追加するオブジェクトは既存のマスター・オブジェクトをベースに作成できますが、必要な場合は、DDL を定義して新しいテンプレート・オブジェクトを作成することもできます。新しいオブジェクト DDL を作成した場合は、実行エラーを防ぐために DDL が字句規則に適合しているかどうかのチェックが行われます。配置テンプレートに追加する更新可能スナップショットはマスター・グループ内の表をベースとする必要があります

が、読取り専用スナップショットなどのその他のオブジェクトは、マスター・グループに属していないオブジェクトをベースとしても構いません。

通常、テンプレートにはスナップショットを追加しますが、必要な場合はその他のオブジェクトも追加できます。たとえば、スナップショット・サイトでのデータの整合性を保証するための制約を追加したり、データ表示のためのビューを追加したり、またはデータをローカルに格納するための表を追加したりできます。場合によっては、配置テンプレートにアプリケーションのすべてのオブジェクトを含めることもできます。配置テンプレートを使用して作成したスナップショットは、テンプレートに定義されたリフレッシュ・グループに自動的に追加されます。

配置テンプレートを Oracle8*i* Lite のスナップショット・サイトでインスタンシエートすると、スナップショットのみが作成されます。スナップショットとその他の種類のオブジェクトを含む配置テンプレートを Oracle8*i* Lite サイトでインスタンシエートすると、スナップショットは作成されますが、その他のオブジェクトは作成されません。

> **関連項目：** リフレッシュ・グループの詳細は、4-4 ページの「テンプレート一般情報」を参照してください。

テンプレート・パラメータ

各ターゲット・スナップショット・サイトで一意のデータ・セットが必要な場合は、オブジェクト DDL で変数を定義できます。変数を定義してパラメータ付きテンプレートを作成すると、テンプレートをインスタンシエートするときにカスタム・データ・セットを使用できるため、スナップショット・サイトごとに異なるデータ・セットを使用できます。これらのパラメータは、オブジェクト DDL に組み込まれます。テンプレートのインスタンシエーション時に、個別のユーザー値がパラメータに代入されます。

Oracle では、テンプレートにデフォルト値およびユーザー固有のパラメータ値を指定できます。パラメータ値は、配置テンプレートの作成時または作成後に指定できますが、テンプレートをインスタンシエートする前に指定する必要があります。ユーザーは、テンプレートのインスタンシエーション時にパラメータに値を指定できません。

ユーザー固有のパラメータ値を使用した場合は、指定されたユーザーがテンプレートをインスタンシエートするときにそのパラメータ値が自動的に使用されます。たとえば、変数 REGION について考えます。テンプレート SALES_TEMP に対して次のようなユーザー固有のパラメータ値を指定したとします。

USER	REGION
SCOTT	EAST
LARRY	WEST

この場合、スナップショットの定義 SELECT 文は次のようになります。

```
SELECT cust_id, sales_to_date, status FROM table_x WHERE region_id=:region;
```

ユーザー SCOTT と LARRY がテンプレート SALES_TEMP をインスタンシエートすると、スナップショットのデータ・セットはそれぞれ次のようになります。

ユーザー SCOTT	
CUST ID	**REGION**
A123	EAST
A234	EAST
A345	EAST
A456	EAST

ユーザー LARRY	
CUST ID	**REGION**
B123	WEST
B234	WEST
B345	WEST
B456	WEST

WHERE 句のテンプレート・パラメータとセキュリティ

カスタマイズしたデータ・サブセットを作成する他に、CREATE SNAPSHOT 文の WHERE 句でテンプレート・パラメータを指定して、WHERE 句の条件を満たすデータのみをスナップショット・サイトで表示および変更するようにできます。たとえば、ユーザー固有のパラメータ・リストの REGION パラメータに次のように指定したとします。

USER	REGION
SCOTT	EAST
LARRY	WEST

ユーザー SCOTT がインスタンシエートしたスナップショットにアクセスするユーザーは、地域 EAST のデータのみを参照でき、この WHERE 句に適合するデータのみを表示、更新または削除できます。つまり、このスナップショットには地域 EAST のデータしか含まれていないため、このスナップショットのユーザーは地域 WEST のデータを表示、更新または削除できません。

ユーザー認証

配置テンプレートは、パブリックまたはプライベートのいずれかとして定義できます。これは、テンプレートの作成時に設定します。パブリック・テンプレートは、マスター・サイトへのアクセス権を持つすべてのユーザーがインスタンシエートできます。

テンプレートがプライベート用として作成されている場合は、認証ユーザーのみがターゲット・テンプレートをインスタンシエートできます。プライベート用として使用する場合は、マスター・サイトに認証されたユーザーのリストを作成します。認証されていないユーザー

がターゲット・テンプレートをインスタンシエートしようとすると、そのプロセスは失敗します。

配置サイト

集中制御の目的で、リモート・スナップショット・サイトでインスタンシエートされた環境の特性の一部を監視および管理できます。具体的には、配置テンプレートをインスタンシエートしたサイト（配置テンプレート名、認証ユーザーおよびインスタンシエートされた環境の状態を含みます）を監視できます。

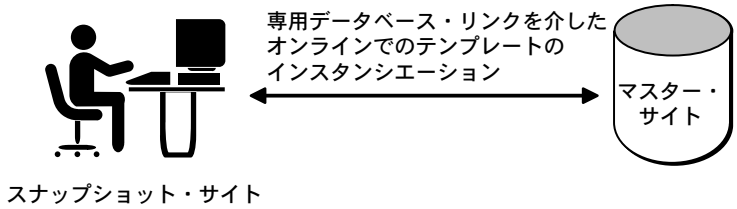
配置テンプレートのパッケージ化およびインスタンシエーション

配置テンプレートの定義が完了した後で、テンプレートをパッケージ化して、リモート・スナップショット・サイトでインスタンシエートできます。パッケージ化した配置テンプレートがスナップショット・サイトでインスタンシエートされると、スナップショット・サイト・オブジェクトが作成され、スナップショットにデータが移入されます。リモート・スナップショット・サイトは、オンライン・インスタンシエーションまたはオフライン・インスタンシエーションのいずれかを使用して作成できます。

オンライン・インスタンシエーション

オンライン・インスタンシエーションでは、スナップショット・サイトはターゲット・マスター・サイトに接続した状態で配置テンプレートをインスタンシエートできます。オンラインのインスタンシエーション・プロセスでは、スナップショット・サイトの構造が作成され、指定したデータ・サブセットがマスター・サイトから取り出されて適切なスナップショットに格納されます。

図 4-3 オンライン・インスタンシエーション



Oracle8*i* Enterprise Edition、Oracle8*i* Standard Edition または Oracle8*i* Personal Edition を使用するスナップショット・クライアントの場合、オンライン・インスタンシエーション用の配置テンプレートをパッケージ化することは、スクリプト・ファイルを生成することを意味します。このスクリプト・ファイルがスナップショット・サイトで実行されると、スナップショット・オブジェクトが作成された後に、マスター・サイトに接続されて、スナップショットにデータが移入されます。マスター・サイトからスナップショットへのネットワークを介したデータの移入には、CREATE SNAPSHOT ... AS SELECT などの SQL 文が使用されます。

Oracle8*i* Lite スナップショット・クライアントのオンライン・インスタンシエーション用のテンプレートをパッケージ化するときは、インスタンシエーション用のスクリプトまたはバイナリ・ファイルは必要ありません。ユーザーは、1 回の操作で配置テンプレートのパッケージ化とインスタンシエーションの両方を行えるように、Oracle Client Replication Tool などのクライアント・アプリケーションを使用してマスター・サイトに接続します。

オンライン・インスタンシエーションのメリットの 1 つは、データ・サブセットがインスタンシエーション・プロセス実行時点の最新の状態である点です。ただし、これにはデメリットもあります。オンライン・インスタンシエーションの実行中はスナップショット・サイトとマスター・サイトが接続されている必要があるため、作成するスナップショット環境のサイズによっては、ネットワーク・トラフィックが増大する可能性があります。

また、モデム接続を使用しているラップトップ・ユーザーは、接続時間が長くなることがあります。接続時間は、作成されるオブジェクトの数、スナップショットの問合せの複雑さ、送信されるデータの量によって変わります。低帯域幅のモデムで接続している場合、これらの要素による影響は特に大きくなります。

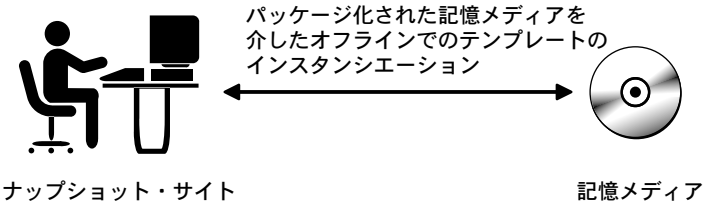
オフライン・インスタンシエーション

テンプレートをオフラインでインスタンシエートすれば、ピーク使用期間中のサーバー負荷を小さくし、リモートの接続時間を短縮できます。オフライン・インスタンシエーション用のテンプレートをパッケージ化することは、DDL およびデータ操作言語（DML）を含むスクリプトまたはバイナリ・ファイルを生成することを意味します。このスクリプトまたはバイナリ・ファイルによって、配置テンプレートに定義したスナップショット環境が作成され、データが移入されます。DBA は、スクリプトまたはバイナリ・ファイルをパッケージ化して記憶メディア（テープ、CD-ROM など）に保存し、スナップショット・サイトに転送する方法を提供します。各スナップショット・サイトで、別々のオフライン・インスタンシエーション・スクリプトが必要です。

インスタンシエーション用のテンプレートをパッケージ化すると、スナップショットのベースとなっているテンプレートの各マスター表のスナップショット・ログに、変更のロギングが開始されます。特定のマスター表のスナップショット・ログに記録された変更内容は、マスター表をベースとするすべてのスナップショットがインスタンシエーション後にリフレッシュされるまで消去されません。したがって、テンプレートをパッケージ化した後で、テンプレートのインスタンシエーションとスナップショットのリフレッシュをできるだけ早く実行して、スナップショット・ログが大きくなりすぎないようにする必要があります。

インスタンシエーション時は、テンプレートとデータがマスター・サイトではなく記憶メディアから取り出されます。これには、ネットワーク・トラフィックを軽減でき、ネットワークに常時接続しなくてよいというメリットがあります。ただし、インスタンシエーション後は、スナップショット・サイトにはパッケージ化した時点のマスター・サイトのデータが反映されているので、リフレッシュによって更新する必要があります。

図 4-4 オフライン・インスタンシエーション



オフライン・インスタンシエーションは、多数のラップトップやその他の非接続環境のコンピュータでターゲット・テンプレートをインスタンシエートする大量配布に最適なソリューションです。

配置テンプレートのインスタンシエーションのシナリオ

ターゲット・インスタンシエーション・サイトで、次のいずれかの Oracle データベース・クライアントが稼動している必要があります。

- Oracle8*i* Enterprise Edition
- Oracle8*i* Standard Edition
- Oracle8*i* Personal Edition
- Oracle8*i* Lite

最も一般的な大量配布のシナリオでは、マスター・サイトで Oracle8*i* Enterprise Edition を使用し、リモート・サイト（通常、ラップトップ）で Oracle8*i* Lite を使用します。

表 4-1 に、配置テンプレートのインスタンシエーションのシナリオの概要を示します。

表 4-1 配置テンプレートのインスタンシエーションのシナリオ

インスタンシエーションのタイプ	スナップショット・クライアントのタイプ	説明
オフライン	Oracle8*i* Enterprise Edition Oracle8*i* Standard Edition Oracle8*i* Personal Edition	ユーザーは、SQL*Plus を使用してオフライン・インスタンシエーション・スクリプトを実行します。オフライン・インスタンシエーション・スクリプトには、スナップショット・サイトのオブジェクトを作成する CREATE 文と、スナップショットにデータを移入する INSERT 文が含まれます。
オフライン	Oracle8*i* Lite （Java RepAPI）	ユーザーは、Oracle Client Replication Tool などのクライアント・アプリケーション・ツールを使用して、オフライン・インスタンシエーション用のバイナリ・ファイルを開きます。次に、開いたバイナリ・ファイルによってスナップショット・サイト・オブジェクトが作成され、スナップショットにデータが移入されます。
オンライン	Oracle8*i* Enterprise Edition Oracle8*i* Standard Edition Oracle8*i* Personal Edition	ユーザーは、SQL*Plus を使用してオンライン・インスタンシエーション・スクリプトを実行します。オンライン・インスタンシエーション・スクリプトには、スナップショット・サイトのオブジェクトを作成する CREATE 文が含まれます。スナップショット・オブジェクトの作成時に、オンライン・インスタンシエーション・スクリプトはマスター・サイトに接続します。次に、CREATE SNAPSHOT ...AS SELECT 文を使用してスナップショットを作成し、データを移入します。
オンライン	Oracle8*i* Lite （Java RepAPI）	ユーザーは、Oracle Client Replication Tool などのクライアント・アプリケーション・ツールを使用してマスター・サイトに接続します。このツールを使用すると、配置テンプレートのパッケージ化とインスタンシエーションを 1 回の操作で実行できます。クライアント側では、ファイルは必要ありません。

DBA またはターゲット・ユーザーのいずれかが、配置テンプレートをパッケージ化できます。配置テンプレートをパッケージ化するには、Replication Manager の「Offline Instantiation Wizard」（オフライン・インスタンシエーションの場合）またはレプリケーション・マネージメント API（オフラインおよびオンライン・インスタンシエーションの場合）を使用します。パッケージ化するには、エンドユーザーはパブリック API を使用し、DBA は一般的にプライベート API を使用します。通常、配置テンプレートをオフラインでインスタンシエートする場合は DBA がパッケージ化を実行しますが、配置テンプレートをオンラインでインスタンシエートする場合はユーザーがパッケージ化を実行する可能性があ

ります。ただし、ユーザーまたは DBA にとっては使用する API コールが異なること以外に制約はなく、どちらもオンラインまたはオフラインによるパッケージ化を実行できます。

配置テンプレートをパッケージ化するには、次のレプリケーション・マネージメント API のファンクションを使用できます。

プライベート・ファンクション（DBA のみ使用）:

- DBMS_REPCAT_RGT.INSTANTIATE_OFFLINE
- DBMS_REPCAT_RGT.INSTANTIATE_OFFLINE_REPAPI
- DBMS_REPCAT_RGT.INSTANTIATE_ONLINE

パブリック・ファンクション :

- DBMS_REPCAT_INSTANTIATE.INSTANTIATE_OFFLINE
- DBMS_REPCAT_INSTANTIATE.INSTANTIATE_OFFLINE_REPAPI
- DBMS_REPCAT_INSTANTIATE.INSTANTIATE_ONLINE

関連項目： 詳細は、7-13 ページの「配置テンプレートをインスタンシエートするスナップショット・サイトの準備」および『Oracle8*i* レプリケーション・マネージメント API リファレンス』を参照してください。

注意： オフライン・インスタンシエーション用の配置テンプレートをパッケージ化すると、関連するスナップショット・ログに、テンプレートにパッケージ化したスナップショットのロギングが開始されます。この即時ロギングによって、スナップショット・サイトでオフライン・インスタンシエーション完了後に高速リフレッシュを実行できます。オフライン・インスタンシエーションの実行後、スナップショット・ログを監視して、リモート・スナップショット・サイトのリフレッシュが適切なときに実行されていることを確認してください。テンプレートをパッケージ化するときにロギングが開始されるため、リモート・スナップショット・サイトがリフレッシュされないとスナップショット・ログが大きくなりすぎることがあります。

配置テンプレートのアーキテクチャ

Oracle では、スナップショット環境を迅速かつ効率的に配布するために、配置テンプレートに標準スナップショット・アーキテクチャを使用しています。配置テンプレートでは、スナップショット定義の作成、リフレッシュ特性、競合解消およびグループ化に、スナップショット環境を手動で作成する場合と同じ方法を使用します。ただし、配置テンプレートでは DDL を実行してオブジェクトを即時に作成するのではなく、テンプレートをインスタンシエートするときに、配置テンプレートに含まれるオブジェクト DDL が実行される点が異なります。

システム表に格納されたテンプレート定義

配置テンプレートでは、スナップショット・サイトで DDL を実行してスナップショット環境を即時に作成するのではなく、スナップショットおよび関連するオブジェクト定義が配置テンプレートに格納されます。すべてのオブジェクト定義を配置テンプレートに追加した後、テンプレートをインスタンシエートすると、格納されているすべての DDL がリモート・スナップショット・サイトで実行され、必要なスナップショット環境が作成されます。

これらのオブジェクト定義はすべて、配置テンプレートの定義サイトにあるシステム表に、配置テンプレート名をキーにして格納されます。配置テンプレートをパッケージ化すると、格納されているオブジェクト DDL がシステム表から取り出され、バイナリ・ファイルのインスタンシエーション・スクリプトが作成されます。唯一の例外は、Oracle8*i* Lite スナップショット・サイトのオンライン・インスタンシエーションでは、ローカル・ファイルを必要としない点です。これは、Oracle Client Replication Tool などのクライアント・アプリケーション・ツールによってマスター・サイトへの接続時にパッケージ化が自動的に実行されるためです。

標準 DDL の使用

テンプレートのオブジェクト定義は、スナップショット・サイトでローカルにオブジェクトを作成する場合と同じ DDL を使用して作成されます。たとえば、スナップショットをローカルで作成するときは、次の DDL を実行します。

```
CREATE SNAPSHOT foo_snap AS SELECT empid, region, dept, salary
   FROM scott.foo@hq.com;
```

これと同じスナップショットを配置テンプレートに追加するには、Replication Manager の「Deployment Template Wizard」を使用するか、または次の例に示すように CREATE_TEMPLATE_OBJECT ファンクションを使用します。

```
DECLARE
a NUMBER;
BEGIN
   a := DBMS_REPCAT_RGT.CREATE_TEMPLATE_OBJECT(
      refresh_template_name => 'dt_snapenv',
      object_name => 'foo_snap',
      object_type => 'snapshot',
      ddl_text => 'CREATE SNAPSHOT foo_snap
                     AS SELECT empid, region, dept, salary
                     FROM scott.foo@hq.com');
END;
/
```

注意： DDL_TEXT パラメータの一重引用符で囲まれた DDL 文の末尾には、セミコロンを付けないでください。

このファンクションを実行すると、DT_SNAPENV という名前の配置テンプレートにスナップショット定義が追加されます。このスナップショットをインスタンシエートすると、スナップショット FOO_SNAP が作成されます。スナップショットを作成するのみでなく、表、トリガー、プロシージャ、索引およびその他のオブジェクト定義を配置テンプレートに追加できます。

スナップショットを作成するときは、スナップショットの問合せで表所有者のスキーマ名を必ず指定してください。前述の例では、FOO 表の所有者に SCOTT が指定されています。

関連項目： CREATE_TEMPLATE_OBJECT ファンクションの使用方法の詳細は、『Oracle8*i* レプリケーション・マネージメント API リファレンス』の「DBMS_REPCAT_RGT パッケージ」の「CREATE_TEMPLATE_OBJECT ファンクション」を参照してください。

パッケージ化およびインスタンシエーションのプロセス

リモート・スナップショット・サイトでのインスタンシエーション用に配置テンプレートをパッケージ化するときは、オンライン・インスタンシエーション用とオフライン・インスタンシエーション用のテンプレートを用意します。インスタンシエーション・プロシージャによって、リモート・スナップショット環境が作成され、その環境にデータが移入されます。

オンライン・インスタンシエーション用の配置テンプレートのパッケージ化

オンライン・インスタンシエーション用に配置テンプレートをパッケージ化すると、リモート・スナップショット環境の作成に必要な DDL が生成され、すべてのテンプレート・パラメータの代入処理が実行されます。生成された DDL が格納される場所は、スナップショット・クライアントのタイプによって異なります。

- スナップショット・クライアントが Oracle8*i* Enterprise Edition、Oracle8*i* Standard Edition または Oracle8*i* Personal Edition の場合、オンライン・インスタンシエーション用のスクリプトは、レプリケーション・マネージメント API を実行してテンプレートをパッケージ化したコンピュータのハード・ドライブにローカルに格納されます。このコンピュータがスナップショット・サイトのコンピュータでない場合は、オンライン・インスタンシエーション・ファイルをスナップショット・サイトに転送する必要があります。
- スナップショット・クライアントが Oracle8*i* Lite の場合は、ファイルは必要ありません。ユーザーは、オンライン・インスタンシエーション時に、Oracle Client Replication Tool などのクライアント・アプリケーション・ツールを使用して、スナップショット・サイトの構造やデータをマスター・サイトからリモート・スナップショット・サイトへ直接取り出します。したがって、クライアント・アプリケーション・ツールによってテンプレートのパッケージ化が自動的に実行されます。

オフライン・インスタンシエーション用の配置テンプレートのパッケージ化

オフライン・インスタンシエーション用に配置テンプレートをパッケージ化するときに、リモート・スナップショット環境の作成に必要な DDL と、スナップショット環境へのデータの移入に必要な DML を、生成したファイルに格納します。また、パッケージ化の最中に、すべてのテンプレート・パラメータの代入が実行されます。

テンプレートをパッケージ化するときに、オフライン・インスタンシエーション用のスクリプトまたはバイナリ・ファイルを作成し、ハード・ディスク、CD-ROM、テープなどの記憶デバイスに保存します。オフライン・インスタンシエーション用の配置テンプレートをパッケージ化するには、Replication Manager の「Offline Instantiation Wizard」またはレプリケーション・マネージメント API を使用できます。

- スナップショット・クライアントが Oracle8*i* Enterprise Edition、Oracle8*i* Standard Edition または Oracle8*i* Personal Edition の場合は、オフライン・インスタンシエーション・スクリプトは、要求を発行してテンプレートをパッケージ化したコンピュータのハード・ドライブにローカルに格納されます。このコンピュータがスナップショット・サイトのコンピュータでない場合は、オフライン・インスタンシエーション・ファイルをスナップショット・サイトに転送する必要があります。
- スナップショット・クライアントが Oracle8*i* Lite の場合は、オフライン・インスタンシエーション用のバイナリ・ファイルはマスター・サイトに格納されるので、このファイルをスナップショット・サイトに転送する必要があります。

リモート・スナップショット・サイトでテンプレートをインスタンシエートするときは、記憶メディアまたはローカルのハード・ドライブからスクリプトまたはバイナリ・ファイルが実行されます。これにより、スナップショット環境が作成され、パッケージ化プロセスで定義したデータ・セットが環境に移入されます。前述のように、個々のサイト用のデータ・セットを定義するテンプレート・パラメータはすべて、パッケージ作成プロセスで定義します。

オンライン・インスタンシエーション

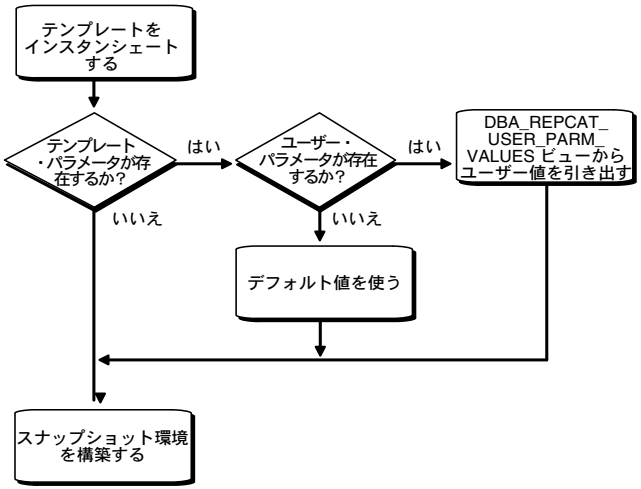
オンライン・インスタンシエーション・プロセスでは、スナップショット・サイトの構造が作成され、指定したデータ・サブセットがマスター・サイトから取り出されて適切なスナップショットに格納されます。リモート・スナップショット・サイトでオンライン・インスタンシエーション・プロセスを開始すると、配置テンプレート用に定義したパラメータが評価されます。パラメータに定義した値は、テンプレートのオブジェクト DDL を実行するときに、リモート・スナップショット・サイトにカスタム・データ・セットをインストールするために使用されます。同時に、スナップショットがマスター・サイトに登録され、スナップショット・ログにマスター表への変更のロギングが開始されます。

テンプレート・パラメータ値は、デフォルト・パラメータ値またはユーザー・パラメータ値を使用して定義できます。Oracle はこれらのパラメータ値が存在するかどうかをチェックし、次の順に使用します。

1. ユーザー・パラメータ値
2. デフォルト・パラメータ値

ユーザー・パラメータ値が定義してあり、リストに含まれているユーザーがテンプレートをインスタンシエートする場合、テンプレートをインスタンシエートするときにユーザー・パラメータ値が使用されます。ユーザー・パラメータ値を定義しないと、デフォルト・パラメータ値が使用されます。図 4-5 は、パラメータのチェック・プロセスを示します。

図 4-5 オンライン・インスタンシエーション時のパラメータのチェック



パラメータのチェックが終わると、テンプレートによって作成されたオブジェクトが、テンプレートの作成時に指定したリフレッシュ・グループに追加されます。

オフライン・インスタンシエーション

大量配布環境では、ほとんどのスナップショット環境でオフライン・インスタンシエーションを使用して、必要なスナップショット環境を作成します。配置テンプレートをパッケージ化すると、スクリプトまたはバイナリ・ファイルが作成され、スナップショット環境の作成に必要な DDL、インスタンシエーション・プロセスで使用されるパラメータ値およびスナップショット環境へのデータの移入に必要な DML が格納されます。

スクリプトまたはバイナリ・ファイルは、CD-ROM やフロッピィ・ディスクなどの記憶メディアにコピーしたり、Web サイトや FTP サイトに保存してリモート・スナップショット・サイトにダウンロードしたりできます。このような柔軟な配布メカニズムによって、管理者とユーザーは最も効率的な方法で配置テンプレートをインスタンシエートできます。

パッケージ化およびインスタンシエーションのオプション

配置テンプレートのパッケージ化およびインスタンシエーションには、様々な形態があります。表 4-2 は、パッケージ化およびインスタンシエーションの形態とメカニズム、および操作の実行時に使用するドキュメントのリストを示します。

表 4-2 パッケージ化およびインスタンシエーションのオプション （1 / 2 ページ）

インスタンシエーションのタイプ	インスタンシエーションを実行するクライアントのタイプ	使用するテンプレートのパッケージ化	パッケージ化ドキュメント	使用するテンプレートのインスタンシエーション	インスタンシエーションのドキュメント
オフライン	Oracle8*i* Enterprise Edition Oracle8*i* Standard Edition Oracle8*i* Personal Edition	Replication Manager の「Offline Instantiation Wizard」	Replication Manager のオンライン・ヘルプの「Package for Offline Instantiation: Overview」のトピックを参照してください（「Help Contents」で「Deployment Templates」>「Packaging and Instantiation」の順に選択します）	オフライン・インスタンシエーション・スクリプトおよび SQL*Plus	Replication Manager のオンライン・ヘルプの「Instantiate at Remote Snapshot Site」のトピックを参照してください（「Help Contents」で「Deployment Templates」>「Packaging and Instantiation」の順に選択します）
オフライン	Oracle8*i* Enterprise Edition Oracle8*i* Standard Edition Oracle8*i* Personal Edition	レプリケーション・マネージメント API（PL/SQL パッケージおよび SQL*Plus）	『Oracle8*i* レプリケーション・マネージメント API リファレンス』の第 4 章の「インスタンシエーションのためのパッケージ化」の項を参照してください	オフライン・インスタンシエーション・スクリプトおよび SQL*Plus	『Oracle8*i* レプリケーション・マネージメント API リファレンス』の第 4 章の「配置テンプレートのインスタンシエーション」の項を参照してください
オンライン	Oracle8*i* Enterprise Edition Oracle8*i* Standard Edition Oracle8*i* Personal Edition	レプリケーション・マネージメント API（PL/SQL パッケージおよび SQL*Plus）	『Oracle8*i* レプリケーション・マネージメント API リファレンス』の第 4 章の「インスタンシエーションのためのパッケージ化」の項を参照してください	オンライン・インスタンシエーション・スクリプトおよび SQL*Plus	『Oracle8*i* レプリケーション・マネージメント API リファレンス』の第 4 章の「配置テンプレートのインスタンシエーション」の項を参照してください

表 4-2 パッケージ化およびインスタンシエーションのオプション （2 / 2 ページ）

インスタンシエーションのタイプ	インスタンシエーションを実行するクライアントのタイプ	使用するテンプレートのパッケージ化	パッケージ化ドキュメント	使用するテンプレートのインスタンシエーション	インスタンシエーションのドキュメント
オフライン	Oracle8*i* Lite（Java RepAPI）	Replication Manager の「Offline Instantiation Wizard」	Replication Manager のオンライン・ヘルプの「Package for Offline Instantiation: Overview」のトピックを参照してください（「Help Contents」で「Deployment Templates」>「Packaging and Instantiation」の順に選択します）	オフライン・インスタンシエーション用のバイナリ・ファイル、および Oracle Client Replication Tool などのクライアント・アプリケーション・ツール	テンプレートのインスタンシエーションの詳細は、Oracle8*i* Lite のドキュメントを参照してください
オフライン	Oracle8*i* Lite（Java RepAPI）	レプリケーション・マネージメント API（PL/SQL パッケージおよび SQL*Plus）	『Oracle8*i* レプリケーション・マネージメント API リファレンス』の第 4 章の「インスタンシエーションのためのパッケージ化」の項を参照してください	オフライン・インスタンシエーション用のバイナリ・ファイル、および Oracle Client Replication Tool などのクライアント・アプリケーション・ツール	テンプレートのインスタンシエーションの詳細は、Oracle8*i* Lite のドキュメントを参照してください
オンライン	Oracle8*i* Lite（Java RepAPI）	Oracle Client Replication Tool などのクライアント・アプリケーション・ツール	テンプレートのパッケージ化の詳細は、Oracle8*i* Lite のドキュメントを参照してください	Oracle Client Replication Tool などのクライアント・アプリケーション・ツール	テンプレートのインスタンシエーションの詳細は、Oracle8*i* Lite のドキュメントを参照してください

インスタンシエーション後の作業

リモート・スナップショット・サイトで配置テンプレートのインスタンシエーションを実行すると、スナップショット・サイトでローカルにスナップショット環境を作成した場合とまったく同じ構造が作成されます。具体的には、指定した名前でスナップショットが作成され、制約整合性を保つために主キーに基づいて索引が作成されます。また、テンプレートのその他のオブジェクトも、スナップショット・サイトで手動で作成した場合と同じように作成されます。

オフライン・インスタンシエーションの場合は、サーバーでパッケージ化してからリモート・スナップショット・サイトでインスタンシエートするまでの時間が長くなれば、インスタンシエーション後の最初のリフレッシュに要する時間も長くなります。スナップショット・サイトでは、マスター・サイトのスナップショット・ログを使用して、テンプレートがパッケージ化された時点以降の高速リフレッシュを実行します。前述のように、マスター表

に対して行った変更は、配置テンプレートをパッケージ化するとただちにスナップショット・ログに記録されます。

> **関連項目：** 詳細は、3-17 ページの「スナップショットのアーキテクチャ」を参照してください。

スナップショット・グループ

インスタンシエートした配置テンプレートによって作成されたオブジェクトは、オブジェクトのマスター・グループ名から付けられた名前を持つスナップショット・グループに自動的に追加されます。たとえば、マスター・グループ PERSONNEL および TECHNICAL のオブジェクトを含む配置テンプレート DT_SNAPENV をインスタンシエートした場合、テンプレート・オブジェクトはスナップショット・グループ PERSONNEL01 および TECHNICAL01 にそれぞれ追加されます（同じマスター・グループのオブジェクトが複数のテンプレートに含まれる場合は、それぞれ異なる番号付き接尾辞が付けられます）。前述のように、スナップショット・グループはターゲット・マスター・グループと編成上の整合性を保つのみでなく、更新可能なスナップショットを使用するために必要です。

> **関連項目：** 詳細は、3-23 ページの「スナップショット・グループ」を参照してください。

リフレッシュ・グループ

最初に配置テンプレートを作成するときは、テンプレートのスナップショット・オブジェクトを追加するリフレッシュ・グループの名前を定義する必要があります。インスタンシエーション・プロセスが完了した後で、マスター・サイトに常時ネットワーク接続している場合は、リフレッシュ・グループ内のスナップショットを設定した間隔で自動的にリフレッシュするように指定できます。

Oracle8*i* Enterprise Edition、Oracle8*i* Standard Edition および Oracle8*i* Personal Edition の場合は、Replication Manager または DBMS_REFRESH.CHANGE プロシージャを使用して、リフレッシュ・グループのリフレッシュ間隔と次回リフレッシュするデータを変更できます。Replication Manager でこれらの設定を変更するには、リフレッシュ・グループを選択して、「Next Date」フィールドと「Interval」フィールドを編集します。DBMS_REFRESH.CHANGE プロシージャを使用してこれらの設定を変更する場合は、interval パラメータと next_date パラメータを適切に設定します。スナップショット・サイトからマスター・サイトに常時ネットワーク接続していない場合は、必要に応じてリフレッシュ・グループをリフレッシュできます。

注意： Oracle8*i* Lite サイトのリフレッシュ・グループのリフレッシュ間隔をあらかじめ設定しておくことはできません。必要に応じてリフレッシュしてください。また、Oracle8*i* Lite サイトでインスタンシエートする配置テンプレートを作成する場合は、リフレッシュ・グループの名前を配置テンプレートの名前と同じにする必要があります。

配置テンプレートの設計

配置テンプレートを作成する前に、テンプレートの設計を行う必要があります。配置テンプレートのパラメータと副問合せのサブセット化を組み合せることで、DBA は、割当て表と垂直パーティション化されたデータを使用して広範囲に分散したデータベース環境を管理できます。垂直パーティション化の要件およびレプリケート環境に必要なデータ・セットについては、さらにいくつかの設計上の考慮事項があります。

> **関連項目：** データのサブセット化の詳細は、3-8 ページの「スナップショットでのデータのサブセット化」を参照してください。垂直パーティション化の詳細は、4-22 ページの「垂直パーティション化」を参照してください。

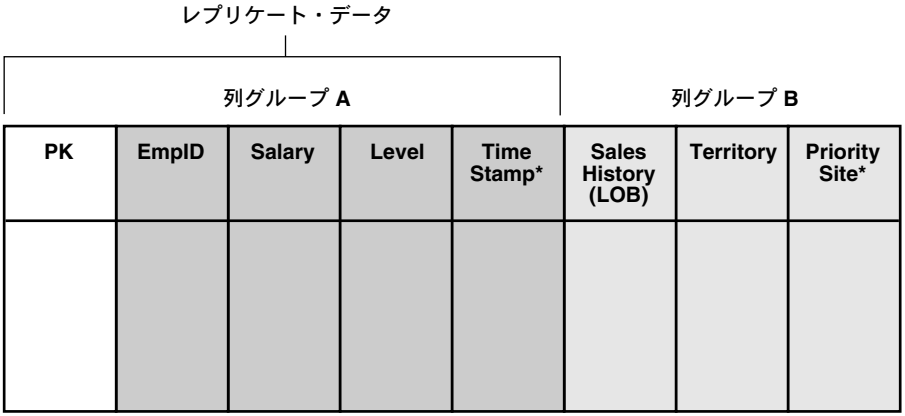
垂直パーティション化

配置テンプレートを使用すると、垂直パーティション化された更新可能スナップショットを作成できる利点もあります。垂直パーティション化は、列のサブセット化とも呼ばれます。たとえば、多くの軽量なクライアントを含む大量配布環境では、LOB データ自体を実際にレプリケートするのではなく、LOB データを含む表をレプリケートすることが必要な場合があります。これを行うには、垂直パーティションを定義するときに、レプリケート対象として選択した列から LOB 列を除外します。

配置テンプレートに追加する垂直パーティション化されたスナップショットは、次の要素を含んでいる必要があります。

- 主キー
- レプリケート列の競合解消に使用するすべての列（図 4-6 を参照してください）

図 4-6 垂直パーティション化されたデータのレプリケーション



*列グループの競合解消列を示す

列 PK、EmpID、Salary および Level（図 4-6 を参照してください）をレプリケートするスナップショット・オブジェクトを追加する場合は、列グループ A に含まれる各列の競合解消に使用するため、Time Stamp 列も含める必要があります。

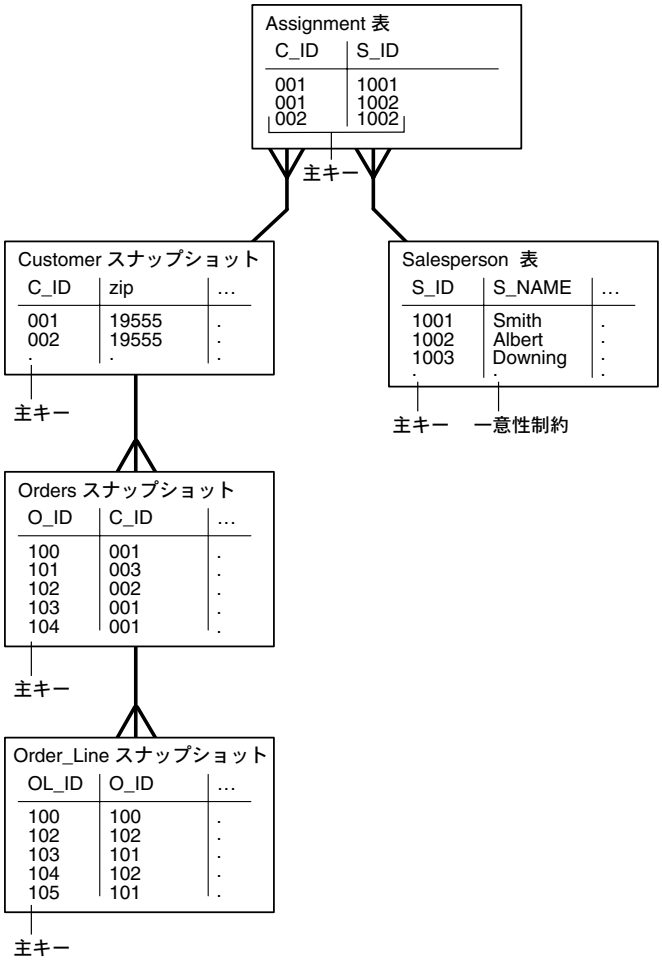
注意： 垂直パーティション化を使用できるのは、Oracle Replication Manager を使用してスナップショット・オブジェクトを配置テンプレートに追加する場合のみです。レプリケーション・マネージメント API を使用している場合は、垂直パーティション化は使用できません。

注意： 垂直パーティションを定義するときは、マスター定義サイトが使用できる必要があります。配置テンプレートに複数のマスター・グループの垂直パーティション化されたスナップショットが含まれている場合は、各グループのマスター定義サイトが使用可能であることが必要です。

割当て表による水平パーティション化

前の項で説明したように、スナップショット・データ・セットはスナップショットの問合せに基づいて定義されます。つまり、ユーザーはスナップショットの定義問合せに適合するデータのみ参照できます。

図 4-7 顧客と営業担当員の関連付け



このことを念頭において副問合せに基づくデータのサブセット化と割当て表を併用すれば、スナップショット環境に加えられた変更を容易に制御できます。

たとえば、図 4-7 に示す顧客と営業担当員の関連付けを考えてみます。この例では、営業担当員は割当て表に基づいて顧客に割り当てられています。営業担当員の採用や退職があった場合は、割当て表の内容を変更すれば既存の顧客を新しい営業担当員に割り当てることができます。副問合せに基づくデータのサブセット化と割当て表を併用すると、一元管理が可能になるのみでなく、高い安全性を保ちながら容易に管理ができます。たとえば、営業担当員 #1001 は他の営業担当員の顧客情報を参照できません。顧客情報に機密データが含まれる場合、これは非常に重要です。

図 4-7 に示す関連付けにおいて、ORDRS スナップショットの定義副問合せが次のように指定されているとします。

```
CREATE SNAPSHOT sales.orders AS
 SELECT * FROM sales.orders@hq.acme.com o
    -- conditions for customers
  WHERE EXISTS
  ( SELECT c_id FROM sales.customer@hq.acme.com c
    WHERE o.c_id = c.c_id
      AND EXISTS
    ( SELECT * FROM sales.assignment@hq.acme.com a
     WHERE a.c_id = c.c_id
     AND EXISTS
     ( SELECT * FROM sales.salesperson@hq.acme.com s
       WHERE s.s_id = ':salesperson_id')));
```

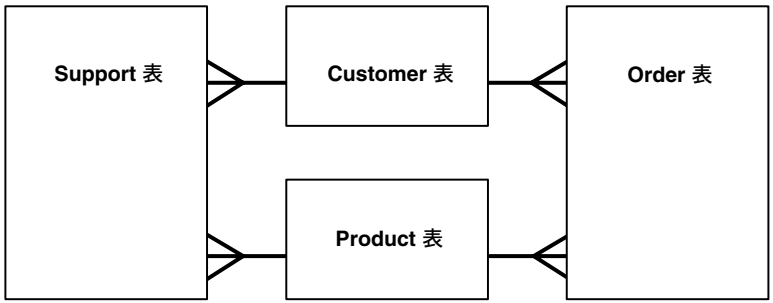
この場合、ORDERS スナップショットには、変数 *:salesperson_id* に指定された営業担当員に割り当てられている顧客の受注データが移入されます。CREATE SNAPSHOT 文の最後の行の変数 *:salesperson_id* に注意してください。

このような柔軟性があるため、管理者は SQL を変更せずに、割当て表を変更することによってスナップショットのデータ・セットを容易に制御できます。たとえば、ある営業担当員に 2 人の新しい顧客が割り当てられた場合、管理者の作業は割当て表でその営業担当員に新しい顧客を割り当てることのみです。スナップショットの次回の高速リフレッシュ後、2 人の顧客のデータは営業担当員のラップトップなどのターゲット・スナップショット・サイトに伝播されます。反対に、指定した営業担当員から 1 人の顧客の割当てを外した場合は、次回のリフレッシュ後にその顧客に関するすべてのデータがスナップショット・サイトから削除され、営業担当員はその情報にアクセスできなくなります。

データ・セット

配置テンプレートを設計するときは、ターゲット・データにアクセスする様々なユーザーのタイプを考慮する必要があります。たとえば、営業担当員と技術者はいずれも顧客情報を必要としますが、両方が販売情報を必要とするわけではありません。

図 4-8 一部のユーザーはすべてのデータを必要とする



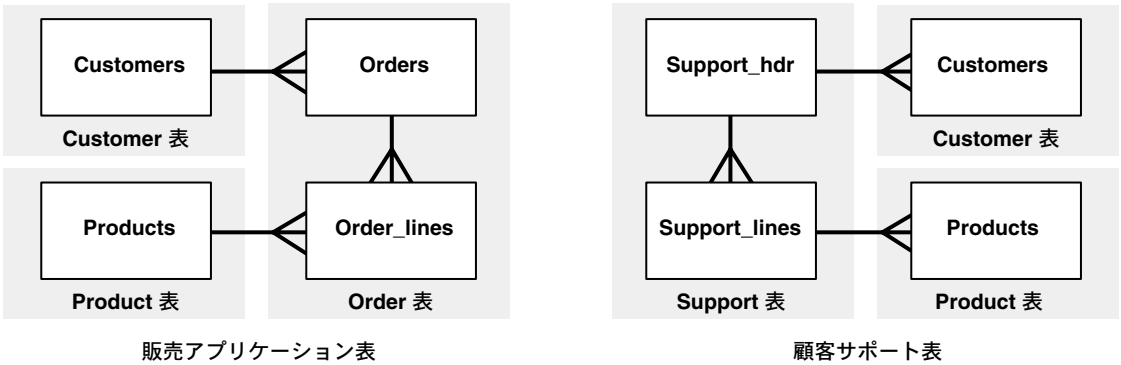
ユーザーが自分に無関係なデータを含む配置テンプレートをインスタンシエートすると余分な記憶領域とリフレッシュ時間が必要となるので、避ける必要があります。

逆に、販売情報と顧客情報の両方を必要とするユーザーが、重複したデータを共有する複数の配置テンプレートをインスタンシエートする必要があることも不都合です。複数のテンプレートをインスタンシエートすると、データ整合性の問題が発生することがあります。つまり、各配置テンプレートが異なるリフレッシュ・グループを使用するため、2 つの配置テンプレートのデータが別々のタイミングでリフレッシュされてデータ整合性の問題が生じる可能性があります。

この場合は、営業担当員、顧客サービス技術者、および両方のデータ・セットを必要とするユーザー向けに、それぞれ 1 つずつ配置テンプレートを作成するのが最善のソリューションです。

時間と DBA の手間をかけずに 3 種類のテンプレートを作成するには、まず両方のデータ・セットを含むテンプレートを作成し、それを 2 つコピーして、それぞれ不要な項目を削除して 2 つのテンプレートを作成するのが最善の方法です。

図 4-9 営業担当員用テンプレートと顧客サービス技術者用テンプレートの相違



もう 1 つの設計上の考慮事項は、パラメータの使用方法です。上の表の多くで Customer ID フィールドが使用される場合、同じパラメータを各テンプレート・オブジェクトで定義できます。同じパラメータを使用すると、デフォルト・パラメータ値を一度定義すると、その定義がインスタンシエーション・プロセスですべてのオブジェクトに使用されます。

副問合せに基づくサブセット化を行うスナップショットで使用する場合は、1 つのテンプレート・パラメータを使用するとさらに便利です。パラメータを 1 つにすることによって、ユーザーは必要な顧客に関するデータのみを受け取ることができます。たとえば、次の CREATE SNAPSHOT 文について考えます。

```
CREATE SNAPSHOT sales.orders AS
 SELECT * FROM sales.orders@hq.acme.com o
    -- conditions for customers
  WHERE EXISTS
  ( SELECT c_id FROM sales.customer@hq.acme.com c
    WHERE o.c_id = c.c_id
      AND EXISTS
    ( SELECT * FROM sales.assignment@hq.acme.com a
     WHERE a.c_id = c.c_id
     AND EXISTS
     ( SELECT * FROM sales.salesperson@hq.acme.com s
       WHERE s.s_id = ':salesperson_id')));
```

```
CREATE SNAPSHOT sales.customer AS
SELECT c_id FROM sales.customer@hq.acme.com c
  -- conditions for customers
  WHERE EXISTS
   ( SELECT * FROM sales.assignment@hq.acme.com a
     WHERE a.c_id = c.c_id
     AND EXISTS
     ( SELECT * FROM sales.salesperson@hq.acme.com s
       WHERE s.s_id = ':salesperson_id')));
```

作成する 2 つのスナップショットには明示的な SALESPERSON_ID フィールドが含まれていませんが、副問合せに基づくサブセット化を行うことによって、パラメータを使用して必要なデータ・セットのみをインスタンシエートできるので、非常に効率的です。1 つのパラメータ（*:salesperson_id*）を使用すると、スナップショットの管理およびインスタンシエーションが、DBA および配置テンプレートを使用するユーザーの両方にとって容易になります。

設計に関するその他の考慮事項

最後に、リモート・スナップショット・サイトで他にどのようなオブジェクトを作成する必要があるかを考えます。具体的には、次の点を検討してください。

- スナップショット・サイトとマスター・サイト間に必要なデータベース・リンクを作成する DDL を含める必要があるかどうか
- スナップショット環境で必要となるトリガーまたはプロシージャの種類
- 非レプリケート・データを格納する表を作成する必要があるかどうか
- 索引を追加する必要があるかどうか

スナップショット作成のローカル制御

配置テンプレートは、スナップショット環境を作成および配布するための最も効率的な方法です。配布先が 2 つまたは 3 つのサイトのみの場合でも、配置テンプレートを使用することによって、個々のサイトにスナップショット環境を作成するよりもはるかに少ないステップでスナップショット環境を作成できます。配置テンプレートを使用すると、一度スナップショット環境を作成すると、必要な数のサイトにそれを配布できます。

ただし、疑問が 1 つ残っています。スナップショット環境を作成して配布するのに配置テンプレートを使用することが最も効率的な方法ならば、どのような場合にリモート・スナップショット・サイトでローカルにスナップショット環境を作成するのかということです。通常、スナップショット・サイトでローカル制御を維持する必要がある場合に、「Snapshot Group Wizard」を使用してスナップショット環境を作成するか、スナップショット・サイトでローカルにスナップショット環境を作成します。

スナップショットの作成をローカル制御すると便利なシナリオの 1 つは、スナップショット・サイトで、受け取るデータを制御することが望ましい場合です。たとえば、通常は読取り専用のスナップショット・サイトである意思決定支援サイト（DSS）の場合、特に便利です。DDS サイトでは複雑な問合せを実行することがありますが、OLTP サイトの速度を低下させないようにしたり、OLTP サイトの DBA をわずらわせないようにするために、ローカル制御が必要です。

スナップショットのローカル制御

配置テンプレートの主な利点の 1 つに、DBA が配置テンプレートの作成を集中して管理できる点があります。しかし、スナップショット・サイトでの制御が必要な場合もあります。

次の場合はローカル制御が必要です。

- スナップショット・サイトに熟練した DBA がいる場合
- スナップショット・サイトが信頼できるサイトである場合
- スナップショット・サイトでデータの水平パーティション化を実行する場合

スナップショット・グループは、スナップショット・サイトの DBA または SQL の知識を持ったシステム・アナリストが Replication Manager の「Snapshot Group Wizard」を使用してスナップショット・サイトでローカルに作成するため、スナップショット・サイトでも制御できます。

ローカル制御を行う最適な例として、次の例を考えます。マルチマスター・レプリケーションではデータの垂直パーティション化および水平パーティション化ができないため、主にデータのパーティション化を実行するために更新可能スナップショット・サイトを作成する場合があります。これらのサイトは通常は安全で、熟練した DBA が存在し、ユーザーやアプリケーションの要件を満たすためにローカル制御を行う必要があります。「Snapshot Group Wizard」またはレプリケーション・マネージメント API を使用して作成したスナップショット・グループでは、安全性の高い更新可能スナップショット・サイトの要件を満たすために必要なローカル制御を行うことができます。

また、前述のように、配置テンプレートを使用してスナップショット環境を作成すると、スナップショット環境のすべてのオブジェクトが同じリフレッシュ・グループに追加されます。このことはほとんどのインストレーションでは問題ありませんが、場合によってはスナップショット・グループ内のオブジェクトをいくつかの異なるリフレッシュ・グループに割り当てる必要があります。

5

競合解消の概念およびアーキテクチャ

この章のトピックは次のとおりです。

- 競合解消の概念
- 競合解消のアーキテクチャ

競合解消の概念

レプリケーションの競合は、複数のサイトから同じデータを同時に更新できるレプリケーション環境で発生します。たとえば、別々のサイトから発生した 2 つのトランザクションによって、同じ行がほとんど同時に更新されたときに競合が発生します。レプリケーション環境を構成するときは、レプリケーションの競合が発生する可能性の有無を考慮する必要があります。システム設計に競合を発生させる可能性がある場合や、競合が発生した場合には、なんらかの方法で競合を解消しない限りシステム・データは収束しません。

一般的に、まず必要なのは競合が発生しないレプリケート環境を設計することです。ほとんどのシステム設計では、いくつかのテクニックを使用することによって、レプリケートするデータの全体または大部分で競合を回避できます。しかし、多くのアプリケーションでは、データの一部を複数のサイトからいつでも更新できる必要があります。その場合、レプリケーションの競合の発生に対処できる必要があります。

以降の項では、レプリケーションの競合に関する情報、レプリケーションの競合を考慮してレプリケーション・システムを設計する方法、レプリケート・システム設計においてレプリケーションの競合を回避する方法、設計で競合を回避できない場合に競合を検出および解消する方法について説明します。

データ要件およびアプリケーション要件の理解

どのようなタイプのデータベース・アプリケーションおよびサポート・データベースを設計する場合でも、データベースまたはアプリケーション自体を設計する前にアプリケーション要件を理解することは非常に重要です。たとえば、各アプリケーションは、機能境界や依存性（受注入力、出荷および請求など）が正しく定義され、モジュール化されている必要があります。さらに、サポート・データベースのデータを正規化して、アプリケーション・システムのモジュール間の隠れた依存性を軽減する必要があります。

レプリケーション環境で操作するデータベースを作成する場合は、基本的なデータベース設計のルールの他に考慮が必要な要件があります。まず、レプリケート・データを扱うアプリケーションの一般的な要件を考慮します。たとえば、一部のアプリケーションは読取り専用スナップショットの使用でも十分に対応できるので、結果的にレプリケーションの競合の発生を完全に回避できます。これに対して、大部分のレプリケート・データが読取り専用で、一部のデータ（たとえば、1 つまたは 2 つの表、あるいは特定の表内の 1 つまたは 2 つの列）をすべてのレプリケーション・サイトで更新可能にする必要があるアプリケーションもあります。この場合は、レプリケーションの競合が発生したときにレプリケート・データの整合性が完全に保たれるように、競合解消方法を決定する必要があります。

例

競合を考慮したレプリケート・データベースの設計方法をよく理解するために、競合の検出および解消が可能な場合と不可能な場合がある、次のような環境を考えてみます。

- 通常、同じ商品に対して複数の予約を受け付けることができない予約システムでは、競合を解消できません。たとえば、コンサートの特定の座席を予約するときには競合を解消する方法がないため、予約システムの異なるレプリカにアクセスする異なるエージェントが、複数の顧客のために同じ座席を予約できません。
- 顧客管理システムでは、通常、競合を解消できます。たとえば、複数の営業担当員がレプリケート環境内の異なるデータベースの顧客住所情報をメンテナンスできます。競合が発生しても、レコードに最新の更新内容を適用することで、更新の競合を解消できます。

レプリケーションの競合のタイプ

レプリケート・データベース環境で発生するデータ競合には、更新競合、一意性競合および削除競合の、3 つのタイプの競合があります。この章の後半部分で述べるように、レプリケーション環境で発生する可能性が高いのは更新競合です。ただし、一意性競合および削除競合にも対処できるように準備する必要があります。これらの 3 つのタイプの競合を回避できるようにデータベースを設計することをお薦めします。

更新競合

更新競合は、ある行に対する更新のレプリケーションが同じ行に対する別の更新と競合したときに発生します。異なるサイトから発生した 2 つの異なるトランザクションによって、同じ行がほとんど同時に更新されたときに発生することがあります。

一意性競合

一意性競合は、行のレプリケーションによってエンティティの整合性（PRIMARY KEY 制約または UNIQUE 制約など）が失われるときに発生します。たとえば、2 つの異なるサイトから発生した 2 つのトランザクションで、同じ主キー値を使用してそれぞれの表レプリカに行を挿入した場合について考えます。この場合、トランザクションのレプリケーションが原因で一意性競合が発生します。

削除競合

削除競合は、異なるサイトから発生した 2 つのトランザクションのうち、1 つのトランザクションで行が削除され、もう 1 つのトランザクションで同じ行に対して更新または削除が行われたときに発生します。これは、更新または削除の対象となる行が存在しなくなるためです。

データの競合およびトランザクションの順序

競合

順序競合は、マスター・サイトが 3 つ以上あるレプリケーション環境で発生します。なんらかの理由でマスター・サイト X への伝播がブロックされた場合、レプリケート・データへの更新を他のマスター・サイト間で伝播できます。伝播を再開したときに、これらの更新がその他のマスターで発生した順序と異なる順序でサイト X に伝播されると、更新競合が発生する可能性があります。デフォルトでは、発生した競合はエラー・ログに記録され、依存関係にあるトランザクションが伝播および適用された後で再実行できます。順序競合の例は、5-9 ページの表 5-1 を参照してください。

マスター・サイトが 3 つ以上あるレプリケーション環境でデータ収束を保証するには、マスター・サイトがいくつあっても確実にデータ収束を行える競合解消方法（最も新しいタイムスタンプ、最小値[1]、最大値[1]、優先グループ[1]、加算[1]）を選択する必要があります。

参照整合性

サポート・データがリモート・サイトに正常に伝播されなかった場合、レプリケート・トランザクションが順不同に適用されて、データ競合の他に参照整合性の問題が発生する可能性もあります。新規顧客が受注部門に電話をかけた場合を考えます。この場合、顧客レコードが作成され、受注が設定されます。顧客データより先に受注データがリモート・サイトに伝播された場合、受注によって参照される顧客がリモート・サイトに存在しないため、参照整合性のエラーが発生します。

参照整合性エラーが発生した場合は、エラーが生じたトランザクションをサポート・データがリモート・サイトに伝播された後で再度実行すれば、簡単に解決できます。

[1] 競合解消方法を実行すると、一定の条件が存在する限り、マスター・サイトがいくつあってもデータ収束が保証されます。詳細は、5-10 ページの「競合解消のアーキテクチャ」で説明している競合解消方法を参照してください。

競合の検出

レプリケーション・システムの各マスター・サイトは、レプリケーションの競合が発生するとそれを自動的に検出し、解消します。たとえば、あるマスター・サイトがシステム内の別のマスター・サイトに遅延トランザクション・キューをプッシュするときに、レプリケーションの競合があれば、その競合は呼び出された受信サイトのリモート・プロシージャによって自動的に検出されます。

スナップショット・サイトが対応するマスター・サイトに遅延トランザクションをプッシュすると、受信マスター・サイトで競合の検出および解消が行われます。スナップショット・サイトでは、スナップショットのリフレッシュを実行することによってデータをリフレッシュします。このリフレッシュの仕組みによって、リフレッシュ完了後には、競合解消結果を含めたスナップショットのデータが、対応するマスターのデータと同一になります。したがって、スナップショット・サイトでレプリケーションの競合の検出または解消を実行する必要はありません。

異なるタイプの競合の検出方法

レプリケーション・システムの受信マスター・サイトでは、次のような更新競合、一意性競合および削除競合が検出されます。

- レプリケート行の古い値（変更前の値）と受信サイトの同じ行のカレント値が異なる場合、受信サイトで更新競合が検出されます。
- レプリケート行の INSERT または UPDATE の実行中に一意制約違反が発生した場合、受信サイトで一意性競合が検出されます。
- 行の主キーが存在しないために受信サイトで UPDATE 文または DELETE 文の対象の行が見付からない場合、受信サイトで削除競合が検出されます。

注意： 行の更新競合を検出および解消するには、伝播サイトから受信サイトに行の新旧の値に関するデータを送信する必要があります。パフォーマンスを最大にするには、更新競合の検出および解消のサポートに使用するデータの量をチューニングします。詳細は、5-26 ページの「パフォーマンス・メカニズム」を参照してください。

競合検出時の行の識別

レプリケーションの競合を正確に検出するには、データ・レプリケーション時に別々のサイトの対応する行を一意に識別し、一致させる必要があります。通常、Oracle のレプリケーション機能では、表の主キーを使用して表内の行を一意に識別します。表に主キーがない場合は、代替キーを指定します。代替キーとは、データ・レプリケーション時に表内の行を識別するために使用する列または列のセットです。

注意： アプリケーションによって表の識別列を更新しないでください。これにより、行を識別してレプリケート・データの整合性を保つことができます。

競合解消

競合が検出された場合、すべてのサイトでデータが収束するまで競合を解消することが重要です。Oracle には、更新競合を解消する事前作成の競合解消方法が用意されています。通常は、この方法を使用することで、様々なレプリケーション環境のデータ収束を保証できます。また、一意性競合に対する競合解消方法もありますが、これはデータ収束を保証しません。

削除競合または順序競合については、事前作成の競合解消方法はありません。ただし、ビジネス・ルールに固有の競合解消方法を独自に作成できます。データ収束を保証しない競合解消方法（特に、一意性競合および削除競合の解消方法）を作成する場合、通知機能を作成して DBA にデータ収束を手動で実現するように依頼する必要があります。

事前作成の競合解消方法またはユーザー定義の競合解消方法のいずれを使用する場合も、競合が検出されると即時に適用されます。定義した競合解消方法で競合を解消できない場合は、その競合はエラー・キューにロギングされます。

競合解消が 1 つの時点で失敗するのを回避するために、別の競合解消方法を定義して、主要な競合解消方法の予備を作成できます。たとえば、タイムスタンプが同一であるため、最も新しいタイムスタンプによる競合解消方法では競合を解消できない場合（このような状況が発生することはほとんどにありません）、サイト優先の競合解消方法を定義できます。これにより、タイムスタンプを使用せずにデータ競合を解消できます。

競合の回避

Oracle ではデータ競合を解消する強力な競合解消方法がありますが、レプリケート・データベースやフロントエンド・アプリケーションの設計時にまず優先する必要があるのは、データ競合を回避することです。以降の項では、レプリケーション競合の一部または全部を回避するためのテクニックを簡単に説明します。

プライマリ・サイトおよび動的サイト所有権データ・モデル

レプリケーションの競合の発生を回避する 1 つの方法として、システム内でレプリケート・データに同時更新アクセスできるサイトの数を制限する方法があります。この方法をサポートするレプリケート・データ所有権モデルとして、プライマリ・サイト所有権および動的サイト所有権という 2 つのモデルがあります。

プライマリ・サイト所有権 プライマリ所有権は、読取り専用レプリケーション環境でサポートされるレプリケート・データ・モデルです。プライマリ所有権によって、一連のレプリケート・データに更新アクセスできるサーバーは 1 つに限定されるため、すべてのタイプのレプリケーションの競合が発生しません。

表レベルでデータの所有権を制御するかわりに、アプリケーションで水平パーティション化および垂直パーティション化の機能を使用すると、データの静的所有権をより細かく設定できます。たとえば、サイト別の基準でレプリケート表の特定の列または行に更新アクセスできます。

動的サイト所有権 動的所有権レプリケート・データ・モデルは、プライマリ・サイト所有権よりも制限の少ないモデルです。動的所有権の場合、データ・レプリカを更新する権利はサイトからサイトへ移動しますが、ある時点で特定のデータに更新アクセスできるサイトは 1 つしかありません。ワークフロー・システムに動的所有権の概念がわかりやすく示されています。たとえば、関連する部門のアプリケーションでは、注文を更新できる時期を判断するために、製品注文のステータス・コード（ENTERABLE、SHIPPABLE、BILLABLE など）を使用できます。

> **関連項目：** 動的所有権データ・モデルの使用方法の詳細は、6-11 ページの「動的所有権の競合回避の使用方法」を参照してください。

特定のタイプの競合の回避

プライマリ・サイト所有権および動的所有権データ・モデルを使用するとアプリケーション要件に対して制限が多すぎる場合には、共有所有権データ・モデルを使用する必要があります。その場合でも、通常は、特定のタイプの競合を回避するためにいくつかの簡単な方法を使用する必要があります。

一意性競合の回避 一意性競合が発生しないようにレプリケーション環境を構成するのは非常に簡単です。たとえば、順序が互いに重複しない一連の順序番号を生成するように、各サイトで順序を作成できます。ただし、この方法を使用する場合、サイト数またはレプリケート表内の項目数が増加するほど問題が多く発生します。

また、複合主キーの一部として一意のサイト識別子を追加する方法もあります。

最後に、Oracle8*i* リリース 8.1.5 以降では、SYS_GUID () ファンクションを使用してグローバルに一意な値を選択する方法も利用できます。選択した値を主キー（一意の値）として使用すると、一意性競合をグローバルに回避できます。

注意： 順序は、レプリケーションの有効なオブジェクト型ではありません。したがって、各サイトで順序を作成する必要があります。

関連項目： 詳細は、2-18 ページの「順序」を参照してください。

削除競合の回避 削除競合は、すべてのレプリケート・データ環境で必ず回避する必要があります。一般的に、非同期の共有所有権データ・モデル環境で動作するアプリケーションでは、DELETE 文を使用して行を削除しないでください。かわりに、アプリケーションで行に削除のマークを付けて、論理的に削除された行をプロシージャ・レプリケーションを使用して定期的にパージするようにシステムを構成できます。

関連項目： 削除を回避する表の作成方法およびマークを付けた行をパージするレプリケート・プロシージャの作成方法は、『Oracle8*i* レプリケーション・マネージメント API リファレンス』の第 6 章の「削除競合の競合回避方法の作成」の項を参照してください。

更新競合の回避 レプリケーション・システムでは、一意性競合と削除競合の発生を回避するように試みた後、さらに更新競合の発生数を少なく抑える必要があります。ただし、共有所有権データ・モデルでは、更新競合をすべて回避することはできません。更新競合をすべて回避できない場合は、発生する可能性のあるレプリケーションの競合のタイプを正確に理解し、競合が発生したときに解消できるようにシステムを構成する必要があります。

順序競合の回避 順序競合は、可能な限り、回避するかまたは自動的に解消するのが最善です。たとえば、順序競合が発生する可能性のあるマルチマスター構成で収束を保証する競合解消方法を選択します。

表 5-1 の例は、3 つのマスター・サイトでどのような場合に順序競合が発生するかを示しています。マスター・サイト A の優先順位は 30、マスター・サイト B の優先順位は 25、マスター・サイト C の優先順位は 10 です。*x* は、サイト優先順位による競合解消方法が割り当てられている列グループ内の特定の行の列です。優先順位の値が最も高いサイトに、最も高い優先順位が割り当てられています。優先順位値は任意の Oracle 番号で、連続する整数でなくてもかまいません。

表 5-1 例： サイト優先順位による競合解消方法を使用した場合の順序競合

時間	アクション	サイト A	サイト B	サイト C
1	すべてのサイトで x=2 に更新され、一致しています。	2	2	2
2	サイト A で x=5 に更新します。	5	2	2
3	サイト C が使用不能になります。	5	2	ダウン
4	サイト A がサイト B に更新をプッシュします。 サイト A とサイト B が x=5 で一致します。 サイト C は使用不能のままです。 更新トランザクションはサイト A のキューに残っています。	5	5	ダウン
5	サイト C が x=2 で使用可能になります。 サイト A とサイト B が x=5 で一致します。	5	5	2
6	サイト B で x=5 を x=7 に更新します。	5	7	2
7	サイト B がトランザクションをサイト A にプッシュします。 サイト A とサイト B が x=7 で一致します。 サイト C は x=2 のままです。	7	7	2
8	サイト B がトランザクションをサイト C にプッシュします。 サイト C の古い値は x=2 です。 サイト B の古い値は x=5 です。 競合が検出され、サイト C の優先順位（10）より高い優先順位（25）を持つサイト B の更新を適用することで、競合が解消されます。 すべてのサイトが x=7 で一致します。	7	7	7
9	サイト A がそのトランザクション（x=5）を正常にサイト C にプッシュします。 競合が検出され、サイト C の優先順位（10）より高い優先順位（25）を持つサイト B の更新を適用することで、競合が解消されます。 サイト C（x=7）がサイト A の古い値（x=2）と一致しません。 サイト A はサイト C（10）より高い優先順位（30）を持ちます。 サイト A（x=5）の古い更新を適用することで、競合が解消されます。 この競合処理の順序が原因で、サイトのデータは収束しません。	7	7	5

競合解消のアーキテクチャ

レプリケーション環境に競合解消方法をインプリメントするときに、アーキテクチャ上のメカニズムやプロセスを意識することはほとんどありません。この項では、競合解消に関連するいくつかのサポート・メカニズムについて説明します。ただしこの項では、Oracle の事前作成の競合解消方法を異なる側面から説明することに重点を置いています。

サポート・メカニズム

Oracle の競合解消に関連する最も重要なメカニズムは、列グループです。更新競合の検出および解消はすべて、列グループに基づいて行われます。また、エラー・キューには、レプリケーション環境の競合検出アクティビティを監視するための重要な情報が記録されています。

列グループ

Oracle では、更新競合を検出および解消するために列グループを使用します。列グループは、レプリケート表の 1 つ以上の列を論理的にグループ化したものです。レプリケート表の各列は、1 つの列グループの一部になります。マスター定義サイトでレプリケート表を構成するときに、列グループを作成して、各グループに列および対応する競合解消方法を割り当てることができます。

レプリケート表に複数の列グループが含まれている場合、更新競合が発生しているかどうかの分析は各列グループごとに個別に行われます。たとえば、列 A1、A2 および A3 が列グループ A に含まれ、列 B1、B2 および B3 が列グループ B に含まれている場合を考えます。ユーザー 01 がサイト 01 で行 1 の列 A1 を更新し、ユーザー 02 がサイト 02 で行 1 の列 B2 を更新した場合、これらの更新内容はレプリケートされ、データ競合は発生しません。

複数のサイトで同じ行が更新されたことで、両方の更新内容のレプリケーションに影響があります。しかし、各更新は個別に分析される別々の列グループで発生しており、Oracle では各グループを個別に分析しているので、競合は発生しません。ユーザー 02 がサイト 02 で行 1 の列 A2 を更新した場合は、競合が発生します。

複数の列グループでのデータ整合性の保証 列グループを使用すると、データのタイプに応じて異なる競合解消方法を指定できます。たとえば、数値データの多くには算術的な解消方法が適しており、文字データの多くにはタイムスタンプによる解消方法が適しています。ただし、列グループの列を選択するときには、十分に考えた上で列をグループ化することが重要です。表内の複数の列で相互に整合性を保つ必要がある場合は、データの整合性を保証するためにそれらの列を同じ列グループに入れます。たとえば、顧客表の郵便番号列で 1 つの解消方法を使用し、都市名列には別の解消方法を使用している場合、サイトは都市名に一致する郵便番号に収束します。したがって、一般的には、競合解消方法が住所全体に対して適用されるように、1 つの住所のすべての構成要素を 1 つの列グループに入れる必要があります。

シャドウ列グループ デフォルトでは、各レプリケート表にシャドウ列グループがあります。表のシャドウ列グループには、特定の列グループ内にないすべての列が含まれます。表の

シャドウ・グループには、競合解消方法を割り当てることはできません。したがって、列に競合解消方法を使用する必要がある場合は、列グループに必ずその列を入れてください。

エラー・キュー

競合解消方法によってデータ競合を解消できない場合、または競合解消方法を定義していなかった場合は、エラー・キューにデータ競合に関する情報が記録されます。

> **関連項目：** エラー・キューの詳細は、2-20 ページの「エラー・キュー」を参照してください。

更新競合の一般的な解消方法

Oracle では更新競合の解消方法として 8 つの事前作成の方法を用意していますが、このうち最も一般的にインプリメントされるのは、最も新しいタイムスタンプによる競合解消方法および上書きによる競合解消方法です。

この 2 つの競合解消方法が最も一般的なのは、概念的に扱いやすく、適切な環境ではデータ収束が保証されるためです。最も新しいタイムスタンプによる競合解消方法および上書きによる競合解消方法の詳細は、以降の 2 つの項で説明します。

表 5-2 更新競合の一般的な解消方法のデータ収束プロパティ

解消方法	1 つのマスター・サイト	任意の数のマスター・サイト
最も新しいタイムスタンプ	収束します（予備の方法を設定する場合）	収束します（予備の方法を設定する場合）
上書き	収束します	収束しません

最も新しいタイムスタンプ

最も新しいタイムスタンプによる方法を使用すると、更新発生時のタイムスタンプによって識別された最も新しい更新を原因とする競合を解消できます。

最も新しいタイムスタンプを使用した更新競合の解消方法の例を次に示します。

1. フェニックスの顧客が地元の営業担当員に電話をかけ、自分の住所情報を更新します。
2. 電話を切った後で、営業担当員に伝えた郵便番号が間違っていたことに顧客が気付きます。
3. 顧客は営業担当員に電話して正しい郵便番号を伝えようとしますが、担当者に電話がつながりません。
4. 顧客はニューヨーク本社に電話します。フェニックス・サイトではなく、ニューヨーク・サイトで住所情報が正しく更新されます。
5. ニューヨーク本社とフェニックス営業所を接続するネットワークが一時的にダウンします。
6. ニューヨークとフェニックスを接続するネットワークが回復すると、同じ住所に対する 2 つの更新が認識され、各サイトでの競合が検出されます。
7. 最も新しいタイムスタンプによる方法を使用して最新の更新内容が選択され、郵便番号が正しい住所が適用されます。

ターゲット環境　最も新しいタイムスタンプによる競合解消方法は、複数のマスター・サイトがあるレプリケーション環境でデータが収束するように機能します。タイムスタンプは常に増加しているので、この方法は複数のマスター・サイト間でのデータ収束を保証できる数少ない競合解消方法の 1 つです。また、任意の数のスナップショットがある環境でもうまく機能します。

サポート・メカニズム　タイムスタンプによる方法を使用するには、レプリケート表内の DATE 型の列を指定する必要があります。アプリケーションで列グループ内の列を更新するときは、ローカル SYSDATE で指定されたタイムスタンプ列の値も更新する必要があります。他のサイトから適用される変更については、タイムスタンプ値を発行元サイトのタイムスタンプ値に設定する必要があります。

注意：　タイムスタンプによる競合解消方法を使用するときは、2 つのサイトのタイムスタンプが同じ場合にサイト優先順位などの予備の方法がコールされるように設定する必要があります。

タイムスタンプの設定に関する問題

タイムスタンプによる解消方法を使用する場合は、レプリケート・データを管理している異なるサイトの標準時間帯を慎重に考慮してください。たとえば、レプリケート環境が複数の標準時間帯にわたっている場合、システムを使用するアプリケーションは、すべてのタイムスタンプをグリニッジ標準時（GMT）などの共通の標準時間帯に変換する必要があります。さらに、システム内の 2 つのサイトのシステム時計が正しく同期化されていない場合、タイムスタンプの比較が不正確になり、アプリケーション要件を満たすことができない可能性があります。

最も古いタイムスタンプまたは最も新しいタイムスタンプによる更新競合の解消方法を使用する場合、タイムスタンプ列のメンテナンスには 2 つの方法があります。

- 各アプリケーションに、タイムスタンプを同期化する機能を組み込む方法
- すべてのアプリケーションのタイムスタンプを自動的に同期化させる、レプリケート表のトリガーを作成する方法

クロックは、秒数を増加する値としてカウントします。タイムスタンプのメカニズムを正しく設計し、2 つのサイトのタイムスタンプが同一である場合の予備の方法を設定したと想定すると、最も新しいタイムスタンプによる方法で収束が保証されます（最大値による方法と同様）。ただし、最も古いタイムスタンプによる方法では、複数のマスターの収束は保証されません。

最も新しいタイムスタンプによる方法の実現　Replication Manager を使用して最も新しいタイムスタンプによる競合解消方法を定義する方法は、Replication Manager のオンライン・ヘルプを参照してください。レプリケーション・マネージメント API を使用した場合の定義方法は、『Oracle8*i* レプリケーション・マネージメント API リファレンス』の第 6 章の「タイムスタンプによる方法」の項を参照してください。

上書き

上書きによる競合解消方法では、発行元サイトの値は無視されるため、複数のマスター・サイトでのデータ収束が保証されることはありません。この方法は、1 つのマスター・サイトおよび複数のスナップショット・サイトで使用するように指定されています。複数のマスター・サイトでこの方法を使用することもできますが、データ収束は保証されないので、なんらかのユーザー定義による通知機能とともに使用する必要があります。

たとえば、主に問合せに使用する 1 つのマスター・サイトがあり、すべての更新をそのスナップショット・サイトで実行する場合は、上書きによる競合解消方法を選択できます。上書きによる方法は、次の場合にも役に立ちます。

- 最大の目的がデータ収束である場合
- マスター・サイトが 1 つしかない場合
- 更新の選択について、ビジネス・ルールが特にない場合
- 複数のマスター・サイトがある環境で、DEFERROR ビューでロギングした競合の解消をローカルの DBA にまかせる方法ではなく、データが正しく適用されるように確認する担当者への通知機能を使用する場合

上書きによる方法では、接続先サイトのカレント値を、発行元サイトの新しい値で上書きします。

ターゲット環境　上書きによる競合解消方法では、1 つのマスター・サイトと任意の数のスナップショットから構成されるレプリケーション環境でのデータ収束が保証されます。この点を考慮すると、上書きによる競合解消方法は大量配布環境で理想的な方法です。

競合が検出された場合、スナップショット・サイトの値が使用されます。つまり、最後にリフレッシュが実行されたスナップショットに優先順位が与えられます。

サポート・メカニズム　上書きによる競合解消方法では、その他のサポート・メカニズムは必要ありません。

実施の概要　Replication Manager を使用して上書きによる競合解消方法を定義する方法は、Replication Manager のオンライン・ヘルプを参照してください。レプリケーション・マネージメント API を使用した場合の定義方法は、『Oracle8*i* レプリケーション・マネージメント API リファレンス』の第 6 章の「上書きによる方法と廃棄による方法」の項を参照してください。

更新競合のその他の解消方法

最も新しいタイムスタンプまたは上書き（あるいはその両方）による競合解消方法を使用してもレプリケーション環境で発生した更新競合を解消できない場合は、その他の 6 つの事前作成の競合解消方法を使用できます。

表 5-3 更新競合のその他の解消方法のデータ収束プロパティ

解消方法	1 つのマスター・サイト	任意の数のマスター・サイト
加算	収束する	収束する
平均	収束する	収束しない
廃棄	収束する	収束しない
最も古いタイムスタンプ	収束する（予備の方法を設定する場合）	収束しない
最大値	収束する	収束する（列値は常に増加する必要あり）
最小値	収束する	収束する（列値は常に減少する必要あり）
優先グループ	収束する	収束する（順序付けられた更新値がある場合）
サイト優先順位	収束する	収束しない

加算

加算による競合解消方法は、数値型の単一列のみで構成される列グループで機能します。競合が発生した場合、いずれか一方の値を選択するのではなく、2 つの値の差がカレント値に加算されます。

加算による方法では、発行元サイトの古い値と新しい値の差を、接続先サイトのカレント値に加算します。

```
current value = current value + (new value - old value)
```

この方法では、マスター・サイトやスナップショット・サイトがいくつあっても値は収束します。

ターゲット環境　加算による競合解消方法では、最も適切なデータを選択するのではなくデータを収束するように設定されています。この方法は、預金の預入れや引出しが頻繁に行われてデータ競合が発生する可能性がある金融環境で非常に便利です。収支では、いずれか一方の値を選択するのではなく、データを収束させることが重要です（引出しの値ではなく預入れの値を常に選択できればよいのですが）。

サポート・メカニズム　上書きによる競合解消方法では、その他のサポート・メカニズムは必要ありません。

加算による方法の実現　Replication Manager を使用して加算による競合解消方法を定義する方法は、Replication Manager のオンライン・ヘルプを参照してください。レプリケーション・マネージメント API を使用した場合の定義方法は、『Oracle8*i* レプリケーション・マネージメント API リファレンス』の第 6 章の「加算による方法と平均による方法」の項を参照してください。

平均

平均による競合解消方法は、加算による場合と同様に、数値型の単一列のみで構成される列グループで機能します。この方法では、カレント値に 2 つの値の差を追加するのではなく、カレント値と新しい値の平均を計算することで競合を解消します。

平均による競合解消方法では、発行元サイトの新しい列値と接続先サイトのカレント値の平均をとります。

```
current value = (current value + new value)/2
```

レプリケート環境に複数のマスター・サイトが存在する場合、平均による方法では収束を保証できません。

ターゲット環境　平均による方法では、複数のマスター・サイトが存在するレプリケーション環境でのデータ収束を保証できないので、この方法は、1 つのマスター・サイトと任意の数の更新可能スナップショットから構成される大量配布環境でインプリメントするのが理想的です。

平均による方法は、いずれか一方の値を選択するのではなく、2 つの値の平均を取る科学アプリケーションで使用すると便利です（たとえば、平均温度や平均体重を計算する場合などです）。

サポート・メカニズム　平均による競合解消方法では、その他のサポート・メカニズムは必要ありません。

平均による方法の実現　Replication Manager を使用して平均による競合解消方法を定義する方法は、Replication Manager のオンライン・ヘルプを参照してください。レプリケーション・マネージメント API を使用した場合の定義方法は、『Oracle8*i* レプリケーション・マネージメント API リファレンス』の第 6 章の「加算による方法と平均による方法」の項を参照してください。

廃棄

廃棄による競合解消方法では、発行元サイトの値は無視されるため、複数のマスター・サイトでのデータ収束は保証されません。この方法は、1 つのマスター・サイトと複数のスナップショット・サイトの間で使用するか、またはなんらかのユーザー定義による通知機能とともに使用してください。

たとえば、主に問合せに使用する 1 つのマスター・サイトがあり、すべての更新をそのスナップショット・サイトで実行する場合は、廃棄による競合解消方法を選択できます。廃棄による方法は、次の場合にも役に立ちます。

- 最大の目的がデータ収束である場合
- マスター・サイトが 1 つしかない場合
- 更新の選択について、ビジネス・ルールが特にない場合
- 複数のマスター・サイトがある環境で、DEFERROR ビューでロギングした競合の解消をローカルの DBA にまかせるのではなく、データが正しく適用されることを確認する担当者への通知機能を使用する場合

ターゲット環境　廃棄による競合解消方法は、1 つのマスター・サイトと任意の数のスナップショット・サイトから構成される大量配布モデルに最も適しています。競合が検出された場合、スナップショット・サイトの値は無視されます。つまり、最初にリフレッシュが実行されたスナップショットに優先順位が与えられます。

サポート・メカニズム　廃棄による競合解消方法では、その他のサポート・メカニズムは必要ありません。

廃棄による方法の実現　Replication Manager を使用して廃棄による競合解消方法を定義する方法は、Replication Manager のオンライン・ヘルプを参照してください。レプリケーション・マネージメント API を使用した場合の定義方法は、『Oracle8*i* レプリケーション・マネージメント API リファレンス』の第 6 章の「上書きによる方法と廃棄による方法」の項を参照してください。

最も古いタイムスタンプ

最も古いタイムスタンプによる方法を使用すると、更新発生時のタイムスタンプによって識別された最も古い更新を原因とする競合を解消できます。

ターゲット環境　最も古いタイムスタンプによる競合解消方法は、1 つのマスター・サイトと任意の数のスナップショットから構成されるレプリケーション環境でデータが収束するように機能します。タイムスタンプは常に増加しているので、最も古いタイムスタンプによる競合解消方法では、複数のマスター・サイトがあるレプリケーション環境のデータ収束は保証できません。この方法は、2 つのトランザクションのタイムスタンプが同じである場合に備えて予備の競合解消方法を作成してある場合は、スナップショットの数にかかわらず機能します。

サポート・メカニズム　タイムスタンプによる方法を使用するには、レプリケート表内の DATE 型の列を指定する必要があります。アプリケーションで列グループ内の列を更新するときは、ローカル SYSDATE で指定されたタイムスタンプ列の値も更新する必要があります。他のサイトから適用される変更については、タイムスタンプ値は発行元サイトのタイムスタンプ値に設定する必要があります。必ず、5-13 ページの「タイムスタンプの設定に関する問題」を確認してください。

注意：　タイムスタンプによる競合解消方法を使用するときは、2 つのサイトのタイムスタンプが同じ場合にサイト優先順位などの予備の方法がコールされるように設定する必要があります。

最も古いタイムスタンプによる方法の実現　Replication Manager を使用して最も古いタイムスタンプによる競合解消方法を定義する方法は、Replication Manager のオンライン・ヘルプを参照してください。レプリケーション・マネージメント API を使用した場合の定義方法は、『Oracle8*i* レプリケーション・マネージメント API リファレンス』の第 6 章の「タイムスタンプによる方法」の項を参照してください。

最大値

レプリケーション機能によって列グループに競合が検出され、最大値による競合解消方法がコールされると、列グループの指定した列について、発行元サイトの新しい値と接続先サイトのカレント値が比較されます。最小値による競合解消方法を選択する場合は、この列を指定する必要があります。

指定した列の新しい値がカレント値より大きい場合は、発行元サイトの列グループの値が接続先サイトに適用されます。このとき、その行のその他のエラーはすべて解消されていると想定します。指定した列の新しい値がカレント値より小さい場合は、列グループのカレント値を変更しないと競合は解消されます。

注意： 指定した列の 2 つの値が同じ場合（たとえば、指定した列が競合を発生させた列でない場合）、競合は解消されず、列グループの列の値は変更されません。この場合は、予備の競合解消方法を指定してください。

列グループ内の列のデータ型に制約はありません。複数のマスター・サイトのデータ収束が保証されるのは、列の値が常に増加している場合のみです。

注意： 競合解消が制約によって妨げられる可能性があるため、CHECK 制約を使用して常に増加する値を制限しないでください。

ターゲット環境　最大値による競合解消方法を定義済で、競合解消に使用するターゲット列の値がすべてのサイトで常に増加している場合は、マスター・サイトやスナップショット・サイトの数にかかわらずデータ収束は保証されます。

サポート・メカニズム　最大値による競合解消方法では、その他のサポート・メカニズムは必要ありません。

最大値による方法の実現　Replication Manager を使用して最大値による競合解消方法を定義する方法は、Replication Manager のオンライン・ヘルプを参照してください。レプリケーション・マネージメント API を使用した場合の定義方法は、『Oracle8*i* レプリケーション・マネージメント API リファレンス』の第 6 章の「最小値による方法と最大値による方法」の項を参照してください。

最小値

レプリケーション機能によって列グループに競合が検出され、最小値による競合解消方法がコールされると、列グループの指定した列について、発行元サイトの新しい値と接続先サイトのカレント値が比較されます。最小値による競合解消方法を選択する場合は、この列を指定する必要があります。

指定した列の新しい値がカレント値より小さい場合は、発行元サイトの列グループの値が接続先サイトに適用されます。このとき、その行のその他のエラーはすべて解消されていると想定します。指定した列の新しい値がカレント値より大きい場合は、列グループのカレント値を変更しないと競合は解消されます。

注意： 指定した列の 2 つの値が同じ場合（たとえば、指定した列が競合を発生させた列ではない場合）、競合は解消されず、列グループの列の値は変更されません。この場合は、予備の競合解消方法を指定してください。

列グループ内の列のデータ型に制約はありません。複数のマスター・サイトのデータ収束が保証されるのは、列の値が常に減少している場合のみです。

注意： 競合解消が制約によって妨げられる可能性があるため、CHECK 制約を使用して常に増加する値を制限しないでください。

ターゲット環境　最小値による競合解消方法を定義済で、競合解消に使用するターゲット列の値がすべてのサイトで常に減少している場合は、マスター・サイトやスナップショット・サイトの数にかかわらずデータ収束は保証されます。

サポート・メカニズム　最小値による競合解消方法では、その他のサポート・メカニズムは必要ありません。

最小値による方法の実現　Replication Manager を使用して最小値による競合解消方法を定義する方法は、Replication Manager のオンライン・ヘルプを参照してください。レプリケーション・マネージメント API を使用した場合の定義方法は、『Oracle8*i* レプリケーション・マネージメント API リファレンス』の第 6 章の「最小値による方法と最大値による方法」の項を参照してください。

優先グループ

優先グループを使用すると、特定の列に格納される可能性がある各値に優先順位レベルを割り当てることができます。競合が検出されると、"priority" 列の値が小さい表が、より高い優先順位値を持つ表のデータによって更新されます。したがって、値が大きいことは優先順位が高いことを意味します。

図 5–1 に示すように、DBA_REPPRIORITY ビューに、各優先グループのメンバーに割り当てられた優先順位レベル（"priority" 列に格納される値）が表示されます。"priority" 列に格納される可能性があるすべての値に対して、優先順位を指定する必要があります。

図 5-1 優先グループの使用方法

Customer表

custno	name	addr1	addr2	site
153	Kelly	104 First St.	Jones, NY	new_york.world
118	Klein	22 Iris Ln.	Planes, NE	houston.world
121	Lee	71 Blue Ct.	Aspen, CO	houston.world
204	Potter	181 First Av.	Aspen, CO	houston.world
.	.	.	.	.
.	.	.	.	.
.	.	.	.	.

RepPriorityビュー

...	priority-group	priority	...	value
	site-priority	1		houston.world
	site-priority	2		new_york.world
	order-status	1		ordered
	order-status	2		shipped
	order-status	3		billed
	...	...		...

現行の位置で定義されたすべての優先グループの値は、DBA_REPPRIORITY ビューに表示されます。図 5–1 の例では、site-priority と order-status という 2 つの異なる優先グループがあります。CUSTOMER 表は、site-priority 優先順位グループを使用しています。この例の order-status 優先グループでは、"billed"（優先順位値 =3）は "shipped" （優先順位値 =2）よりも優先順位が高く、"shipped" は "ordered"（優先順位値 =1）よりも優先順位が高くなっています。

Replication Manager で優先グループによる更新競合の解消方法を選択するには、表内に "priority" 列を指定する必要があります。

ターゲット環境　優先グループによる競合解消方法は、ワークフロー環境用に設定されたレプリケート環境で役に立ちます。たとえば、order-status が "shipping" 状態になると、"order entry" 部門による更新は常に上書きされます。

サポート・メカニズム　ターゲット列に含まれる値の優先順位を定義する必要があります。この優先順位を定義するのは、指定された列値の優先順位に基づいて競合を解消するためです。優先順位の定義は、優先グループに格納されます。

優先グループによる方法の実現　Replication Manager を使用して優先グループによる競合解消方法を定義する方法は、Replication Manager のオンライン・ヘルプを参照してください。レプリケーション・マネージメント API を使用した場合の定義方法は、『Oracle8*i* レプリケーション・マネージメント API リファレンス』の第 6 章の「優先グループ」の項を参照してください。

サイト優先順位

サイト優先順位は、特殊な種類の優先グループです。サイト優先順位を使用する場合、指定する "priority" 列は、更新が発行された サイトのグローバル・データベース名に自動的に更新されます。各データベース・サイトに割り当てられた優先順位レベルは、DBA_REPPRIORITY ビューに表示されます。あるサイトが最も正確な情報を持っている可能性が高いとみなされる場合、サイト優先順位が役に立つことがあります。たとえば、図 5–1 では、ニューヨーク・サイト（優先順位値 =2）は本社で、ヒューストン・サイト（優先順位値 =1）は営業所にある更新可能スナップショットです。したがって、本社オフィスは、各顧客のクレジット限度額に関する最も正確な情報を持っている可能性が営業所よりも高いと考えられます。

注意： DBA_REPPRIORITY ビューの優先グループ列には、site-priority 優先順位グループおよび order-status 優先グループの両方が表示されます。

サイト優先順位を使用している場合、複数のマスターの収束は保証されません。ただし、優先グループを使用していて、"priority" 列の値が常に増加している場合は、複数のマスターの収束が保証されます。つまり、優先順位列の値は、イベントの順序、たとえば ordered、shipped、billed に対応します。

優先グループの場合と同様、Replication Manager を使用して列グループに対してサイト優先順位による競合解消方法を選択する前に、準備することがあります。

ターゲット環境　優先グループと同様に、サイト優先順位による競合解消方法は、ワークフロー環境でインプリメントされるのが一般的です。また、サイト優先順位による競合解消方法を大量配布環境（1 つのマスター・サイトおよび任意の数のスナップショット・サイトから構成される環境）で使用すると、データ収束を保証できます。

サイト優先順位による競合解消方法は、主要な競合解消方法が失敗した場合の予備の方法としても適しています。

サポート・メカニズム　行が更新されたときにサイト情報を格納するための列を指定する必要があります。また、行が更新または挿入されたときにこのサイト列に更新中のサイトのグローバル名を移入するトリガーも作成する必要があります。このトリガーの例は、Replication Manager のオンライン・ヘルプ、および『Oracle8*i* レプリケーション・マネージメント API リファレンス』の第 6 章の「サイトの優先順位」の項に示してあります。

レプリケーション環境に関係する各サイトの優先順位も定義する必要があります。この優先順位を定義するのは、更新または挿入を実行したサイトの優先順位に基づいて競合を解消するためです。サイト優先順位の定義は、優先グループに格納されます。

サイト優先順位による方法の実現　Replication Manager を使用してサイト優先順位による競合解消方法を定義する方法は、Replication Manager のオンライン・ヘルプを参照してください。レプリケーション・マネージメント API を使用した場合の定義方法は、『Oracle8*i* レプリケーション・マネージメント API リファレンス』の第 6 章の「サイトの優先順位」の項を参照してください。

一意性競合の解消方法

Oracle には、一意性競合を解消する事前作成の方法が 3 つ用意されています。

- 発行元サイトのグローバル・サイト名を、発行元サイトの列値に追加する方法
- 生成された順序番号を、発行元サイトの列値に追加する方法
- 発行元サイトの行値を廃棄する方法

次の項では、一意性競合の各解消方法について、詳しく説明します。

注意： Oracle の事前作成の一意性競合の解消方法では、実際にはレプリケート環境のデータは収束しません。これらの方法は、制約違反を解決するテクニックのみを提供します。Oracle の一意性競合の解消方法を使用する場合には、一意性競合の発生を知らせる通知メカニズムを同時に使用して、必要に応じて手動でレプリケート・データを収束する必要があります。

サイト名の追加

サイト名の追加による競合解消方法では、トランザクションの発生元サイトのグローバル・データベース名が、DUP_VAL_ON_INDEX 例外を生成しているレプリケートされた列値に追加されます。この方法を使用すると一意の整合性制約に違反せずに列を挿入または更新できますが、複数のマスター・サイト間での収束は提供されません。そのために生じた矛盾は、手動で解決する必要があります。したがって、この方法は、なんらかの通知機能とともに使用するように想定されています。

注意： サイト名の追加および順序の追加は、いずれもキャラクタ列のみに使用できます。

サイト名の追加による解消方法は、データの完全な正確さよりもデータの可用性の方が重要な場合に有用です。データがレプリケートされた後ですぐに使用可能な状態にするには、次のようにします。

- サイト名の追加による方法を選択します。
- 競合をロギングするのではなく、通知スキームを使用して、適切な担当者に重複を解消するように警告します。

サイト名の追加による方法では、一意性競合が発生すると、トランザクションの発行元サイトのグローバル・データベース名が、レプリケートされた列値に追加されます。名前は最初のピリオド（.）までが追加されます。たとえば、HOUSTON.WORLD は HOUSTON になります。

順序の追加

順序の追加による競合解消方法では、生成された順序番号が、DUP_VAL_ON_INDEX 例外を生成している列値に追加されます。この方法を使用すると一意の整合性制約に違反せずに列を挿入または更新できますが、複数のマスター・サイト間での収束は提供されません。そのために生じた矛盾は、手動で解決する必要があります。したがって、この方法は、なんらかの通知機能とともに使用するように想定されています。

注意： サイト名の追加および順序の追加は、いずれもキャラクタ列のみに使用できます。

順序の追加による解消方法は、データの完全な正確さよりもデータの可用性の方が重要な場合に有用です。データがレプリケートされた後ですぐに使用可能な状態にするには、次のようにします。

- 順序の追加による方法を選択します。
- 競合をロギングするのではなく、通知スキームを使用して、適切な担当者に重複を解消するように警告します。

順序の追加による方法では、生成された順序番号が列値に追加されます。列値は必要に応じて切り捨てられます。列値の生成された部分が列の長さを超えた場合、この競合解消方法ではエラーは解決されません。

廃棄

一意性の廃棄による競合解消方法では、エラーを発生させた発行元サイトの列を単に廃棄することによって一意性競合を解消します。この方法では、複数のマスター間での収束は保証されないため、通知機能とともに使用する必要があります。

追加による方法とは異なり、一意性の廃棄による方法では、データの正確さを検証できるまで、データの伝播が最小限にとどめられます。

削除による競合解消方法

Oracle では、削除競合を解消する事前作成の方法は提供していません。5-8 ページの「削除競合の回避」で説明したように、削除競合を回避するようにデータベースおよびフロントエンド・アプリケーションを設計する必要があります。そのためには、削除する行に一定の時間間隔でマーク付けし、マーク付けした行をパージするプロシージャ・レプリケーションを使用してそれらの行を削除するようにします。

削除競合の回避方法は、5-8 ページの「削除競合の回避」を参照してください。レプリケーション環境での削除競合の回避方法は、『Oracle8*i* レプリケーション・マネージメント API リファレンス』の第 6 章「競合解消」を参照してください。

パフォーマンス・メカニズム

行の更新競合を検出および解消するには、伝播サイトから受信サイトに行の新旧の値に関するデータを送信する必要があります。環境によって、更新競合の検出および解消をサポートするために伝播されるデータの量は異なります。

最小限の通信

すべての関連サイトが Oracle8 以降のデータベースを使用している環境でレプリケート表を作成する場合、最小限の通信機能を使用可能にすると、表内の変更された各行に競合が発生しているかどうかを判断するために通信する必要があるデータの量を最小化できます。最小限の通信機能を使用可能にすると、次の値が伝播されます。

- 主キーおよび列グループの各列の古い値（変更前の値）
- 列グループの更新された各列の新しい値

注意： 挿入された行には、古い値はありません。また、削除された行には、新しい値はありません。

一般的に、Oracle8 以降のレプリケーション環境では、データ伝播を最小化して、ネットワーク上で送信されるデータの量を減らすことが可能です。それにより、システム全体のパフォーマンスが向上します。

これに対して、レプリケート環境に Oracle7 のサイトと Oracle8 以降のサイトが混在している場合は、行データの通信を最小化して更新競合を解消することはできません。この場合、競合を解消するには、変更された各行の新旧の値をすべて伝播する必要があります。

古い値の送信および比較

前述の方法でデータ伝播を最小化している場合、DBMS_REPCAT.SEND_OLD_VALUES プロシージャおよび DBMS_REPCAT.COMPARE_OLD_VALUES プロシージャを使用することで、データ伝播をさらに削減できる場合があります。これらのプロシージャを使用すると、競合の検出および解消に必要な場合に限って古い値が送信されます。たとえば、最も新しいタイムスタンプによる競合の検出および解消方法では、キーおよびタイムスタンプが設定されていない列の古い値を必要としません。

提案： LOB データ型をレプリケートしていて、これらの列で競合が発生しないと考えられる場合は、古い値の伝播をさらに最小化することが特に有効です。

注意： 予想される競合を検出および解消できるように、適切な古い値を伝播する必要があります。ユーザー指定の競合解消プロシージャでは、送信される古い列値 NULL を適切に処理できる必要があります。データ伝播をさらに削減するために SEND_OLD_VALUES プロシージャおよび COMPARE_OLD_VALUES プロシージャを使用すると、予期しない競合に対する保護が低下します。

データ伝播をさらに削減するには、次のプロシージャを実行します。

```
DBMS_REPCAT.SEND_OLD_VALUES(
   sname           IN  VARCHAR2,
   oname           IN  VARCHAR2,
   { column_list   IN  VARCHAR2,
   | column_table  IN  DBMS_REPCAT.VARCHAR2s,}
   operation       IN  VARCHAR2 := 'UPDATE',
   send            IN  BOOLEAN  := TRUE );

DBMS_REPCAT.COMPARE_OLD_VALUES(
   sname           IN  VARCHAR2,
   oname           IN  VARCHAR2,
   { column_list   IN  VARCHAR2,
   | column_table  IN  DBMS_REPCAT.VARCHAR2s,}
   operation       IN  VARCHAR2 := 'UPDATE',
   compare         IN  BOOLEAN  := TRUE );
```

これらのプロシージャを実行した後で、変更を有効にするために、DBMS_REPCAT.GENERATE_REPLICATION_SUPPORT プロシージャで MIN_COMMUNICATION を TRUE に設定してレプリケーション・サポートを生成する必要があります。

注意： `operation` パラメータを使用すると、行が削除されたときまたは非キー列が更新されたとき（あるいはその両方）に、非キー列の古い値を送信するかどうかを決定できます。古い値を送信しない場合、古い値のかわりに NULL が送信され、更新または削除が適用されるときに古い値とターゲット・サイドの列のカレント値が等しいとみなされます。

古い列値に指定した動作は、DBA_REPCOLUMN ビューの 2 つの列、COMPARE_OLD_ON_DELETE（'Y' または 'N'）および COMPARE_OLD_ON_UPDATE（'Y' または 'N'）に表示されます。

次の例では、これらのプロシージャを使用してデータ伝播をさらに削減する方法を示します。3 つの列を持つ 'SCOTT.REPORTS' 表を考えてみます。列 1 は主キーであり、独自の列グループ（列グループ 1）を構成します。列 2 および列 3 は、2 番目の列グループ（列グループ 2）に属します。

2 番目の列グループの競合解消方法は、サイト優先順位です。列 2 は、サイト名を含む VARCHAR2 型の列です。列 3 は、LOB 列です。LOB を更新するときは必ず、更新が行われたサイトのグローバル名で列 2 も更新する必要があります。LOB に対する個々の更新にはトリガーがないので、LOB に対して個々の更新を行う場合は常に列 2 を明示的に更新する必要があります。

DBMS_REPCAT.GENERATE_REPLICATION_SUPPORT プロシージャで MIN_COMMUNICATION を TRUE に設定して SCOTT.REPORTS のレプリケーション・サポートを生成し、次に UPDATE 文を使用して列 2（サイト名）と列 3（LOB）を変更するとします。サイト名の新しい値と LOB の新しい値が更新されているので、遅延リモート・プロシージャ・コール（RPC）にはこれらの値が含まれます。遅延 RPC には、主キー（列 1）の古い値、サイト名（列 2）の古い値および LOB（列 3）の古い値も含まれます。

注意： 競合の検出および解消には、LOB の古い値は必要ありません。競合の検出と解消に必要なのは、列 C2（サイト名）のみです。LOB の古い値を送信すると、伝播時間が大幅に増加する可能性があります。

列 C2 または列 C3 が更新されたときに LOB の古い値を伝播しない場合は、次のコールを実行します。

```
DBMS_REPCAT.SEND_OLD_VALUES(
   sname           IN  'SCOTT',
   oname           IN  'REPORTS'
   column_list     IN  'C3',
   operation       IN  'UPDATE',
   send            IN  FALSE );

DBMS_REPCAT.COMPARE_OLD_VALUES(
   sname           IN  'SCOTT',
   oname           IN  'REPORTS'
   column_list     IN  'C3',
   operation       IN  'UPDATE',
   compare         IN  FALSE);
```

変更を有効にするには、DBMS_REPCAT.GENERATE_REPLICATION_SUPPORT プロシージャで MIN_COMMUNICATION を TRUE に設定して SCOTT.REPORTS のレプリケーション・サポートを生成する必要があります。続いて UPDATE 文を使用して列 2（サイト名）と列 3（LOB）を変更するとします。遅延 RPC には、主キー（列 1）の古い値、サイト名（列 2）の古い値と新しい値、および LOB（列 3）の新しい値が含まれます。遅延 RPC は、主キーの新しい値と LOB の古い値については NULL を含みます。

注意： Oracle の競合解消では、LOB の個々の更新はサポートしていません。

関連項目： DBMS_REPCAT.SEND_OLD_VALUES プロシージャおよび DBMS_REPCAT.COMPARE_OLD_VALUES プロシージャの詳細は、『Oracle8*i* レプリケーション・マネージメント API リファレンス』を参照してください。

6

高度な設定

この章では、Oracle のレプリケート・データベース環境をインプリメントするときに使用できるいくつかの高度なテクニックについて説明します。

- プロシージャ・レプリケーションの使用方法
- 耐障害性を実現するための設計
- 動的所有権の競合回避の使用方法
- 変更をレプリケートしない表の変更
- レプリケーションの保護メカニズムについて

プロシージャ・レプリケーションの使用方法

プロシージャ・レプリケーションを使用すると、レプリケート環境でシリアルに実行される大量の行を処理する、大規模なバッチ指向操作のパフォーマンスが向上します。

典型的なアプリケーション適用例は、古いデータや論理的にオンライン・データベースから削除されたデータを削除するために営業時間外に時々（たとえば 15 分ごとなど）実行されるパージ操作（アーカイブ操作とも呼ばれます）です。プロシージャ・レプリケーションを使用して、削除した行をパージする例は、5-8 ページの「削除競合の回避」を参照してください。

プロシージャ・レプリケーションの制限事項

レプリケート・プロシージャのパラメータはすべて、IN パラメータである必要があります。OUT モードおよび IN/OUT モードはサポートされていません。これらのパラメータにサポートされているデータ型は、NUMBER、DATE、VARCHAR2、CHAR、ROWID、RAW、BLOB、CLOB、NCHAR、NVARCHAR および NCLOB です。

Oracle では、レプリケート・プロシージャによって発生した更新競合を検出できません。競合の検出および解消は、レプリケート・プロシージャで行う必要があります。独自の競合解消ルーチンを作成することは困難なので、単に競合の発生を回避することが最善です。

次のガイドラインを遵守すると、プロシージャ・レプリケーションの使用時にすべてのサイトで表の整合性を保持できます。

- 遅延プロシージャの本体内で、行レベルのレプリケーションを使用禁止にします。6-17 ページの「変更をレプリケートしない表の変更」を参照してください。
- 一度に 1 つのレプリケート・プロシージャを実行します。詳細は、6-3 ページの「トランザクションのシリアライズ化」を参照してください。
- 遅延トランザクションをシリアル伝播します。詳細は、7-10 ページの「スケジュール・リンクのガイドライン」を参照してください。
- レプリケート・プロシージャをパッケージ化します。このパッケージにファンクションを含めることはできません。スタンドアロンの遅延プロシージャとスタンドアロンまたはパッケージ化した遅延ファンクションは現在サポートされていません。
- 遅延プロシージャは、ローカルで所有するデータのみを参照します。
- プロシージャでは、ローカルで生成されたフィールド、値または環境に依存する SQL ファンクションを使用しません。たとえば、プロシージャで SYSDATE をコールしません。
- データ所有権を静的にパーティション化します。つまり、行の所有権がサイト間で変更されません。
- マスター・サイトに複数のマスター・グループがあり、1 つ以上のマスター・グループが停止状態にある場合、マスター・サイトのマスター・グループではプロシージャ・レプリケーションを実行できません。これは、1 つのマスター・グループでプロシージャ

を実行すると、別のマスター・グループのオブジェクトが更新される可能性があるためです。マスター・サイトのすべてのマスター・グループがデータを普通にレプリケートしている場合（つまり、停止状態のマスター・グループがない場合）にのみ、プロシージャ・レプリケーションを実行できます。

たとえば、マスター・サイト DB1 で、マスター・グループ A に SAL_RAISE という名前のプロシージャがあり、そのマスター・サイト DB1 のマスター・グループ B が停止状態にある場合は、マスター・グループ A が普通にデータをレプリケートしているときでも SAL_RAISE プロシージャを実行できません。

トランザクションのシリアライズ化

シリアル処理によって、データの整合性が保証されます。レプリケーション機能は、レプリケート・トランザクションを一度に 1 つずつ伝播し、実行します。たとえば、ローカル・データの更新を実行する 2 つのプロシージャ A および B があると想定します。次のアクションを順序どおり実行するとします。

1. A および B をローカルで実行します。
2. 他のノードにある A および B のレプリカを実行する要求をキューに入れます。
3. コミットします。

他のノードにある A および B のレプリカは、発行元サイトでコミットされた順序と同じ順序で、完全にシリアルに処理されます。ただし、A および B が発行元サイトで同時に実行されると、生じる結果がローカルとリモートで異なる場合があります。発行元サイトで A および B をシリアルに処理した場合、すべてのサイトの結果は必ず同じになります。トランザクションをシリアル伝播すると、A と B はターゲット・サイトでどのような場合にも必ずシリアルに処理されます。

また、確実にシリアライズ化できるように考慮してプロシージャを作成することもできます。たとえば、パラレル伝播を使用している場合、SELECT... FOR UPDATE を使用して問合せをすると、発行元サイトとターゲット・サイトでシリアライズ化を実現できます。

レプリケート・プロシージャのサポートの生成

行レベルのレプリケーション・サポートは、プロシージャの最初で使用禁止にし、終わりに再び使用可能にする必要があります。これにより、プロシージャを実行した結果として生じた更新が他のサイトに伝播しないようにできます。行レベルのレプリケーションは、それぞれ次のプロシージャをコールすることによって、使用可能および使用禁止になります。

- DBMS_REPUTIL.REPLICATION_ON
- DBMS_REPUTIL.REPLICATION_OFF

関連項目： 6-18 ページの「レプリケーション機能を使用禁止にする」

レプリケートしたパッケージにレプリケーション・サポートを生成すると、レプリケーション・プロパゲータのスキーマにラッパー・パッケージが作成されます。

注意： 現行のプロパゲータを登録解除すると、プロパゲータのスキーマに生成された既存のラッパーがすべて削除されます。新しいプロパゲータを登録した後で、ラップ・ストアド・プロシージャのレプリケーション・サポートを再生成する必要があります。

ラッパー・パッケージの名前は元の名前と同じですが、接頭辞にはプロシージャのレプリケーション・サポートを生成したときに指定した文字列が使用されます。接頭辞を指定しない場合、デフォルトの接頭辞 "defer_" が使用されます。ラッパー・プロシージャには元のプロシージャと同じパラメータがあり、さらに 2 つのパラメータ CALL_LOCAL および CALL_REMOTE が追加されます。これらの 2 つの CHAR 型パラメータによって、プロシージャが実行される場所が決まります。CALL_LOCAL が 'Y' の場合は、プロシージャはローカルで実行されます。CALL_REMOTE が 'Y' の場合は、プロシージャは最終的にレプリケート環境内の他のすべてのマスター・サイトで実行されます。

変更を同期伝播している場合は、リモート・プロシージャが直接コールされます。変更を非同期で伝播している場合は、そのプロシージャのコールが遅延トランザクション・キューに追加されます。デフォルトでは、CALL_LOCAL は 'N' に、CALL_REMOTE は 'Y' に設定されています。

パッケージのレプリケーション・サポートは、2 つのフェーズで生成されます。最初のフェーズで、すべてのサイトでパッケージ仕様部が生成されます。次のフェーズで、すべてのサイトでパッケージ本体が生成されます。これら 2 つのフェーズは、同期レプリケーションをサポートするために必要です。

たとえば、1 つの引数 ENAME を使用するプロシージャ NEW_DEPT を含むパッケージ EMP_MGMT を生成すると想定します。このパッケージをシステムのすべてのマスター・サイトにレプリケートするには、Replication Manager を使用してこのパッケージをマスター・グループに追加してから、オブジェクトのレプリケーション・サポートを生成します。Replication Manager を使用したマスター・グループおよびレプリケート・オブジェクトの管理の詳細は、Replication Manager のオンライン・ヘルプを参照してください。前述のス

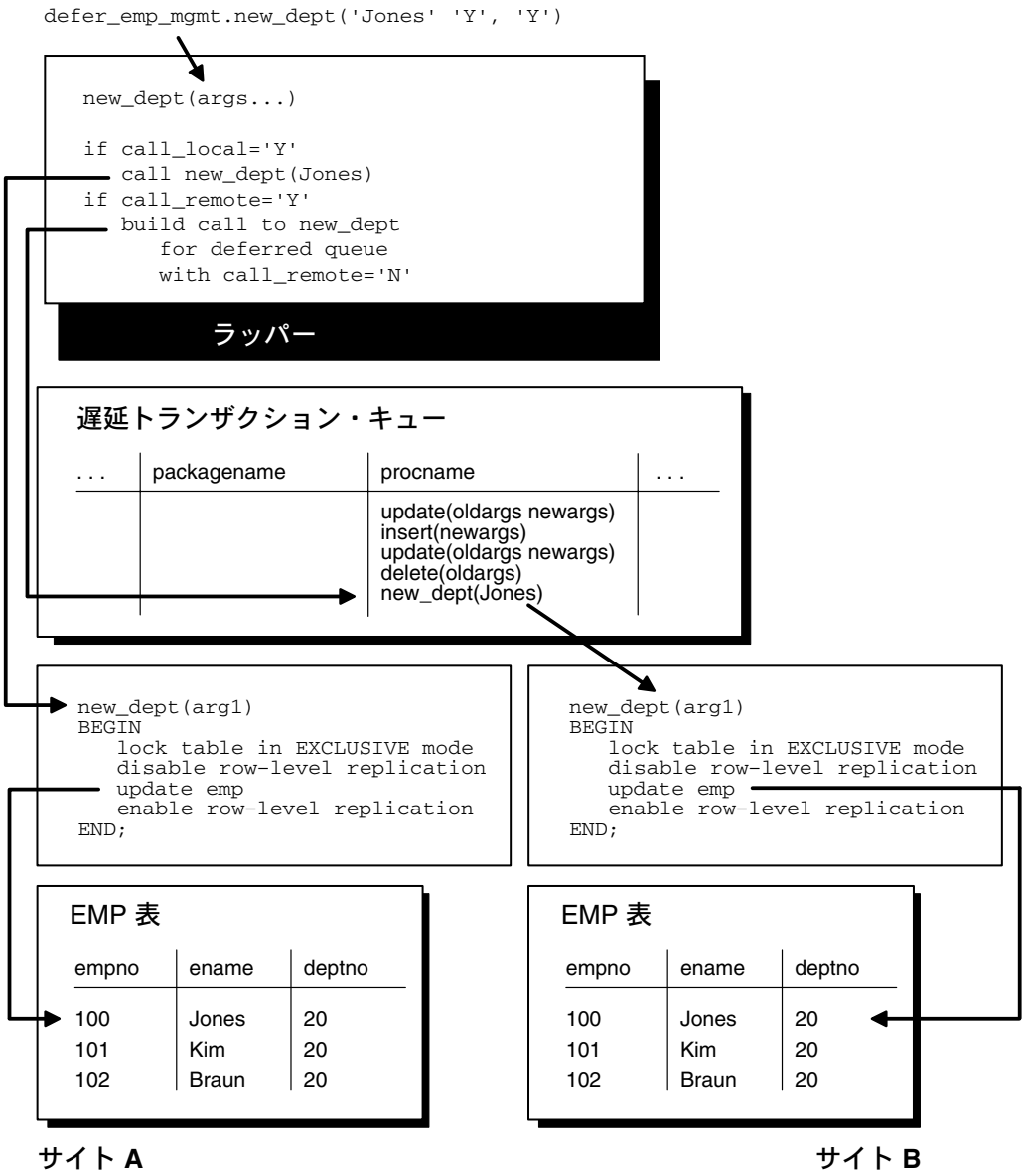
テップが完了した後で、レプリケートしたパッケージのプロシージャを次のようにアプリケーションでコールできます。

```
defer_emp_mgmt.new_dept( ename         => 'Jones',
                         call_local    => 'Y',
                         call_remote   => 'Y');
```

図 6-1 に示すように、ラッパー・プロシージャの論理では、まずローカル・サイトで、続いてすべてのリモート・サイトでプロシージャが確実にコールされます。また、ラッパー・プロシージャの論理では、レプリケート・プロシージャがリモート・サイトでコールされるときは必ず CALL_REMOTE が 'N' となり、そのプロシージャがそれ以上伝播されることはありません。

データ所有権の静的パーティション化で複合レプリケート環境を稼働する場合（つまり、行レベル・レプリケーションを防止していない場合）は、行レベル・レプリケーションとプロシージャ・レプリケーションがともに同じ非同期キューを使用するので、レプリケーション機能はリモート・ノードでの操作の順序を保持します。

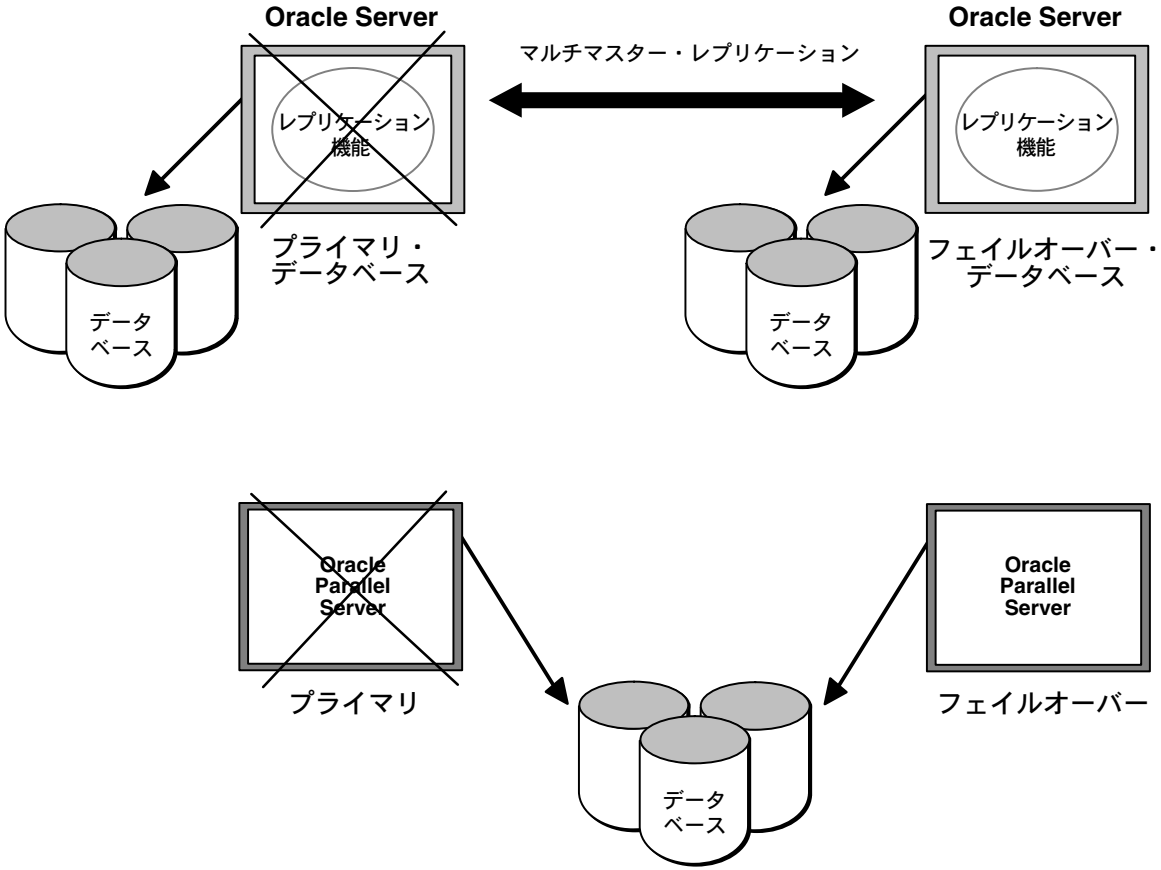
図 6-1 非同期プロシージャ・レプリケーション



耐障害性を実現するための設計

耐障害性の機能により、システムまたはサイトの障害時にもアプリケーションを継続して実行できます。また、アプリケーションをフェイルオーバー・システム上で実行でき、障害発生時にプライマリ・システム上でアクセスしていたのと同じか、またはほとんど同じデータにアクセスできます。図 6-2 に示すように、Oracle Server には、Oracle Parallel Server およびレプリケーション機能という、耐障害性を実現するための 2 つのテクノロジがあります。

図 6-2 耐障害性を実現するための方法： レプリケーションと Parallel Server



Oracle Parallel Server とレプリケーション

Oracle Parallel Server は、Oracle Server のインスタンスをサポートするシステムの障害時にシステムの耐障害性を確保するフェイルオーバーをサポートしています。Oracle Parallel Server には、クラスタまたは超並列ハードウェア・プラットフォームが必要です。それにより、クラスタまたは超並列システムが稼動しているローカル環境での処理システムの障害に対する保護が可能となります。

このような環境では、Oracle Parallel Server は耐障害性を実現するための理想的なソリューションです。大量のトランザクションをサポートしており、インスタンスに障害が発生した場合でもトランザクションの消失やデータの不整合が生じることがありません。インスタンスに障害が発生した場合、Oracle Parallel Server の正常なインスタンスによって未完了のトランザクションが自動的にリカバリされます。障害が生じたシステムで実行されていたアプリケーションはフェイルオーバー・システムで実行でき、データベースのすべてのデータにアクセスできます。

ただし、Oracle Parallel Server には、サイト全体におよぶ障害（洪水、火事、破壊行為など）によってクラスタまたは超並列システム全体が操作不能になるような状況に対しては耐障害性がありません。サイト障害に対して耐障害性を確保するために、レプリケーション機能を使用して、地理的に離れた場所でデータベースのレプリカをメンテナンスできます。

ローカル・システムに障害が発生しても、アプリケーションはリモート・サイトで実行を継続できます。ただし、レプリケーションによって、すべてのトランザクションの保護が保証されるわけではありません。また、プライマリ・サイトをリカバリするときは、データの整合性が損なわれないように細心の注意が必要です。

注意： スタンバイ・データベースを構成して、Oracle データベースをサイト障害から守ることもできます。Oracle のスタンバイ・データベース機能の詳細は、『Oracle8*i* バックアップおよびリカバリ・ガイド』を参照してください。

耐障害性を実現するための設計

耐障害性を実現するためにレプリケーション機能を使用する場合は、次の事項を考慮してください。

- レプリケーション機能によって、プライマリ・システムと同じ量のトランザクションを処理できる必要があります。これはアプリケーションによって異なりますが、一般的に、Oracle Parallel Server を使用する場合よりもスループットはかなり低くなります。
- プライマリ・サイトで障害が生じた場合、プライマリ・サイトで少し前にコミットされたトランザクションは、フェイルオーバー・サイトにまだ非同期に伝播されていない可能性があります。このトランザクションは消失したように見えます。
- この消失したトランザクションは、プライマリ・サイトがリカバリされたときに処理する必要があります。

たとえば、リモートのフェイルオーバー・サイトのメンテナンスにレプリケーションを使用する受注入力システムにおいて、システムの実行時にプライマリ・システムで障害が発生したとします。

障害発生時、少し前にプライマリ・サイトで実行された 2 つのトランザクションの変更は、フェイルオーバー・サイトに伝播および適用させていません。最初のトランザクションは新規の受注の入力で、2 番目のトランザクションは既存の受注の取消しです。

新規受注入力の場合、フェイルオーバー・システムでの処理の継続中にその受注が存在しないことが発見されると、再入力されると考えられます。既存の受注の取消しの場合は、受注の取消しは発見されずに、受注処理が継続されると考えられます。つまり、取り消された商品が出荷されて、顧客に請求が送られます。

プライマリ・サイトがリストアされた場合を考えます。フェイルオーバー・システムで実行されたすべての変更をプライマリ・システムに単に戻すと、競合が発生します。

具体的には、プライマリ・システムに障害が発生する直前に受注された商品が重複することになります。さらに、プライマリ・システムでは取り消されているはずの受注の出荷および請求トランザクションにより、データが変更されます。

こうした状況に対処するには、システムを綿密に設計する必要があります。次の項では、このプロセスについて説明します。

耐障害性の高いシステムの実現

Oracle のレプリケーション機能を使用すると、複数のレプリケート・マスター・サイトにより、サイト障害に対する耐障害性を実現できます。次のいずれかの方法でシステムを構成する必要があります。実現の難易度が低いものから順に列記しています。

- フェイルオーバー・サイトは読取り専用で使用します。つまり、プライマリ・サイトで障害が発生した場合でも、フェイルオーバー・サイトでは更新できません。
- 障害の後、プライマリ・サイトは、エクスポートおよびインポート、または完全バックアップによって、フェイルオーバー・サイトからリストアされます。
- すべてのデータおよびトランザクションでは、競合を完全に解消します。このためには、綿密な設計および実現が必要です。重複トランザクションなど、プライマリ・サイトがリストアしたときに発生する競合を、適切かつ確実に解消する必要があります。
- アプリケーション・レベルのルーチンやプロシージャを独自に作成して、プライマリ・サイトがリストアされたとき、およびアクティブなフェイルオーバー・システムでキューに入れられていたトランザクションがプライマリ・サイトに伝播および適用されたときに発生するデータの不整合を処理します。

Net8 を使用すると自動的に接続時フェイルオーバーを構成できます。つまり、Net8 を使用すれば、最初のマスター・サイトで障害が発生した場合に別のマスター・サイトに処理を切替えが可能です。自動的に接続時フェイルオーバーを設定するには、`tnsnames.ora` ファイルで、FAILOVER オプションを ON に設定し、複数の接続記述子を指定します。

関連項目： 接続時フェイルオーバーの設定方法の詳細は、『Oracle8*i* Net8 管理者ガイド』を参照してください。

レプリケーション・システムにおけるデータベースのバックアップおよびリカバリ

レプリケーションを使用するデータベースは分散データベースです。レプリケーション・データベースのバックアップを作成するときは、『Oracle8*i* 管理者ガイド』で概説している分散データベースのバックアップのガイドラインに従ってください。レプリケーション・データベースをリカバリするときは、『Oracle8*i* 管理者ガイド 』で説明している協調分散リカバリのガイドラインに従ってください。

協調分散リカバリのガイドラインに従わない場合、レプリケーション・データベースの整合性は保証されません。たとえば、リストアされたマスター・サイトが別のマスター・サイトに異なるトランザクションを伝播していることがあります。場合によっては、不正確なリカバリ操作を修正するために、追加のステップを実行する必要があります。その方法の 1 つは、リカバリしたデータベース内のすべてのレプリケート・オブジェクトを削除し、再作成する方法です。

推奨事項： レプリケート・オブジェクトの削除および再作成を実行する前に、リストアしたデータベースから保留中の遅延トランザクションおよび遅延エラー・レコードを削除し、未解決の分散トランザクションを解決してください。リストアされたデータベースがいくつかのレプリケート環境のマスター定義サイトであった場合、オブジェクトを削除および作成する前に新しいマスター定義サイトを設定する必要があります。リストアされたデータベースがマスターであるスナップショット、およびリストアされたデータベース内のスナップショットは、完全にリフレッシュする必要があります。

データに対する継続的なアクセスを可能にするため、マスター定義サイトの変更（リストアされるデータベースがマスター定義サイトであった場合）、またはスナップショット・サイトのマスターの変更（それらのマスター・サイトがリストアされている場合）が必要な場合があります。

インポートされたデータに対するチェックの実行 レプリケート・オブジェクト、またはレプリケーション機能で使用するオブジェクト（たとえば、DBA_REPSITES ビュー）のエクスポートおよびインポートを実行した後で、DBMS_REPCAT パッケージの REPCAT_IMPORT_CHECK プロシージャを実行します。

次の例では、プロシージャは、スナップショット・サイトの ACCT レプリケート・オブジェクト・グループのオブジェクトをチェックして、オブジェクト識別子と状態値が適切であることを確認します。

```
DBMS_REPCAT.REPCAT_IMPORT_CHECK( gname     =>   'acct',
                                 master    =>   FALSE);
```

関連項目： 詳細は、『Oracle8*i* レプリケーション・マネージメント API リファレンス』の「REPCAT_IMPORT_CHECK プロシージャ」の説明を参照してください。

動的所有権の競合回避の使用方法

この項では、競合を回避するアプリケーションを設計するさらに高度な方法を説明します。トークン渡しという方法は、次に説明するワークフローによる方法と類似しています。この項ではトークン渡しを使用して行全体の所有権を制御する方法について説明しますが、この方法に変更を加えた形式を使用して、行内の個々の列グループの所有権を制御することもできます。

ワークフローでもトークン渡しでも、データの動的所有権を使用できます。動的所有権を使用すると、行を更新できるのは一度に 1 つのサイトのみになりますが、行の所有権をサイトからサイトに渡すことができます。ワークフローおよびトークン渡しはいずれも、1 つ以上の識別子列の値を使用して、現在誰が行を更新できるかを判断します。

ワークフロー

ワークフロー・パーティション分割を使用した場合、データの所有権がサイトからサイトへとプッシュされると考えることができます。識別子列の値を変更することによって行の所有権を別のサイトにプッシュできるのは、その行の現行の所有者のみです。

受注、出荷および請求用の個別のサイトの簡単な例を取り上げます。識別子列は受注の状態を示すために使用されます。その状態によって、どのサイトで行を更新できるかが決まります。受注サイトのユーザーは、受注を入力した後でこの行の状態を SHIP に更新します。これ以後は受注サイトのユーザーはこの行を変更できません。所有権は出荷サイトにプッシュされています。

受注品を出荷した後で、出荷サイトのユーザーはこの行の状態を BILL に更新するので、所有権は請求サイトにプッシュされます。これ以降、同様にプッシュされます。

競合を適切に回避するには、動的データ所有権をインプリメントするアプリケーションが次の条件に必ず適合する必要があります。

- 行を更新できるのは行の所有者のみであること
- 行が複数のサイトによって所有されないこと
- 順序競合がすべてのサイトで正常に解消できること

ワークフローをパーティション分割すると、識別子列を更新し、その行の所有権を次のサイトにプッシュできるのは、行の現行の所有者のみになります。あるサイトが所有権を手放すまで、他のサイトは所有権を取得できません。したがって、複数の所有者が存在しないことが確実になります。

作業のフローが順序付けられているので、作業のフローで最後に発生する変更を適用することによって順序競合を解消できます。順序競合はすべて、ワークフロー・プロセスの各ステップごとに優先順位値が増加する、優先順位による競合解消方法で解消できます。

優先順位による競合解消方法では、優先値が常に増加している限り、複数のマスターが正常に収束します。

トークン渡し

トークン渡しは、動的所有権の基準を満たす、より一般的な方法です。トークン渡しをインプリメントするには、識別子列のかわりに、レプリケート表に所有者列およびエポック列が必要です。所有者列には、現在その行を所有していると思われるサイトのグローバル・データベース名が格納されます。

トークン渡しのメカニズムを設計すると、それを使用して、ワークフローを含むデータ所有権の様々な形式の動的パーティション分割をインプリメントできます。

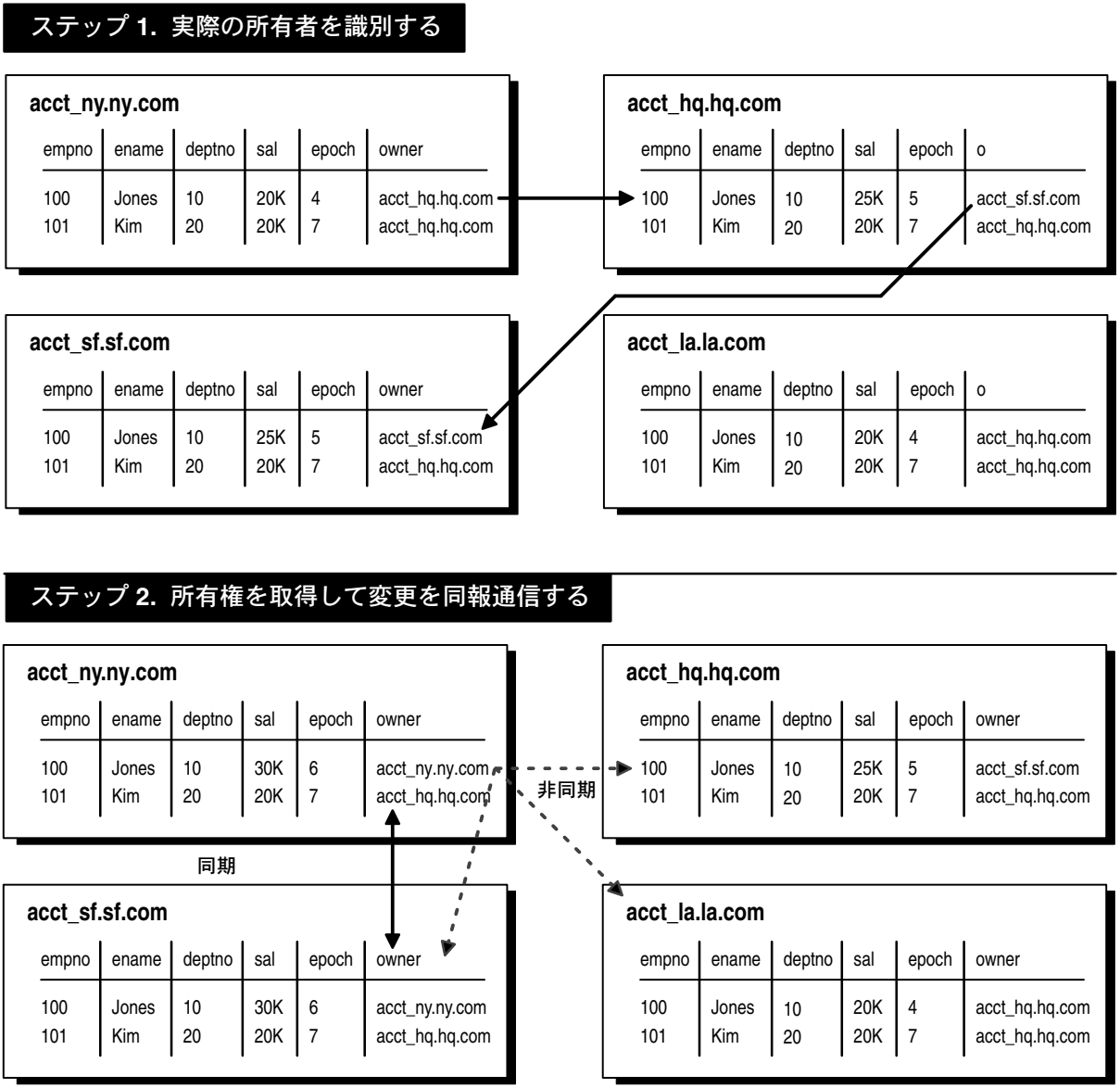
アプリケーションは、トークン渡しを自動的にインプリメントするように設計する必要があります。このアプリケーションの管理外で、所有者列またはエポック列が更新されないようにしてください。

行を更新するときは常に、アプリケーションで次の処理を実行する必要があります。

1. 行の現行の所有者を検索します。
2. 所有権の変更中に行が更新されるのを回避するため、行をロックします。
3. 行の所有権を取得します。
4. 更新を実行します。Oracle では、トランザクションがコミットされるとロックが解除されます。

たとえば、図 6-3 は、従業員番号 100 の所有権が ACCT_SF データベースから ACCT_NY データベースへ渡される仕組みを示します。

図 6-3 トークンの取得


ステップ 1. 実際の所有者を識別する
acct_ny.ny.com
empno ename deptno sal epoch owner
100 Jones 10 20K 4 acct_hq.hq.com
101 Kim 20 20K 7 acct_hq.hq.com
acct_hq.hq.com
empno ename deptno sal epoch o
100 Jones 10 25K 5 acct_sf.sf.com
101 Kim 20 20K 7 acct_hq.hq.com
acct_sf.sf.com
empno ename deptno sal epoch owner
100 Jones 10 25K 5 acct_sf.sf.com
101 Kim 20 20K 7 acct_hq.hq.com
acct_la.la.com
empno ename deptno sal epoch o
100 Jones 10 20K 4 acct_hq.hq.com
101 Kim 20 20K 7 acct_hq.hq.com
ステップ 2. 所有権を取得して変更を同報通信する
acct_ny.ny.com
empno ename deptno sal epoch owner
100 Jones 10 30K 6 acct_ny.ny.com
101 Kim 20 20K 7 acct_hq.hq.com
非同期
acct_hq.hq.com
empno ename deptno sal epoch owner
100 Jones 10 25K 5 acct_sf.sf.com
101 Kim 20 20K 7 acct_hq.hq.com
同期
acct_sf.sf.com
empno ename deptno sal epoch owner
100 Jones 10 30K 6 acct_ny.ny.com
101 Kim 20 20K 7 acct_hq.hq.com
acct_la.la.com
empno ename deptno sal epoch owner
100 Jones 10 20K 4 acct_hq.hq.com
101 Kim 20 20K 7 acct_hq.hq.com

行の所有権の検索

ACCT_NY データベースは、所有権を取得するために簡単な再帰アルゴリズムを使用して行の所有者を検索します。このアルゴリズムの疑似コードは次のとおりです。

```
-- Pseudo code for locating the token owner.
-- This is for a table TABLE_NAME with primary key PK.
-- Initial call should initialize loc_epoch to 0 and loc_owner
-- to the local global name.
get_owner(PK IN primary_key_type, loc_epoch IN OUT NUMBER,
          loc_owner IN OUT VARCHAR2)
{
  -- use dynamic SQL (dbms_sql) to perform a select similar to
  -- the following:
  SELECT owner, epoch into rmt_owner, rmt_epoch
     FROM TABLE_NAME@loc_owner
     WHERE primary_key = PK FOR UPDATE;
  IF rmt_owner = loc_owner AND rmt_epoch >= loc_epoch THEN
   loc_owner := rmt_owner;
   loc_epoch := rmt_epoch;
   RETURN;
  ELSIF rmt_epoch >= loc_epoch THEN
   get_owner(PK, rmt_epoch, rmt_owner);
   loc_owner := rmt_owner;
   loc_epoch := rmt_epoch;
   RETURN;
  ELSE
   raise_application_error(-20000, 'No owner for row');
  END IF;
}
```

所有権の取得

行の所有者が見つかると、ACCT_NY サイトは次のステップを実行することにより ACCT_SF サイトから所有権を取得します。

1. 所有権の交換中に変更が発生するのを回避するため、SF サイトの行をロックします。
2. SF サイトおよび NY サイトの両方で、所有者情報を同期更新します。これにより、常に 1 つのサイトのみが所有者となることが保証されます。SF サイトでの更新は、DBMS_REPUTIL.REPLICATION_OFF を使用してレプリケートできません。NY サイトでの所有権の変更はレプリケートされ（ステップ 4）、最終的に、SF サイトを含め（影響はありませんが）、レプリケート環境のすべてのサイトに伝播されます。
3. 現行の所有者サイト SF の情報を使用して、新しい所有者サイト NY の行の情報を更新します。更新されたデータは、必ず最新のものになります。このとき、NY サイトでの変更はレプリケートされる必要はありません。SF サイトでキューに入れられた、このデータに対する変更はすべて、通常の方法ですべてのサイトに伝播されます。次のス

テップで説明しているように、SF の変更が NY に伝播されると、エポック番号の値のためにこの変更は無視されます。

4. 新しい所有者サイトのエポック番号を、以前のサイトでの値より 1 大きくなるように更新します。この更新を、新しい所有者に対してのみ実行してから、他のマスター・サイトに非同期伝播します。新しい所有者サイトのエポック番号を増やすことで、順序競合が回避されます。

   （前述のステップ 2 で遅延キュー内にあった）SF の変更が最終的に NY サイトに伝播されたとき、そのエポック番号は NY サイトの同じデータのエポック番号よりも小さいので、NY サイトは変更を無視します。

   別の例として、HQ サイトが NY の変更を受信した後に SF の変更を受信したと想定します。この場合、NY サイトから適用された変更にはより大きなエポック番号があるため、HQ サイトは SF の変更を無視します。

変更の適用

更新の実行時、常に自動的にトークン渡しがインプリメントされるようにアプリケーションを設計する必要があります。このアプリケーションの管理外で、所有者列またはエポック列を更新できません。所有権を変更するときに取得したロックは、実際の更新を適用すると解除されます。変更された情報、および更新された所有者情報とエポック情報は、通常の方法で各サイトに非同期伝播されます。

変更をレプリケートしない表の変更

レプリケート・オブジェクトを変更しても、その変更をレプリケート環境の他のサイトにレプリケートしない場合もあります。たとえば、次の状況ではレプリケーションを使用禁止にします。

- 変更の伝播にプロシージャ・レプリケーションを使用している場合。この場合は、プロシージャの先頭で行レベルのレプリケーションを常に使用禁止にする必要があります。
- トリガー・アクションが複数回レプリケートされるのを回避するため、レプリケート表に定義されているトリガーのレプリケーションを使用禁止にする必要がある場合。「トリガーおよびレプリケーション」を参照してください。
- 手動で競合を解消する場合で、この変更を表の他のコピーにレプリケートする必要がない場合。

  たとえば、あるサイトのレコードの状態を訂正し、エラー・トランザクションの再実行時に競合するレプリケート更新を成功させる必要がある場合、この処理が必要となります。あるいは、トランザクションが接続先サイトで適用されなかったため、レプリケートされていない変更を使用して送信元サイトの変更を戻すことが必要な場合もあります。この例では、Replication Manager を使用して、競合するトランザクションを接続先サイトから削除できます。

変更をレプリケートせずに表を変更するには、DBMS_REPUTIL パッケージの REPLICATION_ON プロシージャおよび REPLICATION_OFF プロシージャを使用します。これらのプローシージャは引数を必要とせず、生成されたレプリケーション・トリガー中でフラグとして使用されます。

注意： レプリケーションを使用可能および使用禁止にするには、DBMS_REPUTIL パッケージに対する EXECUTE 権限が必要です。

レプリケーション機能を使用禁止にする

DBMS_REPUTIL.REPLICATION_OFF プロシージャは、カレント・セッションの内部レプリケーション変数の状態を FALSE に設定します。すべてのレプリケート・トリガーはトランザクションをキューに入れる前にこの変数の状態をチェックするため、行レベルのレプリケーションを使用するレプリケート表が変更されても、遅延トランザクションがキューに入ることはありません。

注意： レプリケーションのオンまたはオフは、カレント・セッションのみに影響します。つまり、同じサーバーに接続している他のユーザーは、コミットされた変更を遅延トランザクション・キューに入れる場合の制限は受けません。

プロシージャ・レプリケーションを使用している場合、次の例に示すように、プロシージャの最初に REPLICATION_OFF をコールする必要があります。これにより、レプリケーション機能では、変更の伝播に行レベル・レプリケーションを使用しません。

```
CREATE OR REPLACE PACKAGE update AS
  PROCEDURE update_emp(adjustment IN NUMBER);
END;
/
CREATE OR REPLACE PACKAGE BODY update AS
  PROCEDURE update_emp(adjustment IN NUMBER) IS
  BEGIN
   -- turn off row-level replication for set update
   dbms_reputil.replication_off;
   UPDATE emp . . .;
   -- re-enable replication
   dbms_reputil.replication_on;
  EXCEPTION WHEN OTHERS THEN
   . . .
   dbms_reputil.replication_on;
  END;
END;
```

レプリケーション機能を再び使用可能にする

競合を解消した後、またはレプリケート・プロシージャの終わりで、DBMS_REPUTIL.REPLICATION_ON をコールして、レプリケート表またはスナップショットに対する変更の通常のレプリケーションを再開します。このプロシージャには引数はありません。REPLICATION_ON をコールすると、内部レプリケーション変数が TRUE に設定されます。

トリガーおよびレプリケーション

レプリケート・トリガーをレプリケート表に定義していた場合、トリガーがそれぞれの変更に対して 1 回のみ起動することを確認する必要があります。一般的に、最初の変更時にのみトリガーが起動され、その変更がリモート・サイトにレプリケートされたときには、リモートのトリガーは起動しません。

DBMS_REPUTIL.FROM_REMOTE パッケージ変数の値をトリガーの最初にチェックする必要があります。トリガーが表を更新するのは、この変数の値が FALSE であるときのみです。

また、トリガーが起動される原因となった行以外の行を修正するときは、トリガーの最初にレプリケーションを使用禁止にし、トリガーの終わりで再び使用可能にできます。この方法を使用すると、元の変更のみがリモート・サイトにレプリケートされます。その後、レプリケート・トリガーが各リモート・サイトで起動します。レプリケート・トリガーによって実行される更新は、他のサイトにはプッシュされません。

このアプローチを使用すると、競合解消は実行されません。したがって、トリガーによる変更がデータの整合性に影響していないことを確認する必要があります。

スナップショットのレプリケーションを使用可能または使用禁止にする

現行サイトでのスナップショットに対するローカル・レプリケーション・トリガーをすべて使用禁止にするには、次の例に示すように、SET_I_AM_A_REFRESH をコールして内部リフレッシュ変数を TRUE に設定します。

```
DBMS_SNAPSHOT.SET_I_AM_A_REFRESH(value => TRUE);
```

トリガーを再び使用可能にするには、次に示すように、内部リフレッシュ変数を FALSE に設定します。

```
DBMS_SNAPSHOT.SET_I_AM_A_REFRESH(value => FALSE);
```

内部リフレッシュ変数の値を判別するには、次に示すように、I_AM_A_REFRESH ファンクションをコールします。

```
ref_stat := DBMS_SNAPSHOT.I_AM_A_REFRESH;
```

レプリケーションの保護メカニズムについて

マルチマスター・レプリケーション環境では、障害の発生時でも、リモート・サイトに伝播されたトランザクションは消失されず、再伝播されないことが保証されています。

- 1 つのローカル・トランザクション内で発行された複数のプロシージャ・コールは、1 つのトランザクションとしてリモートで実行されます。
- 伝播の実行中にネットワークまたはリモート・データベースに障害が発生した場合、トランザクションはリモート・サイトでロールバックされ、リモート・データベースが再度アクセス可能になり、伝播が正常に完了するまで、ローカル・キューに残ります。
- トランザクションがローカル・サイトのキューから削除されるのは、すべての接続先サイトへの伝播および適用が正常に完了してからです。

注意： 伝播が正常に完了しても、必ずしもトランザクションがリモート・サイトで正常に適用されるとは限りません。解消不能な競合や記憶領域不足などのエラーが発生すると、トランザクションでエラーが発生することがあり、そのエラーは、リモート・サイトに記録されます。Replication Manager を使用したエラー・トランザクションの表示および管理の詳細は、Replication Manager のオンライン・ヘルプを参照してください。

障害からの保護は、シリアル伝播とパラレル伝播の両方に対して提供されています。

- シリアル伝播の場合、レプリケーション機能で 2 フェーズ・コミットが使用されます。
- パラレル伝播の場合は、パラレル操作用に最適化された特殊な分散トランザクション・プロトコルが使用されます。リモート・サイトは、正常に伝播されたトランザクションを記録し、この情報を定期的にローカル・サイトに戻します。ローカル・サイトは、この情報を記録し、すべての接続先サイトに伝播されたエントリをローカル・キューからパージします。障害が発生した場合、ローカル・サイトは、適切な時点から伝播を続行できるように、正常に伝播されたトランザクションに関する情報をリモート・サイトに要求します。

データ伝播の依存性の維持

Oracle は、レプリケート・トランザクションがリモート・システムに伝播されたときの依存性の順序を維持します。次に例を示します。

1. トランザクション A が受注を取り消します。
2. トランザクション B が取消しを検出して返金を処理します。

トランザクション B は、ローカル・システムでの受注の取消し（トランザクション A）のコミット済の更新を参照するので、トランザクション B はトランザクション A に依存しています。

トランザクション A（受注の取消し）が正常に伝播された後で、トランザクション B（返金）が伝播されます。取消しの適用後に返金を処理する更新が適用されます。

パラレル伝播の依存性の追跡

ローカル・システムで新しいトランザクションが実行されるときに、次のことが行われます。

1. データを更新した最新のシステム・コミット番号が記録されます。これは、新規トランザクションにより従属システム・コミット番号として参照されます。
2. 従属システム・コミット番号以下のシステム・コミット番号を持つトランザクションが、リモート・システムに正常に伝播されます。
3. 待機中の依存トランザクションが伝播されます。

注意： トランザクション間で依存関係の可能性がないときは、トランザクションはパラレルで伝播されます。

パラレル伝播は、シリアル伝播とは異なる方法でデータの整合性を保持します。シリアル伝播では、依存性を維持するために、ローカル・システム上でコミットするときと同じ順序ですべてのトランザクションが適用されます。パラレル伝播では、依存性を追跡し、依存性があるときはコミットされた順に実行し、依存性がないときはパラレルで実行します。シリアル伝播でもパラレル伝播でも、トランザクション内の実行順序は保持されます。遅延トランザクションは各システムで、ローカル・トランザクション内で実行された順序と同じ順序で、リモート・プロシージャ・コールを実行します。

注意： シングル・コーディネータ・プロセスは、リモート・サイトに対する各データベース・リンクに存在しています。同じリモート・サイトに対する各データベース・リンクには、異なる接続修飾子が必要です。

関連項目： 2-14 ページの「接続修飾子」

パラレル化を改善するためのトランザクション依存性の最小化

アプリケーション条件によっては、トランザクション間に依存関係を確立して、遅延トランザクションの伝播を強制的にシリアライズ化できます。複数の非関連トランザクションによってレプリケート表内の同じデータ・ブロックが変更される場合、対応するトランザクションはリモート接続先へシリアル伝播されます。

データ・ブロック・レベルで作成されたトランザクション依存性を最小化するには、データ・ブロックの変更が 1 つまたは少数のデータ・ブロックに集中することを回避します。次に例を示します。

- レプリケート表に INSERT アクティビティが頻繁に発生する場合は、表の空きリストを複数作成することで、新しい行の記憶域を複数のデータ・ブロックに分散できます。
- 多数のトランザクションすべてが同一の小さい表を更新する状態は、可能な限り避けてください。たとえば、複数のトランザクションが 1 つの小さい表に読込みと更新を行い、主キーの順序番号生成をシミュレートするのは、望ましいアプリケーション設計とは言えません。このように設計すると、すべてのトランザクションが同じデータ・ブロックを更新します。より優れたソリューションは、順序を作成し順序番号をキャッシュに格納することで、主キーの生成を最適化し、パラレル伝播のパフォーマンスを改善できます。

7

レプリケーション環境の計画

レプリケーション環境の計画を開始する前に、これまでの章で説明したレプリケーションの概念およびアーキテクチャについて理解することが重要です。この章では、レプリケーションの概念およびアーキテクチャについて理解していることを前提に、レプリケーション環境の計画に関する重要な考慮事項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。

- レプリケート表に関する考慮事項
- 初期化パラメータ
- マスター・サイトおよびスナップショット・サイト
- スケジュール・リンクのガイドライン
- 遅延トランザクション・キューのスケジュール・パージのガイドライン
- シリアル伝播およびパラレル伝播
- 配置テンプレート
- 競合解消
- セキュリティ

レプリケート表に関する考慮事項

次の項では、レプリケーション環境で使用する表に関する考慮事項について説明します。

主キー

各レプリケート表には、可能な限り主キーを設定します。主キーを設定できない場合、各レプリケート表に、表の各行の一意の識別子として使用できる列のセットが含まれている必要があります。レプリケーション環境で使用する表に主キーまたは一意の列のセットがない場合は、レプリケート表を必要に応じて修正します。また、マスター表をベースに主キー・スナップショットを作成する場合は、マスター表に主キーを設定する必要があります。

データ型に関する考慮事項

マルチマスター・レプリケーションでは、NUMBER 型、DATE 型、VARCHAR2 型、CHAR 型、NVARCHAR2 型、NCHAR 型、RAW 型、ROWID 型を使用する列を含む表のレプリケーションをサポートしています。

また、バイナリ型の LOB（BLOB）、文字型の LOB（CLOB）、および各国文字型の LOB（NCLOB）といったラージ・オブジェクト型を使用する列を含む表のレプリケーションもサポートしています。部分単位の更新と追加が LOB 列に適用されると、マルチ・マスター・レプリケーションで使用する遅延および同期のリモート・プロシージャ・コール・メカニズムによって、LOB データ型に対する部分単位の変更のみが伝播されます。

注意： Oracle8*i* は、一部のサイトで Oracle7 リリース 7.3 が稼動しているレプリケーション環境での LOB 型のレプリケーションをサポートしていません。

Oracle では、LONG 型および LONG RAW 型を使用する列のレプリケーションはサポートしていません。これらのデータ型を含む列は、レプリケート表から単純に削除されます。Oracle8*i* では、LONG 型を LOB に変換する必要があります。

また、ユーザー定義オブジェクト型や、外部 LOB またはファイル・ベース LOB（BFILE）もサポートしていません。これらのデータ型の列を含む表をマスター表として構成すると、エラー・メッセージが戻されます。

初期化パラメータ

初期化パラメータには、レプリケーション環境の操作性、信頼性およびパフォーマンスにとって重要なものがあります。表 7-1 に、それらのパラメータを示します。

関連項目： これらの初期化パラメータの詳細は、『Oracle8*i* リファレンス・マニュアル』を参照してください。

表 7-1 レプリケーションにおいて重要な初期化パラメータ （1 / 5 ページ）

パラメータ	値	説明
COMPATIBLE	**デフォルト：** 8.0.0 **範囲：** 8.0.0 ～カレント・リリース番号	COMPATIBLE パラメータを使用すると、新しいリリースが使用でき、また古いリリースとの下位互換性も保証されます。これは、古いリリースを使用する必要が生じた場合に便利です。 このパラメータには、Oracle Server が互換性を保つ必要のあるリリースを指定します。 スナップショット・サイトの COMPATIBLE 設定が 8.1.0 より前の場合、各スナップショットのスナップショット・ビューが作成され、8.1.0 より前のリリースとの互換性が維持されます。スナップショット・サイトの COMPATIBLE 設定が 8.1.0 以降の場合は、スナップショットのスナップショット・ビューは作成されません。
ENQUEUE_RESOURCES	**デフォルト：** SESSIONS パラメータの値 **範囲：** 10 ～無限	ENQUEUE_RESOURCES パラメータには、ロック・マネージャによって同時にロックできるリソースの数を設定します。エンキューは高度なロック・メカニズムであり、同時に稼動している複数のプロセスによって、異なるレベルで認識済のリソースが共有されます。同時に使用できるオブジェクトはすべて、エンキューを使用して保護できます。たとえば、表を異なるレベルで共有する場合、2 つのプロセスが 1 つの表をそれぞれ共有モードまたは共有更新モードでロックできます。 Oracle Server が Java RepAPI をサポートしている場合は、ENQUEUE_RESOURCES パラメータを 1000 以上に設定する必要があります。
GLOBAL_NAMES	**デフォルト：** FALSE **範囲：** TRUE または FALSE	GLOBAL_NAMES パラメータには、データベース・リンクの名前を接続先データベースの名前と同じにするかどうかを指定します。Oracle レプリケーションを使用する場合は、GLOBAL_NAMES パラメータを TRUE に設定する必要があります。

表 7-1 レプリケーションにおいて重要な初期化パラメータ （2 / 5 ページ）

パラメータ	値	説明
JAVA_POOL_SIZE	**デフォルト：** 0 **範囲：** 0 ～ 4 GB	JAVA_POOL_SIZE パラメータには、Java プールのサイズをバイト単位で指定します。Java メモリー・マネージャは、実行時に、このプールから大部分の Java の状態を割り当てます。この領域には、メモリー上に展開された共有 Java メソッド、クラス定義、およびコールの終わりに Java セッション領域に移行される Java のオブジェクトが含まれます。 Oracle Server が Java RepAPI をサポートしている場合は、JAVA_POOL_SIZE パラメータを 50000000 以上に設定する必要があります。
JOB_QUEUE_INTERVAL	**デフォルト：** 60 （秒） **範囲：** 1 ～ 3600 （秒）	JOB_QUEUE_INTERVAL パラメータには、インスタンスの各 SNP*n* バックグラウンド・プロセスを起動する間隔を秒単位で指定します。このパラメータは、少なくとも最も頻繁に起動されるジョブと同じ値に設定する必要があります。
JOB_QUEUE_PROCESSES	**デフォルト：** 0 **範囲：** 0 ～ 36	JOB_QUEUE_PROCESSES パラメータには、各インスタンスの SNP*n* ジョブ・キュー・プロセスの番号を指定します（SNP0、...SNP9、SNPA、... SNPZ など）。ジョブ・キュー・プロセスでは、DBMS_JOB によって作成された要求が処理されます。 このパラメータは、1 以上で、かつ同時に実行できるジョブの最大数と同じ値に設定する必要があります。 サイトで JOB_QUEUE_PROCESSES を 0 に設定した場合、サイトのすべてのグループに対して手動で管理要求を適用し、遅延トランザクション・キューを手動でプッシュおよびパージする必要があります。

表 7-1 レプリケーションにおいて重要な初期化パラメータ （3 / 5 ページ）

パラメータ	値	説明
MAX_ENABLED_ROLES	**デフォルト：** 20 **範囲：** 0 ～ 148	MAX_ENABLED_ROLES パラメータには、その他のロールに含まれているロールを含め、ユーザーが使用可能にできるデータベース・ロールの最大数を指定します。 各ユーザーは PUBLIC ロールとユーザー自身のロールの 2 つを持っているため、ユーザーが使用可能にできる実際のロールの数は、MAX_ENABLED_ROLES の値に 2 を加えた数になります。たとえば、MAX_ENABLED_ROLES が 5 の場合、ユーザー SCOTT は 7 つのロールを使用可能にできます。つまり、MAX_ENABLED_ROLES で使用可能にした 5 つのロールと、PUBLIC のロールおよび SCOTT のロールです。 Oracle Server が Java RepAPI をサポートしている場合は、MAX_ENABLED_ROLES パラメータを 30 以上に設定する必要があります。

MTS_DISPATCHERS	**デフォルト：** なし
	
	範囲： なし
		MTS_DISPATCHERS パラメータを使用すると、マルチスレッド・サーバー（MTS）のアーキテクチャにディスパッチャ・プロセスが構成されます。解析ソフトウェアは name-value 構文をサポートしているので、位置独立で大文字小文字を区別しない属性を指定できます。
		
		Oracle Server が Java RepAPI をサポートしている場合は、MTS_DISPATCHERS パラメータを次のように設定する必要があります。
		
		`MTS_DISPATCHERS = "(PROTOCOL=TCP)`
		`(PRE=oracle.aurora.server.SGiopServer)"`

表 7-1 レプリケーションにおいて重要な初期化パラメータ （4 / 5 ページ）

パラメータ	値	説明
OPEN_CURSORS	**デフォルト：** 50 **範囲：** 1 ～ 4294967295 (4 GB -1)	OPEN_CURSORS パラメータには、セッションが一度に保持できるオープン・カーソル（プライベート SQL 領域に対するハンドル）の最大数を指定します。このパラメータを使用すると、セッションでカーソルをオープンしすぎるのを回避できます。また、このパラメータは、PL/SQL のカーソル・キャッシュのサイズも制約します。このカーソル・キャッシュは PL/SQL によって使用され、ユーザーが文を再実行するときの再解析を回避します。 アプリケーションのオープン・カーソル数が不足しないように、OPEN_CURSORS には十分に大きな値を設定することが重要です。この値は、アプリケーションによって異なります。OPEN_CURSORS に指定した数のカーソルがセッションでオープンされないとしても、実際に必要な数よりも大きい値を設定したことによってオーバーヘッドが増えることはありません。 Oracle Server が Java RepAPI をサポートしている場合は、OPEN_CURSORS を 100 以上に設定する必要があります。
OPEN_LINKS	**デフォルト：** 4 **範囲：** 0 ～ 255	OPEN_LINKS パラメータには、1 つのセッションで同時にオープンするリモート・データベース接続の最大数を指定します。この接続には、データベース・リンクと呼ばれるスキーマ・オブジェクトと、外部プロシージャおよびカートリッジ（それぞれ別のプロセスを使用します）が含まれます。 Oracle Server が Java RepAPI をサポートしている場合は、OPEN_LINKS を 10 以上に設定する必要があります。
PARALLEL_MAX_SERVERS	**デフォルト：** 次のパラメータの値を使用します。 CPU_COUNT PARALLEL_AUTOMATIC_TUNING PARALLEL_ADAPTIVE_MULTI_USER **範囲：** 0 ～ 3599	PARALLEL_MAX_SERVERS パラメータには、インスタンスのパラレル実行プロセスおよびパラレル・リカバリ・プロセスの最大数を指定します。インスタンス起動時に作成されるプロセスの数は、要求が増えるに従い、このパラメータに指定した値まで増えます。 パラレル伝播を使用する場合は、パラレル伝播をサポートするのに十分に大きい値を指定してください。

表 7-1 レプリケーションにおいて重要な初期化パラメータ （5 / 5ページ）

パラメータ	値	説明
PARALLEL_MIN_SERVERS	**デフォルト：** 0 **範囲：** 0 ～ PARALLEL_MAX_SERVERS の値	PARALLEL_MIN_SERVERS パラメータには、インスタンスのパラレル実行プロセスの最小数を指定します。この値は、インスタンスの起動時に作成されるパラレル実行プロセスの数です。 パラレル伝播を使用する場合、各ストリームに少なくとも 1 つのプロセスがあることを確認してください。
PROCESSES	**デフォルト：** PARALLEL_MAX_SERVERS の値 **範囲：** 6 ～オペレーティング・システムに固有の原価位置	PROCESSES パラメータには、Oracle に同時に接続できる、オペレーティング・システムのユーザー・プロセスの最大数を指定します。この値には、ロック、ジョブ・キュー・プロセスおよびパラレル実行プロセスなどの、すべてのバックグラウンド・プロセスを考慮する必要があります。
REPLICATION_DEPENDENCY_TRACKING	**デフォルト：** TRUE **範囲：** TRUE または FALSE	REPLICATION_DEPENDENCY_TRACKING パラメータは、データベースに対する読込み / 書込み操作の依存性の追跡を使用可能または使用禁止にします。依存性の追跡は、レプリケーション環境で変更をパラレル伝播するときは必須です。 TRUE: 依存性の追跡を使用可能にします。 FALSE: データベースに対する読込み / 書込み操作の実行を高速化しますが、パラレル伝播を実行するための依存性情報は生成しません。アプリケーションがレプリケート表に読込み / 書込み操作を実行しないことが確認できない限り、FALSE を指定しないでください。
SHARED_POOL_SIZE	**デフォルト：** 16 MB (64 ビットの場合は 64MB) **範囲：** 300 KB ～オペレーティング・システムに固有の限界値	SHARED_POOL_SIZE パラメータには、共有プールのサイズをバイト単位で指定します。共有プールには、共有カーソル、ストアド・プロシージャ、制御構造およびその他の構造が含まれます。この値を大きくするほど、マルチユーザー・システムのパフォーマンスは向上します。この値を小さくすれば、使用するメモリー量は減ります。 一般的に、Oracle Server の共有プールは、非レプリケーション環境よりもレプリケーション環境の場合のほうを大きくする必要があります。 共有プールの使用率は、ビュー V$SGASTAT を問い合せることにより監視できます。 Oracle Server が Java RepAPI をサポートしている場合は、SHARED_POOL_SIZE を 50000000 以上に設定する必要があります。

マスター・サイトおよびスナップショット・サイト

レプリケーション環境を計画するときは、レプリケーション環境に関係する各サイトをマスター・サイトにするかスナップショット・サイトにするかを決定する必要があります。これを決定する際は、両方のタイプのレプリケーション・サイトの特性と利点を考慮してください。1 つのレプリケーション環境で、マスター・サイトとスナップショット・サイトの両方をサポートできます。

表 7-2 マスター・サイトおよびスナップショット・サイトの特性

マスター・サイト	スナップショット・サイト
■ 通常、少数のマスター・サイトと通信しますが、多数のスナップショット・サイトとも通信する場合もあります。	■ 1 つのマスター・サイトと通信します。
■ 他のマスター・サイトのデータの完全なコピーである大量のデータを含んでいます。	■ マスター・サイトのデータのサブセットである少量のデータを含んでいます。
■ 通常、データ伝播の間隔は短く、継続的に通信します。	■ バルクデータ移送の間隔は長く、定期的に通信します。

注意： マスター・サイトでは、Oracle8*i* Enterprise Edition または Oracle8*i* Standard Edition を使用する必要があります。Oracle8*i* Standard Edition は、シングル・マスター・サイト構成のみをサポートしています。スナップショット・サイトでは、Oracle8*i* Enterprise Edition、Oracle8*i* Standard Edition、Oracle8*i* Personal Edition または Oracle8*i* Lite のいずれかが必要です。また、レプリケーションは、すべてのマスター・サイトおよびスナップショット・サイトにインストールする必要があります。

マスター・サイトの利点

マスター・サイトには次のような利点があります。

- リモート・サイトからのデータ・アクセスの可用性を高めるサポートを提供します。
- 変更が自動的にかつ通常は短い間隔で伝播されるので、データ操作言語（DML）による変更が頻繁に行われても適切に対処できます。
- 表をロックせずに、DML による変更とデータ伝播を同時に行うことができます。
- フェイルオーバーによる保護を提供できます。

マスター・サイトの設定には、Replication Manager の「Setup Wizard」またはレプリケーション・マネージメント API のいずれかを使用できます。

関連項目：

- Replication Manager の「Setup Wizard」を使用してマスター・サイトを設定する方法の詳細は、Replication Manager のオンライン・ヘルプを参照してください。
- レプリケーション・マネージメント API を使用してマスター・サイトを設定する方法の詳細は、『Oracle8*i* レプリケーション・マネージメント API リファレンス』の第 2 章「マスター・サイトの設定」を参照してください。
- フェイルオーバーによる保護を使用したレプリケーション環境の設計の詳細は、6-7 ページの「耐障害性を実現するための設計」を参照してください。

スナップショット・サイトの利点

スナップショット・サイトには次のような利点があります。

- モバイル・コンピューティングをサポートしています。
- マスター・サイトのデータのサブセットを含めることができます。

スナップショット・サイトの設定には、Replication Manager の「Setup Wizard」またはレプリケーション・マネージメント API のいずれかを使用できます。

関連項目：

- Replication Manager の「Setup Wizard」を使用してスナップショット・サイトを設定する方法の詳細は、Replication Manager のオンライン・ヘルプを参照してください。
- レプリケーション・マネージメント API を使用してスナップショット・サイトを設定する方法の詳細は、『Oracle8*i* レプリケーション・マネージメント API リファレンス』の第 2 章の「スナップショット・サイトの設定」を参照してください。

スケジュール・リンクのガイドライン

スケジュール・リンクによって、あるマスター・サイトから別のマスター・サイトへの遅延トランザクション・キューの伝播方法、あるいは、あるスナップショット・サイトからそのマスター・サイトへの遅延トランザクションの伝播方法が決定されます。スケジュール・リンクを作成すると、遅延トランザクション・キューをシステムの別のサイトにプッシュするジョブがローカル・ジョブ・キューに作成されます。リモート・マスター・サイトへの遅延トランザクションの伝播は、レプリケーション・プロパゲータのセキュリティ・コンテキスト内で行われます。

スケジュール・リンクを構成すると、シリアル伝播またはパラレル伝播を使用して情報をプッシュできます。一般的に、パラレル伝播値を 1 に設定した場合でも、パラレル伝播を使用する必要があります。

レプリケーション・システムのスケジュール・リンクを作成する前に、レプリケーションをシステム全体でグローバルに発生させる方法を慎重に検討する必要があります。たとえば、一定の間隔で遅延トランザクションを伝播できます。この場合、プッシュする頻度と時刻を決定する必要があります。また、リアルタイム・レプリケーション（同期レプリケーション）をシミュレートするために、各スケジュール・リンクを使用して、マスター・サイトの遅延トランザクション・キューをその接続先に絶えずプッシュする場合もあります。

あるいは、夜間など、必ず接続できる時間帯や通信コストが最低の時間帯にリンクをスケジュールする場合もあります。さらに、レプリケーションによるネットワーク・リソースへの負荷を分散するために、各マスター・サイト間でリンクのスケジュールをずらすこともできます。

> **関連項目：** シリアル伝播およびパラレル伝播に関連する問題の詳細は、7-12 ページの「シリアル伝播およびパラレル伝播」を参照してください。

定期的なプッシュのスケジューリング

サイトの遅延トランザクション・キューをリモートの接続先に定期的にプッシュするようにスケジュールできます。たとえば、1 日に 1 回または 1 時間に 1 回というように設定できます。Replication Manager の「Setup Wizard」、「Create Scheduled Link」プロパティ・シートまたは「Edit Scheduled Link」プロパティ・シートを使用してスケジュール・リンクを構成する場合、まず「Option」ページの「Delay Seconds」をデフォルト値の 0 に設定します。次に、「General」ページの「Next Date」および「Interval」を使用して、遅延トランザクション・キューをプッシュする間隔を設定します。

連続的なプッシュのスケジューリング

非同期レプリケーション・メカニズムを使用している場合でも、スケジュール・リンクを構成すると、連続的なリアルタイム・レプリケーションをシミュレートできます。Replication Manager の「Setup Wizard」、「Create Scheduled Link」プロパティ・シートまたは「Edit Scheduled Link」プロパティ・シートを使用してスケジュール・リンクを構成する場合、「Option」ページの「Delay Second」を 1200 に設定し、「Interval」を「Delay Seconds」の値より小さい値に設定します。このように設定すると、「Interval」に設定した時間中ずっと、遅延トランザクション・キューに入れられるトランザクションが連続的にプッシュされます。遅延時間中に伝播するトランザクションが遅延トランザクション・キューにない場合、ジョブが使用するリソースは解放されます。そして、次の SNP プロセスが使用可能になったときにリフレッシュが開始されます。

関連項目： 遅延秒数の設定の詳細は、2-39 ページの「遅延秒数」を参照してください。

遅延トランザクション・キューのスケジュール・パージのガイドライン

スケジュール・パージによって、マスター・サイトまたはスナップショット・サイトの遅延トランザクション・キューから適用済のトランザクションをパージする方法が決定されます。Replication Manager の「Setup Wizard」を使用してマスター・サイトまたはスナップショット・サイトを作成すると、各サイトのローカル・ジョブ・キューに、一定の基準でローカル遅延トランザクション・キューをパージするジョブが作成されます。レプリケーション環境のサイトを構成する前に、パージを発生させる方法を慎重に検討する必要があります。たとえば、次のような方法があります。

- サイトの遅延トランザクション・キューのプッシュ（スケジュール・リンク）とパージを同期化できます。たとえば、トランザクション・キューのプッシュとパージを連続的に行うように構成できます。このように構成した場合、パフォーマンスが向上することがあります。これは、最後にプッシュされたトランザクションに関する情報が、対応するパージ操作に使用するサーバーのバッファ・キャッシュにすでに存在していることが多いためです。
- サーバーが CPU の限界に達していない場合、遅延トランザクション・キューの連続的なパージをスケジュールすると、可能な限りキューのサイズを小さく保つことができます。
- 通常の営業時間帯にトランザクションのスループットが非常に大きくなるサーバーの場合、オフピーク時にパージが行われるようにスケジュールできます。

定期的なパージのスケジューリング

サイトの遅延トランザクション・キューを定期的にパージするようにスケジューリングできます。たとえば、1 日に 1 回または 1 週間に 1 回というように設定できます。Replication Manager の「Setup Wizard」、または「Edit DB Connection」プロパティ・シートの「Purge Scheduling」ページを使用してサイトのスケジュール・パージを構成する場合、まず「Delay Seconds」をデフォルトの 0 に設定します。次に、「Next Date」および「Interval」を使用して遅延トランザクション・キューをパージする間隔を設定します。

連続的なパージのスケジューリング

Replication Manager の「Setup Wizard」または「Edit DB Connection」プロパティ・シートの「Purge Scheduling」ページを使用してサイトの遅延トランザクション・キューの連続的なパージを設定するには、「Delay Seconds」を 500000 に設定し、「Interval」を「Delay Seconds」の値より小さい値に設定します。

シリアル伝播およびパラレル伝播

レプリケーション環境のスケジュール・リンクを作成すると、シリアル伝播またはパラレル伝播を使用して各サイトから接続先に変更を非同期伝播できます。レプリケーション環境を構成する前に、シリアル伝播またはパラレル伝播のいずれを使用するかを決定する必要があります。

- シリアル伝播では、レプリケート・トランザクションは、ソース・システムでコミットされたのと同じ順序で一度に 1 つずつ伝播されます。シリアル伝播を使用するスケジュール・リンクを構成するには、「Create Scheduled Link 」プロパティ・シートまたは「Edit Scheduled Link」プロパティ・シートを使用して、「Options」ページの「Parallel Propagation」設定を使用禁止にします。通常、シリアル伝播は、レプリケート環境内に Oracle7 を使用しているサイトがある場合にのみ使用します。
- パラレル伝播では、スループットを向上させるため、レプリケート・トランザクションは複数のパラレル・ストリームを使用して伝播されます。必要に応じて依存トランザクションの実行命令が発行されるので、データの整合性が保持されます。パラレル伝播を使用するスケジュール・リンクを構成するには、「Create Scheduled Link」プロパティ・シートまたは「Edit Scheduled Link」プロパティ・シートを使用して、「Options」ページの「Parallel Propagation」設定を使用可能にします。「Processes」を必要な並列度（1 以上）に設定します。

関連項目： 2-36 ページの「パラレル伝播」

配置テンプレート

レプリケーション環境にスナップショットを含める場合、配置テンプレートを使用してスナップショット・サイトにレプリケート・オブジェクトを作成できます。

関連項目： 配置テンプレートの詳細は、第 4 章「配置テンプレートの概念およびアーキテクチャ」を参照してください。

配置テンプレートをインスタンシエートするスナップショット・サイトの準備

配置テンプレートを使用する場合、配置テンプレートをインスタンシエートするスナップショット・サイトを準備する必要があります。配置テンプレートがうまく設計されていれば、リモート・スナップショット・サイトで必要な準備作業はほとんどありません。この項では、リモート・スナップショット・サイトで実行する最も一般的な準備作業について説明します。必要な準備作業がすべて完了した後で、オンライン・インスタンシエーションまたはオフライン・インスタンシエーションを実行できます。

次の点を考慮して、リモート・スナップショット・サイトでインスタンシエーションを実行するために必要な準備作業を決定します。

- リモート・スナップショット・サイトからターゲット・マスター・サイトにネットワーク接続されているかどうか。
- スナップショット・サイトで Oracle8*i* Enterprise Edition、Oracle8*i* Standard Edition または Oracle8*i* Personal Edition のリリース 8.1.5 以降のデータベースを使用しているかどうか。あるいは、スナップショット・サイトで Oracle8*i* Lite のリリース 4.0 以降のデータベースを使用しているかどうか。
- リモート・スナップショット・サイトがスナップショット・レプリケーションをサポートするように設定されているかどうか。
- 配置テンプレートに必要なスキーマがスナップショット・サイトに存在するかどうか。
- 必要なデータベース・リンクが配置テンプレートに含まれていない場合、スナップショット・サイトからマスター・サイトに必要なデータベース・リンクが存在するかどうか。
- スナップショットで配置テンプレートをインスタンシエートするために、オンライン・インスタンシエーションまたはオフライン・インスタンシエーションのいずれを使用するか。
- 配置テンプレートに必要なロールバック・セグメントがスナップショット・サイトに存在していて、かつそれがオンラインであるかどうか。

次の項では、これらの考慮事項に対する指針を示します。

ネットワーク接続性

ネットワーク接続性は、すべてのレプリケート環境において Oracle レプリケーションの重要な要素です。Oracle8*i* Enterprise Edition、Oracle8*i* Standard Edition および Oracle8*i* Personal Edition を使用している場合、リモート・スナップショット・サイトからターゲット・マスター・サイトに対して SQL*Net または Net8 による適切なネットワーク接続が確立されていることを確認してください。

> **関連項目：** SQL*Net または Net8 を使用したネットワーク接続の設定の詳細は、『Oracle8*i* Net8 管理者ガイド』を参照してください。

Java RepAPI を使用する Oracle8*i* Lite スナップショット・サイトの場合、マスター・サイトとの通信には Internet Inter-ORB Protocol（IIOP）によるネットワーク接続が必要です。

> **関連項目：** IIOP を使用するスナップショット・クライアントの設定の詳細は、Oracle8*i* Lite のドキュメントを参照してください。このクライアントの接続先サーバーの設定の詳細は、付録 C「RepAPI 用の Oracle8i Server の構成」を参照してください。

データベースのバージョン

スナップショット・サイトで配置テンプレートをインスタンシエートするには、Oracle8*i* リリース 8.1.5 以降のデータベースを使用している必要があります。Oracle8*i* Lite を使用している場合は、リリース 4.0 以降がインストールされている必要があります。スナップショット・サイトがこれらの要件を満たしていない場合は、配置テンプレートをインスタンシエートする前にデータベースをアップグレードする必要があります。

スナップショット・サイトの設定

各スナップショット・サイトには、スナップショット・サイトをサポートするための特別な権限を持ったユーザーが何人か必要です。これらのユーザーは、管理に必要な権限を持ち、さらにデータの伝播とリフレッシュも実行します。

スナップショット・サイトの設定では、スナップショットの自動リフレッシュ（オプション）や遅延トランザクション・キューのパージを実行するいくつかの自動化ジョブをスケジュールする必要もあります。

スナップショット・サイトを設定するには、次のいずれかを使用します。

- **Replication Manager:** Replication Manager を使用してリモート・スナップショット・サイトに接続し、「Setup Wizard」を使用してスナップショット・サイトを設定できます。

  > **関連項目：** Replication Manager を使用してスナップショット・サイトを設定する方法の詳細は、Replication Manager のオンライン・ヘルプを参照してください。

- **レプリケーション・マネージメント API:** Replication Manager でリモート・スナップショット・サイトに接続できない場合は、レプリケーション・マネージメント API を使用してスナップショット・サイトを設定します。スナップショット・サイトを設定する API コールを含む SQL スクリプトを作成するときに、その他の準備作業（必要なスキーマ、データベース・リンクおよびロールバック・セグメントの作成など、次の 3 つの項で説明する作業）を実行するための DDL と API コールも追加できます。作成したスクリプトは、オフライン・インスタンシエーション・ファイルとともに配布し、オフライン・インスタンシエーション・ファイルよりも前に実行する必要があります。

  関連項目： レプリケーション・マネージメント API を使用してスナップショット・サイトを設定する方法の詳細は、『Oracle8*i* レプリケーション・マネージメント API リファレンス』の第 2 章の「スナップショット・サイトの設定」を参照してください。

必要なスキーマの作成

インスタンシエートしようとする配置テンプレートで複数のスキーマ内にオブジェクトを作成する場合、必要なスキーマがすべて作成済であることを確認する必要があります。また、配置テンプレートをインスタンシエートするユーザーは、そのスキーマに対する適切な CREATE 権限を持っている必要があります。たとえば、配置テンプレートでスキーマ MARY にプロシージャを作成し、ユーザー SCOTT がテンプレートをインスタンシエートする場合、SCOTT にはスキーマ MARY に対する CREATE ANY PROCEDURE 権限が必要です。

データベース・リンクの作成

リモート・スナップショット・サイト用に必要なすべてのデータベース・リンクを作成する DDL を配置テンプレートに含めると便利ですが、必ずしもそうする必要はありません。データベース・リンク DDL が配置テンプレートに含まれていない場合は、配置テンプレートをインスタンシエートする前に、ターゲット・マスター・サイトへのデータベース・リンクを手動で作成します。データベース・リンクは、オンライン・インスタンシエーション時にスナップショットのベース表にデータを移入したり、スナップショット環境を適切にメンテナンスするために必要です。

オンライン・インスタンシエーションおよびオフライン・インスタンシエーション

配置テンプレートをスナップショット・サイトでインスタンシエートするときは、オンライン・インスタンシエーションまたはオフライン・インスタンシエーションのいずれかを選択できます。オンライン・インスタンシエーションでは、スナップショットのデータはインスタンシエーション時にマスター・サイトから取り出されます。オフライン・インスタンシエーションでは、スナップショットのデータはテンプレート自体にパッケージ化されており、テンプレートをインスタンシエートするときにローカルで適用されます。一般的に、スナップショットに大量のデータが含まれている場合は、オフライン・インスタンシエーションの方がネットワーク・リソースの使用率を最小化できます。

関連項目： オンライン・インスタンシエーションおよびオフライン・インスタンシエーションの詳細は、4-9 ページの「配置テンプレートのパッケージ化およびインスタンシエーション」を参照してください。

必要なロールバック・セグメントの作成

インスタンシエートしている配置テンプレートで、現在リモート・スナップショット・サイトに存在しない特定のロールバック・セグメントを使用する場合は、必要なロールバック・セグメントを作成します。テンプレート・オブジェクトがデフォルトのロールバック・セグメントと特定のロールバック・セグメントのうちのどちらを使用しているのかを確認するには、DBA_REPCAT_TEMPLATE_OBJECTS ビューを問い合せます。

競合解消

非同期マルチマスター・レプリケーション環境および更新可能スナップショット・レプリケーション環境では、たとえば異なるサイトで発行された 2 つのトランザクションによってほぼ同時に同じ行が更新されたときなどに発生するレプリケーション競合に対処できる必要があります。可能な限り、競合の発生を回避できるようにレプリケーション環境を計画してください。レプリケーション環境でデータ競合が発生した場合は、ビジネス・ルールに従って競合を解消し、すべてのサイトでデータを適正に収束することを保証するメカニズムが必要です。

関連項目： 競合の回避および競合が発生した場合の競合解消方法の詳細は、「競合解消の概念およびアーキテクチャ」を参照してください。

セキュリティ

セキュリティは、マルチマスター・レプリケーション環境およびスナップショット・レプリケーション環境のどちらの環境でも重要な課題です。レプリケーション環境を構成する場合、まずセキュリティ対策を講じる必要があります。

関連項目： レプリケーション環境のセキュリティ・オプションの詳細は、『Oracle8*i* レプリケーション・マネージメント API リファレンス』の付録 A「セキュリティ・オプション」を参照してください。

8

Replication Manager の概要

Oracle Replication Manager では、グラフィカル・ユーザー・インタフェース（GUI）を使用してレプリケーション環境を設定、管理および監視できます。また、多くの重要な操作をウィザードに従って実行できます。Replication Manager を使用すると、マルチマスター・レプリケーション環境およびスナップショット・レプリケーション環境の両方を管理できます。Replication Manager は、Oracle Enterprise Manager と統合されています。

この章では、Replication Manager の機能について説明します。ただし、Replication Manager の使用方法の詳細は、Replication Manager のオンライン・ヘルプを参照してください。この章のトピックは次のとおりです。

- Replication Manager の用途
- Replication Manager の起動
- Replication Manager のインタフェース
- Replication Manager のウィザード
- レプリケート環境を作成するためのフローチャート

関連項目： Oracle Enterprise Manager の使用方法の詳細は、Oracle Enterprise Manager のドキュメント・セットおよびオンライン・ヘルプを参照してください。

Replication Manager の用途

Replication Manager を使用すると、次のことを行うことができます。

- マスター・サイトおよびスナップショット・サイトの設定
- レプリケーション環境でのマスター・サイトの追加および削除
- マスター・グループの作成および管理
- 接続先マップを使用したマスター・グループおよびマスター・サイトの監視
- スナップショット・ログの作成および管理
- スナップショット・グループの作成および管理
- 個々のスナップショットの作成および管理
- スナップショットのリフレッシュ・グループの作成および管理
- 配置テンプレートの作成および管理
- オフライン・インスタンシエーション用配置テンプレートのパッケージ化（オンライン・インスタンシエーション用の配置テンプレートをパッケージ化する場合は、レプリケーション・マネージメント API を使用する必要があります）
- 競合解消方法の設定
- スケジュール・リンクの作成、監視および管理
- 管理要求の監視および管理
- 遅延トランザクションの監視および管理
- エラー・トランザクションの監視および管理
- ローカル・ジョブの作成、監視および管理

Replication Manager の起動

Replication Manager は、Oracle Enterprise Manager Console を使用して起動するか、または独立のアプリケーションとして起動します。

Oracle Enterprise Manager Console からの Replication Manager の起動

Oracle Enterprise Manager Console から起動するには、次の順に選択します。

「Tools」>「Extended Database Applications」>「Oracle Replication Manager」

独立のアプリケーションとしての Replication Manager の起動

Replication Manager を独立のアプリケーションとして起動する方法は、使用しているオペレーティング・システムによって異なります。

Windows プラットフォームの場合 : 「スタート」メニューから次の順に選択します。

「プログラム」>「Oracle - *OEM_LABEL* 」> 「Extended Database Administration」>「Replication Manager」

OEM_LABEL には、Oracle Enterprise Manager のインストール時に指定されたラベルが入ります。

UNIX システムの場合 : コマンド・プロンプトに次のように入力します。

```
oemapp repmgr
```

Replication Manager へのログイン

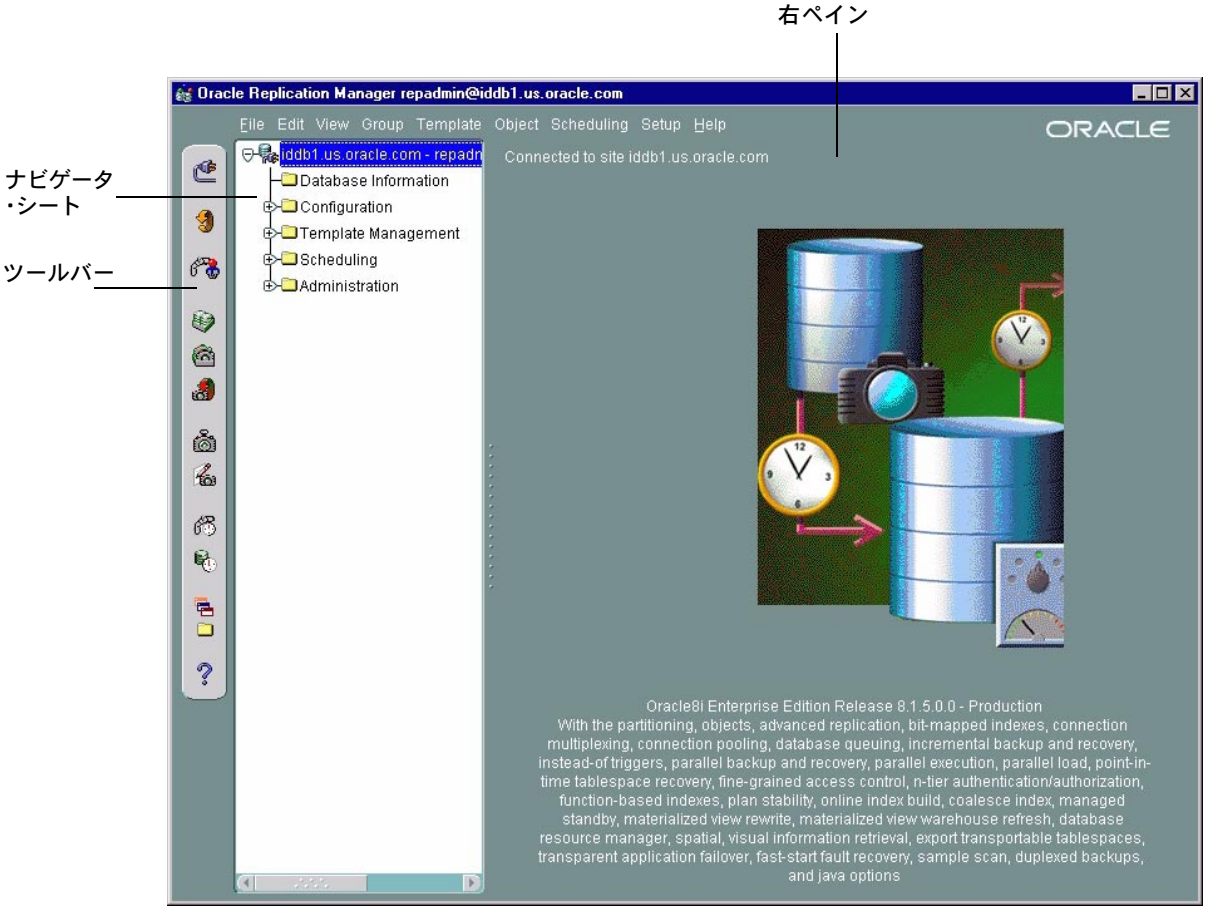
Replication Manager に最初にログインするときは、SYSTEM または SYS でログインする必要があります。SYSTEM または SYS でログインした後で、「Setup Wizard」を使用して、マスター定義サイトと、その他のマスター・サイト（マルチマスター・レプリケーション環境を使用する場合）を設定します。

設定が完了した後で、「Setup Wizard」で指定したレプリケーション管理者として Replication Manager にログインします。規則に従えば、レプリケーション管理者のユーザー名は REPADMIN です。設定完了後は、Replication Manager にログインできるのはレプリケーション管理者のみとし、その他のユーザーはログインしないでください。

Replication Manager のインタフェース

Replication Manager のインタフェースには、2 つのペイン、1 つのツールバーおよび様々なメニュー項目が含まれています。

図 8-1 Replication Manager のインタフェース



ペイン

Replication Manager のインタフェースには、ナビゲータ・シートおよび右ペインの 2 つのペインがあります。

ナビゲータ・シート

Replication Manager のナビゲータ・シートの機能は、Oracle Enterprise Manager の他のアプリケーションのナビゲータ・シートと同じです。ナビゲータ・シートでは、次の処理を実行できます。

- レプリケーション環境内のすべてのノードへのアクセス（Replication Manager を Oracle Enterprise Manager Console から起動した場合）。
- コンテナを拡大および縮小して、監視または管理対象のオブジェクトのナビゲートができます。オブジェクトには、マスター・グループ、スナップショット・グループ、スナップショット、スナップショット・ログ、配置テンプレートなどがあります。
- オブジェクト上の右クリック。オブジェクトに対する操作が実行できます。

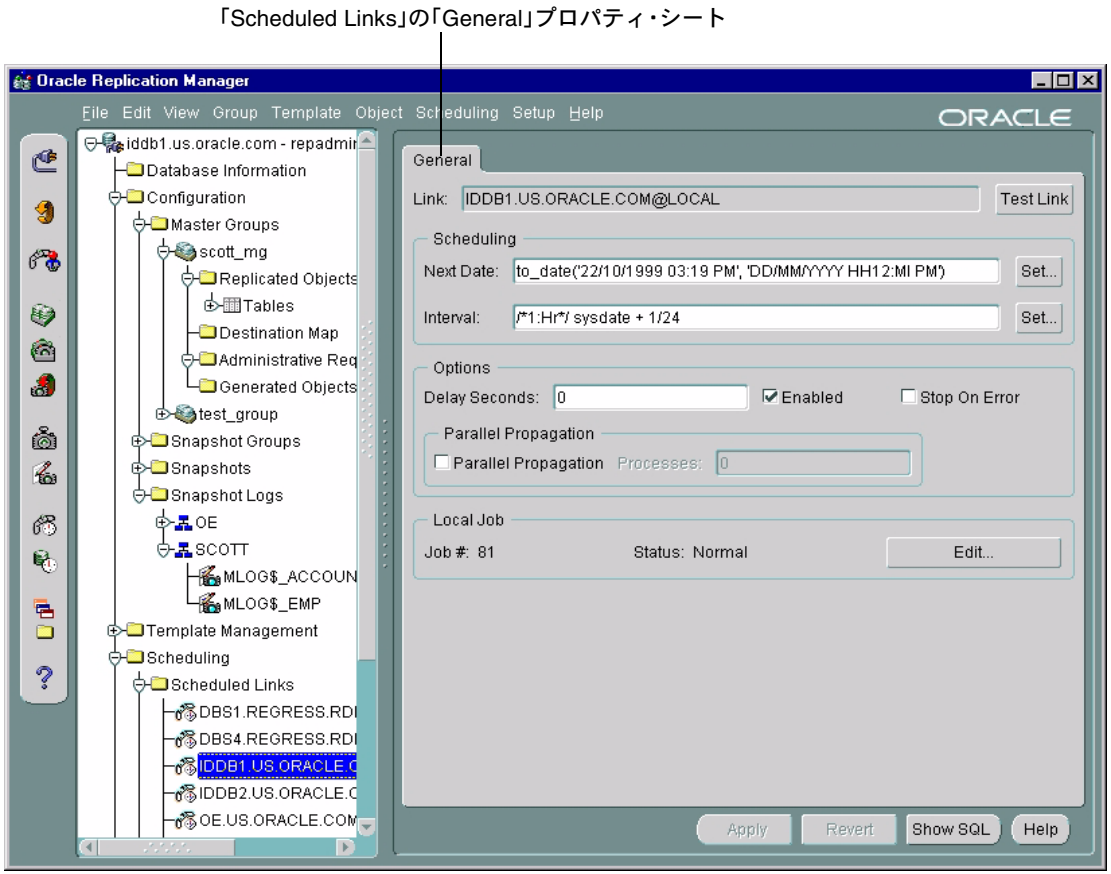
Replication Manager のナビゲータ・シートには、次の主要なコンテナがあります。

「Database Information」	選択したデータベースの一般的なプロパティ、およびデータベースの遅延トランザクション・キューのパージ・スケジューリングを表示および編集できます。
「Configuration」	マスター・グループ、スナップショット・グループ、スナップショットおよびスナップショット・ログを管理できます。
「Template Management」	配置テンプレート、配置テンプレートの認証ユーザー、および配置テンプレートをインスタンシエートしたサイトを管理できます。
「Scheduling」	スケジュール・リンクおよびリフレッシュ・グループを管理できます。
「Administration」	遅延トランザクション、ローカル・エラーおよびローカル・ジョブを管理できます。

右ペイン

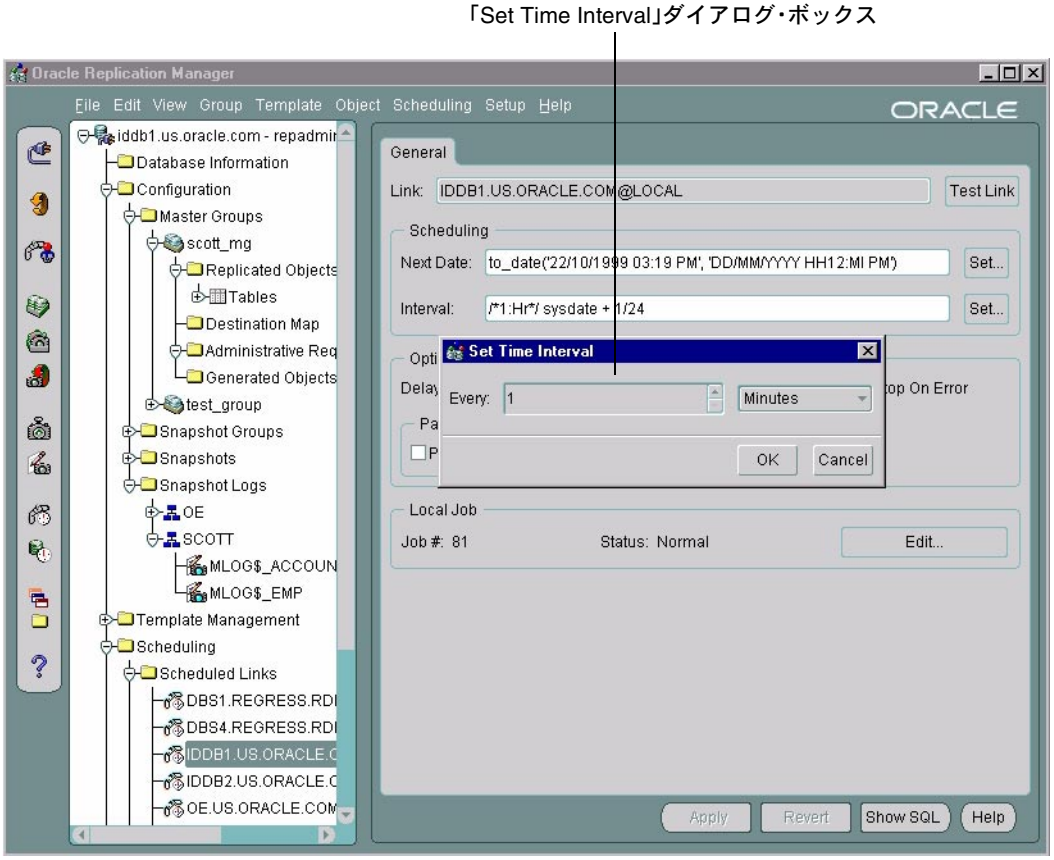
Replication Manager の右ペインには、レプリケーション環境の監視を可能にする情報とレプリケーション環境の管理を可能にするプロパティ・シートが含まれています。図 8-2 は、プロパティ・シートの例を示します。

図 8-2 Replication Manager で「Scheduled Links」を開いたときに表示される「General」プロパティ・シート



プロパティ・シート上のボタンをクリックすると、新しいダイアログ・ボックスが開きます。たとえば、「Interval」フィールドと対応付けられている「Set」ボタンをクリックすると、図 8-3 に示すように、「Set Time Interval」ダイアログ・ボックスが表示されます。

図 8-3 Replication Manager の「Set Time Interval」ダイアログ・ボックス



Replication Manager の右ペインでは、レプリケーション環境の監視も行えます。使用できる監視ツールの例として「Destination Map」があります。「Destination Map」には、次のような情報が表示されます。

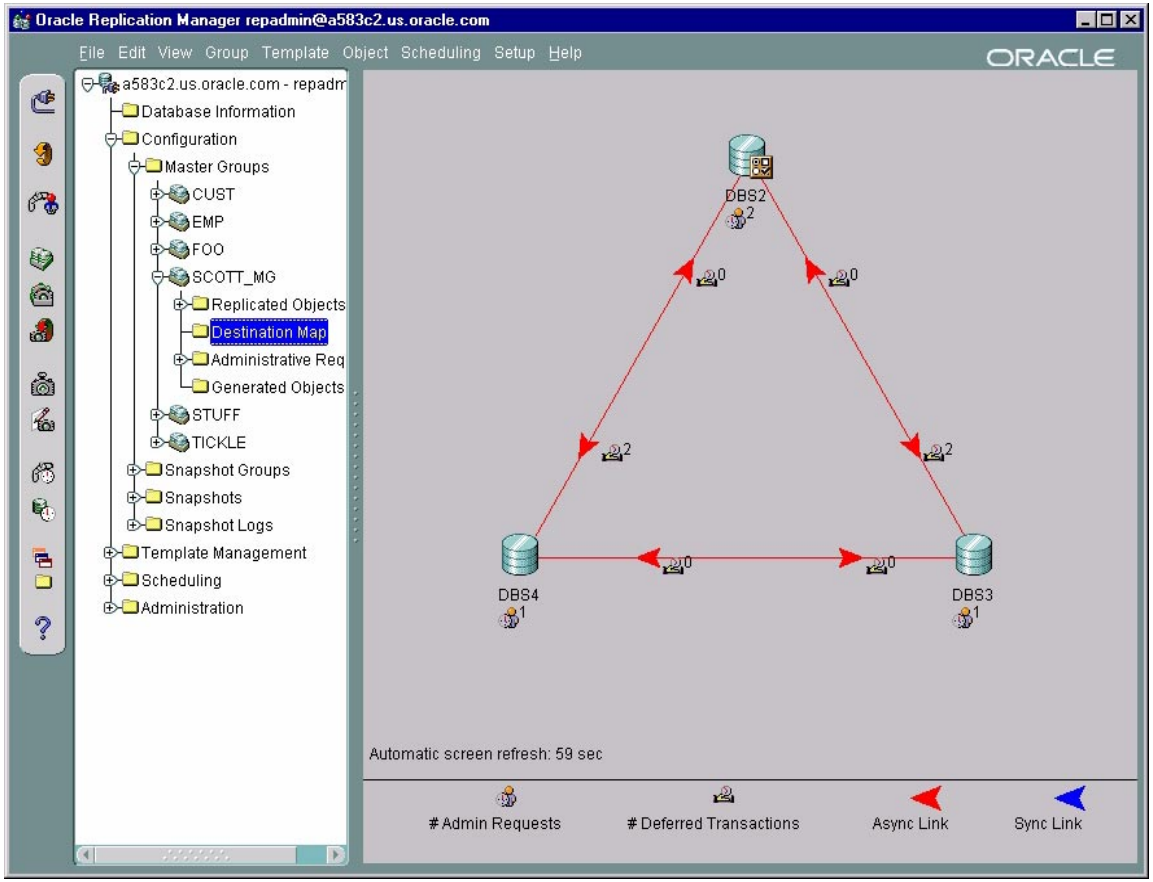
- 選択したマスター・グループ内のすべてのマスター・サイト
- 各マスター・サイトの遅延トランザクションの数
- 各マスター・サイトの管理要求の数
- マスター・サイトで使用しているレプリケーションのタイプ（非同期または同期）
- マスター・サイトでのローカル・エラーの有無

たとえば、図 8-4 の「Destination Map」では次のことが分かります。

- データベース DBS2 には、未処理の管理要求が 2 つあります。
- データベース DBS2 の遅延トランザクション・キューには、DBS3 で適用される 2 つの遅延トランザクションがあります。
- データベース DBS2 の遅延トランザクション・キューには、DBS4 で適用される 2 つの遅延トランザクションがあります。
- データベース DBS3 および DBS4 には、それぞれ未処理の管理要求が 1 つあります。

さらに、「Destination Map」で右クリックすると、「Destination Map」のリフレッシュや、表示されているマスター・グループの編集などのアクションを実行できるメニューが開きます。

図 8-4 Replication Manager の「Destination Map」



ツールバー

Replication Manager のツールバーを使用すると、レプリケーション環境で頻繁に実行する操作を迅速に行えます。ツールバーには、次のようなボタンがあります。

「Change Database Connection」

「Database Connect Information」ダイアログ・ボックスが開きます。このダイアログ・ボックスを使用すると、別のデータベースに接続できます。

> **関連項目：** データベース接続の変更方法の詳細は、Replication Manager のオンライン・ヘルプの「Changing the Database Connection」のトピックを参照してください。このトピックを参照するには、「Help Contents」で「Replication Administration」>「Miscellaneous」の順に選択します。

「Refresh」

Replication Manager のインタフェースをリフレッシュして、最新の情報を表示します。Replication Manager のインタフェースは、自動的にはリフレッシュされません。したがって、レプリケーション環境の様々な側面（遅延トランザクション・キューなど）を監視するには、インタフェースの手動リフレッシュが重要です。

「Setup Wizard」

Replication Manager の「Setup Wizard」が開きます。このウィザードの指示に従って操作すると、マスター・サイトおよびスナップショット・サイトを設定できます。Replication Manager の「Setup Wizard」を使用するときは、必ず SYS または SYSTEM でログインしてください。

> **関連項目：** Replication Manager の「Setup Wizard」の詳細は、8-18 ページの「Setup Wizard」を参照してください。

「Create Master Group」

「Create Master Group」ダイアログ・ボックスが開きます。このダイアログ・ボックスを使用すると、新しいマスター・グループを作成できます。マスター・グループを作成するときは、接続先サイトがマスター・グループのマスター定義サイトになります。

> **関連項目：** 「Create Master Group」ダイアログ・ボックスを使用してマスター・グループを作成する方法の詳細は、Replication Manager のオンライン・ヘルプの「Flowchart for Creating a Master Group」のトピックを参照してください。このトピックを参照するには、「Help Contents」で「Multimaster Replication」>「Create Master Group」の順に選択します。

「Create Snapshot Group」

「Snapshot Group Wizard」が開きます。このウィザードの指示に従って操作すると、スナップショット・グループを設定できます。スナップショット・グループを作成するときは、スナップショット・グループに含めるスナップショットが存在するスナップショット・サイトに接続していることを確認してください。

> **関連項目：**「Snapshot Group Wizard」の詳細は、8-20 ページ の「Snapshot Group Wizard」を参照してください。

「Create Refresh Group」

「Create Refresh Group」ダイアログ・ボックスが開きます。このダイアログ・ボックスを使用すると、新しいリフレッシュ・グループを作成できます。リフレッシュ・グループを作成するときは、リフレッシュ・グループに含めるスナップショットが存在するスナップショット・サイトに接続していることを確認してください。

> **関連項目：**「Create Refresh Group」ダイアログ・ボックスを使用してリフレッシュ・グループを作成する方法の詳細は、Replication Manager のオンライン・ヘルプの「Creating a Refresh Group」のトピックを参照してください。このトピックを参照するには、「Help Contents」で「Snapshot Replication」>「Manage」>「Refresh Groups」の順に選択します。

「Create Snapshot」

「Create Snapshot」ダイアログ・ボックスが開きます。このダイアログ・ボックスを使用すると、新しいスナップショットを作成できます。スナップショットを作成するときは、スナップショットを作成するスナップショット・サイトに接続していることを確認してください。

> **関連項目：**「Create Snapshot」ダイアログ・ボックスを使用してスナップショットを作成する方法の詳細は、Replication Manager のオンライン・ヘルプの「Creating a Snapshot」のトピックを参照してください。このトピックを参照するには、「Help Contents」で「Snapshot Replication」>「Manage」>「Snapshots」の順に選択します。

「Create Snapshot Log」

「Create Snapshot Log」ダイアログ・ボックスが開きます。このダイアログ・ボックスを使用すると、マスター表の新しいスナップショット・ログを作成できます。スナップショット・ログを作成するときは、スナップショット・ログを作成するマスター表が存在するマスター・サイトに接続していることを確認してください。

関連項目：「Create Snapshot Log」ダイアログ・ボックスを使用してスナップショット・ログを作成する方法の詳細は、Replication Manager のオンライン・ヘルプの「Creating Snapshot Logs」のトピックを参照してください。このトピックを参照するには、「Help Contents」で「Snapshot Replication」を選択します。

「Create Scheduled Link」

「Create Scheduled Link」ダイアログ・ボックスが開きます。このダイアログ・ボックスを使用すると、新しいスケジュール・リンクを作成できます。

関連項目：「Create Scheduled Link」ダイアログ・ボックスを使用してスケジュール・リンクを作成する方法の詳細は、Replication Manager のオンライン・ヘルプの「Creating a Scheduled Link」のトピックを参照してください。このトピックを参照するには、「Help Contents」で「Replication Administration」>「Scheduled Links」の順に選択します。

「Create Local Job」

「Create Local Job」ダイアログ・ボックスが開きます。このダイアログ・ボックスを使用すると、新しいローカル・ジョブを作成できます。

関連項目：「Create Local Job」ダイアログ・ボックスを使用してローカル・ジョブを作成する方法の詳細は、Replication Manager のオンライン・ヘルプの「Creating a Local Job」のトピックを参照してください。このトピックを参照するには、「Help Contents」で「Replication Administration」>「Local Jobs」の順に選択します。

「Create Template」

「Deployment Template Wizard」が開きます。このウィザードの指示に従って操作すると、新しい配置テンプレートを作成できます。配置テンプレートを作成するときは、配置テンプレートに含めるオブジェクトが存在するマスター・サイトに接続していることを確認してください。

関連項目：「Deployment Template Wizard」の詳細は、8-22 ページの「Deployment Template Wizard」を参照してください。

「Offline Instantiation File Generation」

「Offline Instantiation Wizard」が開きます。このウィザードの指示に従って操作すると、配置テンプレートのパッケージ化およびオフライン・インスタンシエーション・ファイルの生成を行えます。Oracle8*i* Lite および Oracle Server（Oracle8*i* Enterprise Edition、Oracle8*i* Standard Edition および Oracle8*i* Personal Edition を含みます）用のオフライン・インスタンシエーション・ファイルを生成できます。オフライン・インスタンシエーション・ファイルを生成するときは、配置テンプレートが存在するマスター・サイトに接続していることを確認してください。

関連項目：「Offline Instantiation Wizard」の詳細は、8-24 ページの「Offline Instantiation Wizard」を参照してください。

注意： オンライン・インスタンシエーション・ファイルを生成する場合は、レプリケーション・マネージメント API を使用する必要があります。オンライン・インスタンシエーション・ファイルの生成の詳細は、『Oracle8*i* レプリケーション・マネージメント API リファレンス』を参照してください。

「Help」

Replication Manager のオンライン・ヘルプが開きます。

メニュー

Replication Manager のインタフェースのリフレッシュなどのアクションを実行したり、ウィザードやダイアログ・ボックスを開くときは、Replication Manager のメニューを使用します。各メニューのメニュー項目を次に示します。

「File」メニュー

「File」メニューのメニュー項目は、Oracle Enterprise Manager Console を使用して Replication Manager を起動した場合と、独立のアプリケーションとして Replication Manager を起動した場合とでは異なります。

Replication Manager を Oracle Enterprise Manager Console を使用して起動した場合、「File」メニューのメニュー項目は次のとおりです。

「Connect」	ナビゲータ・シートで選択したデータベースに接続できます。
「Disconnect」	ナビゲータ・シートで選択したデータベースから接続を切断できます。
「Exit」	Replication Manager を終了できます。

Replication Manager を独立のアプリケーションとして起動した場合の、「File」メニューのメニュー項目は次のとおりです。

「Change Database Connection」	データベース接続を変更できます。独立のアプリケーションとして接続する場合は、一度に接続できるデータベースは 1 つのみです。
「Exit」	Replication Manager を終了できます。

「Edit」メニュー

「Edit」メニューのメニュー項目は、次のとおりです。

「Preferences」

マスター・グループ、スナップショット、リフレッシュ・グループ、スケジュール・リンクおよびローカル・ジョブのデフォルト設定を編集できます。デフォルトとして指定した設定は、新しいオブジェクトを作成するときに適用されます。

たとえば、新しいマスター・グループを作成する場合、「Copy Row Data」はオプションとなっています。「Copy Row Data」をデフォルトとして使用可能にすると、新しいマスター・グループを作成するたびにこのオプションが使用可能になります。ただし、マスター・グループの作成時に必要に応じてこのオプションを使用禁止にすることもできます。

注意： 「Preferences」メニューの設定を変更しても、その変更が適用されるのはその時点の Replication Manager セッションのみです。Replication Manager をいったん閉じてから再起動すると、「Preferences」メニューのメニュー項目はデフォルト設定に戻ります。

「View」メニュー

「View」メニューのメニュー項目は、次のとおりです。

「Refresh」

Replication Manager のインタフェースをリフレッシュできます。ツールバーの「Refresh」ボタンと同じアクションが実行されます。

「Group」メニュー

「Group」メニューのサブメニューおよびメニュー項目は、次のとおりです。

「Master Group」

「Master Group」サブメニューには、次のメニュー項目があります。

「Create」： 「Create Master Group」ダイアログ・ボックスが開きます。このダイアログ・ボックスを使用すると、新しいマスター・グループを作成できます。

「Add Object」： 「Add Object(s) to Group」ダイアログ・ボックスが開きます。このダイアログ・ボックスを使用すると、選択したマスター・グループにオブジェクトを追加できます。

「Remove」： 選択したマスター・グループを削除できます。

「Snapshot Group」　「Snapshot Group」サブメニューには、次のメニュー項目があります。

「Create」：「Snapshot Group Wizard」が開きます。このウィザードを使用すると、新しいスナップショット・グループを作成できます。

「Add Object」：「Snapshot Group Edit Wizard」が開きます。このウィザードを使用すると、選択したスナップショット・グループにオブジェクトを追加できます。

「Remove」： 選択したスナップショット・グループを削除できます。

「Template」メニュー

「Template」メニューのサブメニューおよびメニュー項目は、次のとおりです。

「Create」　「Deployment Template Wizard」が開きます。このウィザードの指示に従って操作すると、配置テンプレートを作成できます。

「Copy」　「Copy Template Wizard」が開きます。このウィザードを使用すると、配置テンプレートを同じマスター・サイトまたは別のマスター・サイトにコピーできます。

「Compare」　「Compare Deployment Templates」ダイアログ・ボックスが開きます。このダイアログ・ボックスを使用すると、2 つの配置テンプレートを比較できます。

「Add」　「Add」サブメニューには、次のメニュー項目があります。

「Object」： 選択した配置テンプレートをインスタンシエートしていない場合、テンプレートにオブジェクトを追加できます。

「Parameter」： 選択した配置テンプレートをインスタンシエートしていない場合、テンプレートにパラメータを追加できます。

「Authorized User」： プライベート配置テンプレートに認証ユーザーを追加できます。

「Instantiate」　「Offline Instantiation Wizard」が開きます。このウィザードの指示に従って操作すると、配置テンプレートのオフライン・インスタンシエーション・ファイルを生成できます。

「Remove」　選択した配置テンプレートを削除できます。配置テンプレートを削除しても、そのテンプレートをインスタンシエートしたリモート・スナップショット・サイトに影響はありません。

「Drop Instantiated Site」 インスタンシエート済のスナップショット・サイトを選択した場合、そのサイトを削除できます。インスタンシエート済のサイトを選択するには、ナビゲータ・シートで「Template」>「*template_name* > Sites」を選択します。*template_name* は、配置テンプレートの名前です。

「Object」メニュー

「Object」メニューのメニュー項目は、次のとおりです。

「Create Snapshot」 「Create Snapshot」ダイアログ・ボックスが開きます。このダイアログ・ボックスを使用すると、スナップショット・サイトでスナップショットを作成できます。

「Create Snapshot Log」 「Create Snapshot Log」ダイアログ・ボックスが開きます。このダイアログ・ボックスを使用すると、マスター・サイトのマスター表にスナップショット・ログを作成できます。

「Remove」 選択したスナップショットまたはスナップショット・ログを削除できます。

「Scheduling」メニュー

「Scheduling」メニューのサブメニューおよびメニュー項目は、次のとおりです。

「Create Scheduled Link」 「Create Scheduled Link」ダイアログ・ボックスが開きます。このダイアログ・ボックスを使用すると、新しいスケジュール・リンクを作成できます。

「Create Local Job」 「Create Local Job」ダイアログ・ボックスが開きます。このダイアログ・ボックスを使用すると、新しいローカル・ジョブを作成できます。

「Refresh Group」 「Refresh Group」サブメニューには、次のメニュー項目があります。

「Create」: 「Create Refresh Group」ダイアログ・ボックスが開きます。このダイアログ・ボックスを使用すると、新しいリフレッシュ・グループを作成できます。

「Add Object」: 「Add Snapshot(s) to Refresh Group」ダイアログ・ボックスが開きます。このダイアログ・ボックスを使用すると、選択したリフレッシュ・グループにスナップショットを追加できます。

「Remove」: 選択したリフレッシュ・グループを削除できます。

「Remove」 選択したスケジュール・リンクまたはローカル・ジョブを削除できます。

「Setup」メニュー

「Setup」メニューのメニュー項目は、次のとおりです。

「Master Sites」	Replication Manager の「Setup Wizard」が開きます。このウィザードの指示に従って操作すると、マスター・サイトを設定できます。
「Snapshot Sites」	Replication Manager の「Setup Wizard」が開きます。このウィザードの指示に従って操作すると、スナップショット・サイトを設定できます。

「Help」メニュー

「Help」メニューのメニュー項目は、次のとおりです。

「Contents」	Replication Manager のオンライン・ヘルプが開きます。
「About Replication Manager」	Replication Manager のバージョン番号および著作権情報を表示するウィンドウが開きます。

Replication Manager のウィザード

Replication Manager のウィザードは、複数のステップが必要な作業に対して画面ごとに指示を提供します。ウィザードを使用すると、その指示に従って作業を進めることができるので、複雑な作業を簡単に行えます。次の項では、Replication Manager のウィザードについて説明します。

- Setup Wizard
- Snapshot Group Wizard
- Deployment Template Wizard
- Offline Instantiation Wizard
- Copy Template Wizard

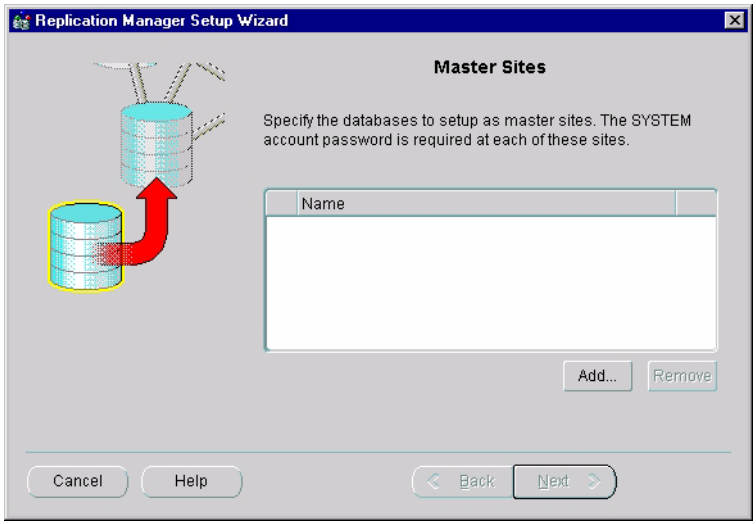
Setup Wizard

「Setup Wizard」の指示に従って操作すると、レプリケーション用のマスター・サイトおよびスナップショット・サイトを設定できます。「Setup Wizard」を使用することで、レプリケーション・サイトを簡単に準備できます。指定した各マスター・サイトごとに、このウィザードで次のステップを実行します。

- レプリケーション管理者の役割をするデータベース・アカウントの作成。デフォルトでは、このアカウントにはレプリケーション・プロパゲータおよび受信者の役割もあります。マスター・サイトのレプリケーション管理者のデフォルトのユーザー名は REPADMIN で、スナップショット・サイトのデフォルトのユーザー名は SNAPADMIN です。このユーザー名は変更可能です。
- レプリケーション管理者アカウントに対する必要な権限の付与。
- 各サイトの新規レプリケーション管理者アカウントに対応するデータベース・リンクの作成。
- マスター・サイトの場合、各マスター・サイト相互間で変更をプッシュするジョブのスケジュール。
- システム内のすべてのサイトの遅延トランザクション・キューから、完了したトランザクションをパージするジョブのスケジュール。

「Setup Wizard」は、マスター・サイト用とスナップショット・サイト用で多少異なります。マスター・サイト用の「Setup Wizard」を開くには、メニュー・バーで「Setup」>「Master Sites」の順に選択します。

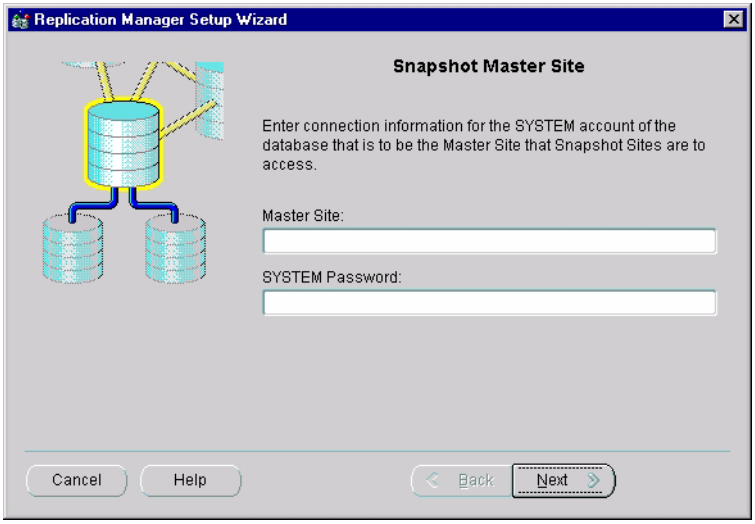
図 8-5 マスター・サイト用「Setup Wizard」の画面



関連項目：「Setup Wizard」を使用してマスター・サイトを設定する方法の詳細は、Replication Manager のオンライン・ヘルプの「Setup Master Site: Overview」のトピックを参照してください。このトピックを参照するには、「Help Contents」で「Setup Replication Sites」>「Master Site」の順に選択します。

スナップショット・サイト用の「Setup Wizard」を開くには、メニュー・バーで「Setup」>「Snapshot Sites」の順に選択します。

図 8-6 スナップショット・サイト用「Setup Wizard」の画面



関連項目：「Setup Wizard」を使用してスナップショット・サイトを設定する方法の詳細は、Replication Manager のオンライン・ヘルプの「Set Up Snapshot Site: Overview」のトピックを参照してください。このトピックを参照するには、「Help Contents」で「Setup Replication Sites」>「Snapshot Site」の順に選択します。

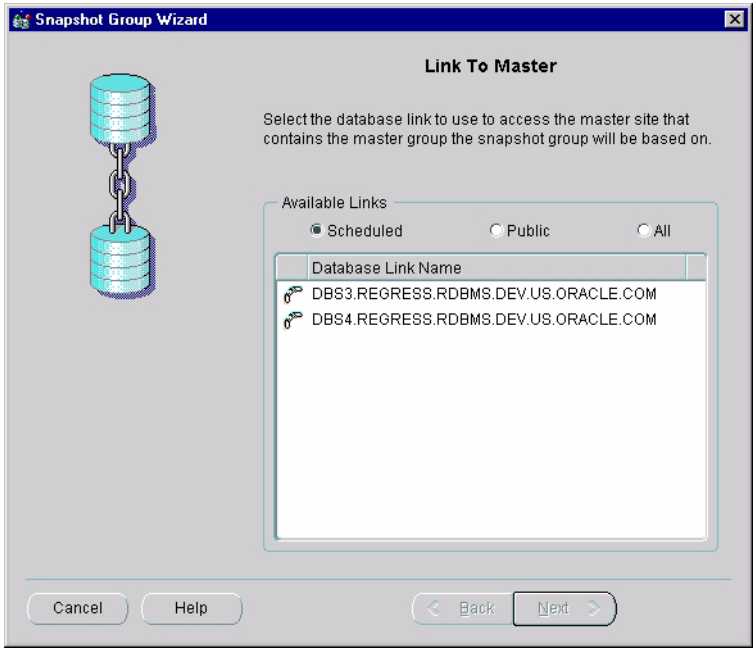
関連項目： マスター・サイトの詳細は、第 2 章「マスター・レプリケーションの概念およびアーキテクチャ」を参照してください。スナップショット・サイトの詳細は、第 3 章「スナップショットの概念およびアーキテクチャ」を参照してください。

Snapshot Group Wizard

「Snapshot Group Wizard」の指示に従って操作すると、マスター・グループをベースとするスナップショット・グループを作成できます。各スナップショットは、ソース・マスター・グループのマスター表の一部のコピーでも、完全なコピーでもかまいません。スナップショット・グループはリモート・スナップショット・サイトに置かれており、マスター・サイトの 1 つのターゲット・マスター・グループをベースとしています。

「Snapshot Group Wizard」は、スナップショット・グループを作成するスナップショット・サイトで実行します。「Snapshot Group Wizard」を開くには、Replication Manager のツール・バーの「Create Snapshot Group」ボタンをクリックします。

図 8-7 「Snapshot Group Wizard」の画面



関連項目： 「Snapshot Group Wizard」を使用してスナップショット・グループを作成する方法の詳細は、Replication Manager のオンライン・ヘルプの「Create Snapshot Group: Overview」のトピックを参照してください。このトピックを参照するには、「Help Contents」で「Setup Snapshot Replication」> 「Create」の順に選択します。

関連項目： スナップショット・グループの詳細は、3-23 ページの「スナップショット・グループ」を参照してください。

Deployment Template Wizard

配置テンプレートを使用すると、多くのリモート・スナップショット・サイトの配置およびメンテナンスの作業が簡単になります。また、配置テンプレートを使用すると、マスター・サイトでスナップショット定義の集合を定義し、その定義中のパラメータを使用してスナップショットを個々のユーザーまたはユーザーのタイプに合わせてカスタマイズできます。

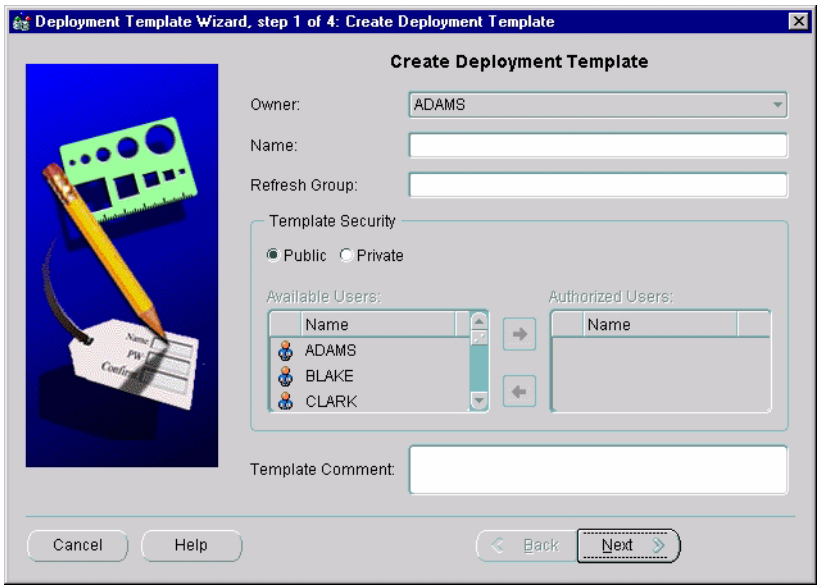
たとえば、営業用に 1 つのテンプレートを作成し、サービス技術員用に別のテンプレートを作成する場合などです。この場合、販売地域または顧客サポート・レベルなどのパラメータ値を使用できます。ユーザーがテンプレートをインスタンシエートすると、リモート・サイトで適切なスナップショットが作成され、データが移入されます。

「Deployment Template Wizard」の指示に従って操作すると、配置テンプレートを作成できます。「Deployment Template Wizard」の各画面では、次の処理を実行できます。

- 配置テンプレートの命名およびパブリックまたはプライベートの指定。プライベート・テンプレートの場合は、認証ユーザーを指定できます。
- 配置テンプレートへのオブジェクトの追加。
- 配置テンプレートのパラメータの指定。

「Deployment Template Wizard」は、配置テンプレートを作成するマスター・サイトから実行します。「Deployment Template Wizard」を開くには、Replication Manager のツール・バーの「Create Template」ボタンをクリックします。

図 8-8 「Deployment Template Wizard」の画面



関連項目： 「Deployment Template Wizard」を使用して配置テンプレートを作成する方法の詳細は、Replication Manager のオンライン・ヘルプの「Overview of Creating a Deployment Template」のトピックを参照してください。このトピックを参照するには、「Help Contents」で「Deployment Templates」>「Create」の順に選択します。

関連項目： 配置テンプレートの詳細は、第 4 章「配置テンプレートの概念およびアーキテクチャ」を参照してください。

Offline Instantiation Wizard

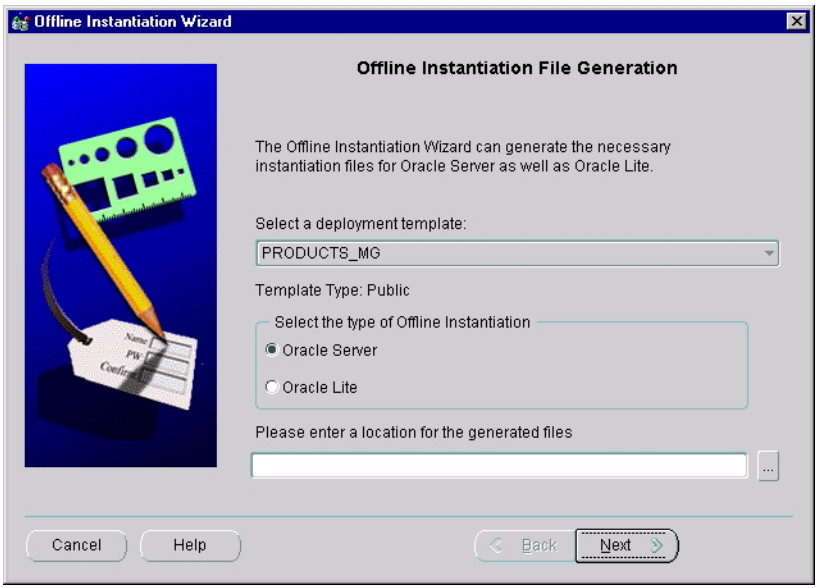
オフライン・インスタンシエーションの場合、エンド・ユーザーは、マスター・サイトにネットワーク接続しない状態で、生成されたファイルを使用してテンプレートをインスタンシエートできます。オフライン・インスタンシエーションでは、必要な DDL とデータをすべてオフライン・インスタンシエーション・ファイルに書き込み、そのファイルをスナップショット・サイトに転送して実行できます。これは、低速の WAN 接続を使用しているラップトップ・ユーザー、あるいはネットワーク接続が不安定または低速な場合に最適なソリューションです。

つまり、このソリューションは、オンライン・インスタンシエーションを困難にする低速の WAN 接続を使用するラップトップ・ユーザーに最適です。

「Offline Instantiation Wizard」の指示に従って操作すると、オフライン・インスタンシエーション用の配置テンプレートをパッケージ化できます。また、「Offline Instantiation Wizard」で生成するオフライン・インスタンシエーション・ファイルを使用して、スナップショット・サイトにスナップショットおよびその他のオブジェクトを作成できます。

「Offline Instantiation Wizard」は、オフライン・インスタンシエーション・ファイルを生成するテンプレートを含むマスター・サイトで実行します。「Offline Instantiation Wizard」を実行するには、Replication Manager のツール・バーの「Offline Instantiation File Generation」ボタンをクリックします。

図 8-9 「Offline Instantiation Wizard」の画面



注意： オンライン・インスタンシエーション・ファイルを生成する場合は、レプリケーション・マネージメント API を使用する必要があります。オンライン・インスタンシエーション・ファイルの生成の詳細は、『Oracle8*i* レプリケーション・マネージメント API リファレンス』を参照してください。

関連項目： 「Offline Instantiation Wizard」を使用してオフライン・インスタンシエーション用に配置テンプレートをパッケージ化する方法の詳細は、Replication Manager のオンライン・ヘルプの「Package for Offline Instantiation: Overview」のトピックを参照してください。このトピックを参照するには、「Help Contents」で「Deployment Templates」>「Packaging and Instantiation」の順に選択します。

関連項目： 配置テンプレートのパッケージ化およびインスタンシエーションの詳細は、4-16 ページの「パッケージ化およびインスタンシエーションのプロセス」を参照してください。

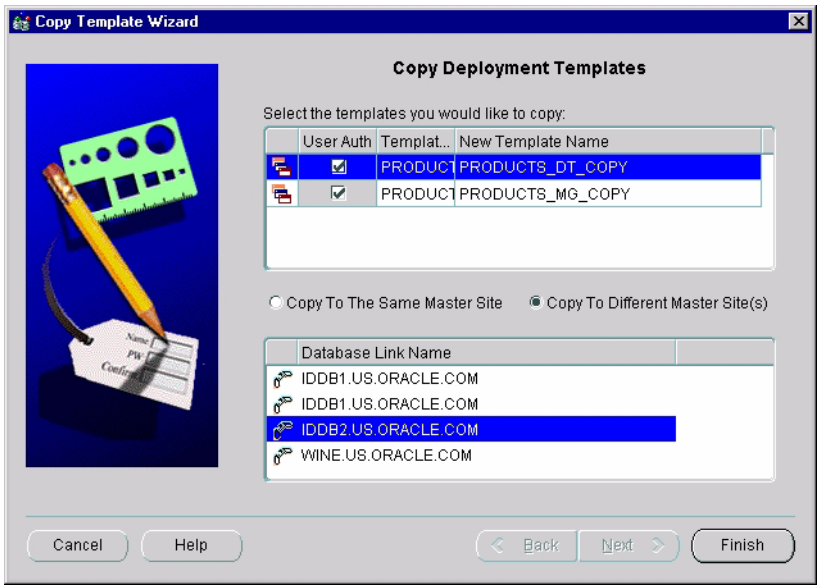
Copy Template Wizard

「Copy Template Wizard」の指示に従って操作すると、配置テンプレートをコピーできます。様々な理由から、配置テンプレートを複数のマスター・サイトにコピーすることが必要な場合があります。たとえば、次の場合です。

- ネットワーク負荷を分散させる場合： ユーザーがテンプレートをインスタンシエートする前に、ターゲット・スナップショット・サイトのマスター・サイトにテンプレートを配置する必要があります。ネットワークの規模が大きい場合、テンプレート定義を複数のマスター・サイトにコピーして、複数のスナップショット・サイトのネットワーク負荷を分散できます。
- テンプレートを変更する場合： 最初に作成したテンプレートと共通の特性を多く持つ別のテンプレートの作成が必要になることがあります。つまり、まったく新しいテンプレートを作成するのではなく、テンプレートをコピーし、必要に応じて変更する場合です。

「Copy Template Wizard」は、配置テンプレートを含むマスター・サイトから実行します。「Copy Template Wizard」を開くには、メニュー・バーで「Template」> 「Copy」の順に選択します。

図 8-10 「Copy Template Wizard」

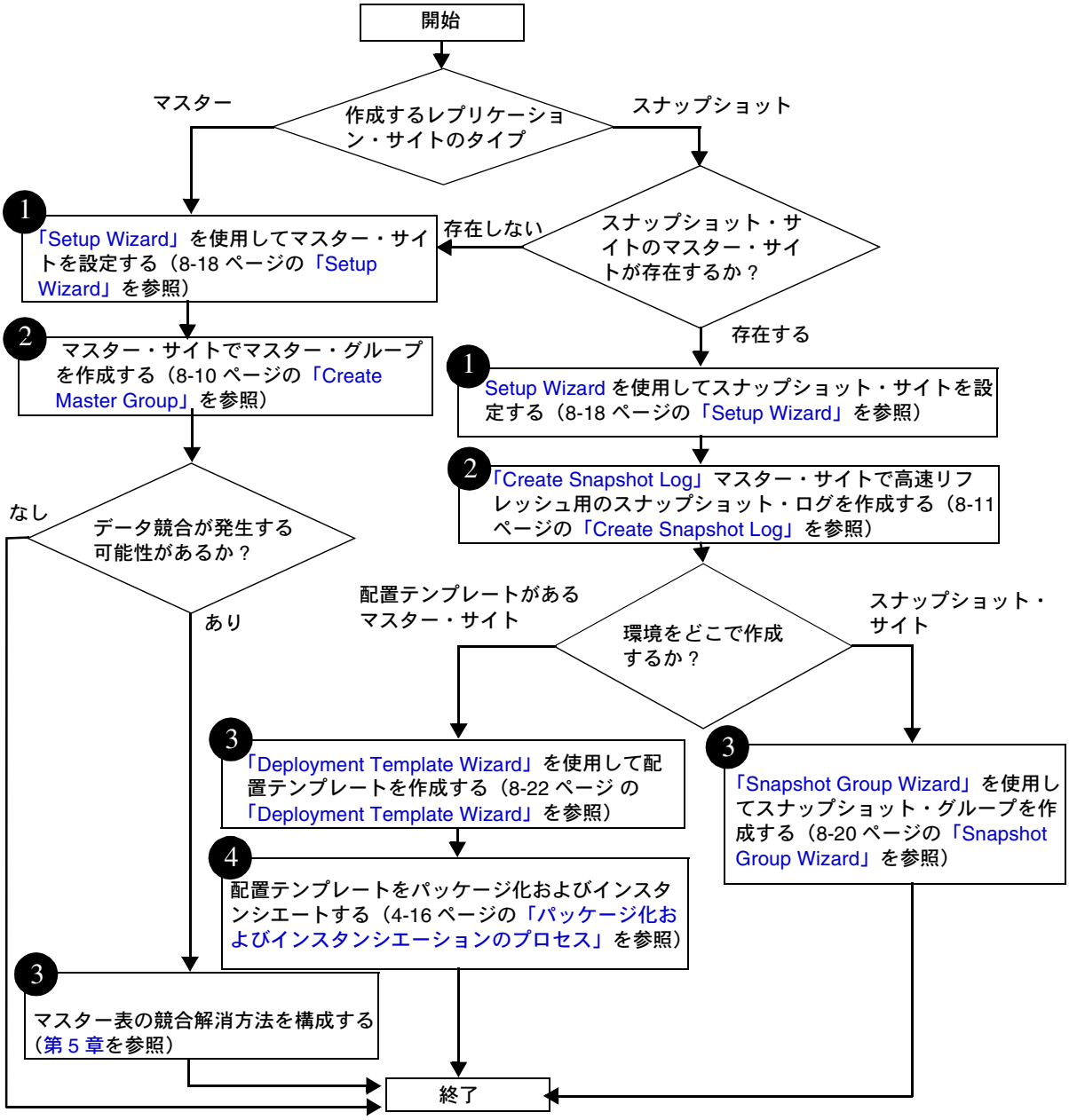


関連項目：「Copy Template Wizard」を使用して配置テンプレートをコピーする方法の詳細は、Replication Manager のオンライン・ヘルプの「Copy Deployment Template」のトピックを参照してください。このトピックを参照するには、「Help Contents」で「Deployment Templates」>「Manage」の順に選択します。

レプリケート環境を作成するためのフローチャート

図 8-11 のフローチャートは、Replication Manager を使用してレプリケーション環境を作成するときに必要な主な決定事項と作業内容を示しています。このフローチャートでは、マルチマスター環境の場合とスナップショット環境の場合に分けて表示しています。フローチャートの各ステップでは、このマニュアルでその作業の詳細を説明している項を示しています。これらの作業を完了する方法の詳細は、Replication Manager のオンライン・ヘルプを参照してください。

図 8-11 Replication Manager を使用してレプリケート環境を作成する手順



A

新機能

この付録では、Oracle8*i* の新しいレプリケーション機能を簡単に説明し、このマニュアルでさらに詳細な情報を参照できる箇所を示します。また、Oracle7 から Oracle8*i*（リリース 8.1）に移行するユーザーのために、Oracle8（リリース 8.0）から提供されている新機能も示します。

新機能には次のものがあります。

Oracle8*i*

- パフォーマンスの改善
- 大量配布サポートの改善
- セキュリティの改善
- Replication Manager
- Oracle8i Lite の統合の改善

Oracle8

- パフォーマンスの強化
- 副問合せに基づくデータのサブセット化
- システムベースのセキュリティ・モデルの強化
- Replication Manager の新機能

Oracle8*i* の新しいレプリケーション機能

以降に示す Oracle8*i*（リリース 8.1）のレプリケーション機能と拡張機能は、レプリケーションのパフォーマンスを最適化し、スナップショット環境の配布とセキュリティの効率を向上させることを目的としています。

パフォーマンスの改善

PL/SQL レプリケーション・パッケージの内部化、およびスナップショット・リフレッシュの最適化によって、レプリケーションのパフォーマンスが大幅に向上しました。

内部適用パッケージ

Oracle8 から続く取組みとして、Oracle8*i* では、より多くのレプリケーション・コードがデータベース・エンジン内に移動しました。リモート・サイトにレプリケート・トランザクションを適用するときに使用される PL/SQL 生成パッケージが内部化されました。その結果、レプリケート・トランザクションはより効率的にリモート・サイトで適用されます。また、パッケージが生成されないため、サイトをさらに短時間でインスタンシエートできます。内部パッケージの改ざんが防止されており、セキュリティも高くなっています。

関連項目： 第 2 章「マスター・レプリケーションの概念およびアーキテクチャ」

スナップショット・リフレッシュの高速化

スナップショット・リフレッシュは、大規模リフレッシュ・グループをサポートするように最適化されました。副問合せスナップショットおよび NULL リフレッシュ（最後のリフレッシュ以降マスター表に変更がない場合）のサポートが改善されました。1 つのリフレッシュ・グループは 400 スナップショットまで含むことができ、リフレッシュ・グループ内のスナップショットのリフレッシュに必要なラウンドトリップ数が削減されました。

関連項目： 第 3 章「スナップショットの概念およびアーキテクチャ」

大量配布サポートの改善

Oracle8*i* では、拡大するフロント・オフィス・アプリケーション市場、特に大量配布アプリケーションに対するレプリケーション・サポートが改善されました。

パラメータ付きスナップショット配置テンプレート

配置テンプレートを使用すると、情報の大量配布が容易になり、フィールド・サービスや営業支援システムなどのアプリケーションをサポートできます。これらのテンプレートは、あるノードでインスタンシエートされるスナップショットやその他のデータベース・オブジェクトをグループ化したものです。DBA は、これらのテンプレートを使用することで、スナップショット環境を一元的にパッケージ化し、1 つ以上のサイトに簡単かつ安全にユー

ザー定義を配布できます。目標は、環境を一度作成した後で、テンプレートを必要なときにいつでも配置できるようにすることです。

テンプレートのパラメータを使用することで、テンプレートを再定義せずにリモート・サイトでデータをサブセット化できます。また、テンプレートはパブリックまたはプライベートとして定義できます。パブリック・テンプレートはどのサイトでもインスタンシエートできますが、プライベート・テンプレートは事前定義された認証サイトでのみインスタンシエートできます。DBA は、Oracle Replication Manager の「Deployment Template Wizard」を使用して、テンプレートに追加するスキーマ・オブジェクトの選択、パラメータの選択、および認可の定義を実行できます。配置テンプレートを使用すると、Oracle8*i* Enterprise Edition、Oracle8*i* Standard Edition、Oracle8*i* Personal Edition および Oracle8*i* Lite を使用しているスナップショット・サイトにスナップショット環境を作成できます。

関連項目： 第 4 章「配置テンプレートの概念およびアーキテクチャ」

列レベルでのスナップショットのサブセット化

更新可能スナップショットを水平方向（行を選択）または垂直方向（列を選択）にサブセット化できるようになりました。前のリリースでは水平方向のサブセット化のみが可能でした。垂直パーティション化により、リモート・サイトで必要な最小限のデータを配置できるので、接続時間が短縮されました。また、対応付けられたマスター・サイトが変更されても、スナップショット・サイトは保護されます。マスター・サイトで列を追加してもスナップショット・サイトに影響しません。また、スナップショット・サイトである列を参照していない場合は、マスター・サイトでその列を削除してもスナップショット・サイトに影響しません。

関連項目：

- 3-8 ページの「スナップショットでのデータのサブセット化」
- 4-22 ページの「垂直パーティション化」
- 4-24 ページの「割当て表による水平パーティション化」

オフライン・インスタンシエーション

スナップショット配置テンプレートは、オンラインまたはオフラインでインスタンシエートできます。オンライン・インスタンシエーションの場合は、スナップショット・サイトとターゲット・マスター・サイトが接続された状態でテンプレートをインスタンシエートできます。オンライン・インスタンシエーションの利点は、データがインスタンシエーション実行時点のデータである点です。ただし、長時間にわたってアクティブな接続が必要となる可能性が高いという短所があります。

オフライン・インスタンシエーションの場合は、DBA は、配置テンプレートと必要なデータをなんらかの記憶メディア（テープ、CD-ROM など）にパッケージ化して、スナップショット・サイトに配布できます。マスター・サイトに接続する必要はなく、記憶メディアからテンプレートとデータを取り出してインスタンシエーションを実行できます。ユーザーは、オフライン・インスタンシエーションの完了後すぐに高速リフレッシュを実行できま

す。完全リフレッシュは必要ありません。オフライン・インスタンシエーションは、ネットワークに接続されていない多数のラップトップがターゲット・テンプレートをインスタンシエートする大量配布環境に理想的なソリューションです。

関連項目： 4-9 ページの「配置テンプレートのパッケージ化およびインスタンシエーション」

セキュリティの改善

スナップショット・サイトでのインスタンシエーションに使用されるスクリプトはマスター・サイトで生成され、データへのアクセスを制御できますが、受信者とプロキシ・スナップショット管理者に接続して、レプリケート・トランザクションの伝播やスナップショットのリフレッシュを行う必要があります。Oracle8*i* では、レプリケーションのセキュリティ・モデルが拡張され、信頼されないサイトへの権限の付与に関するセキュリティの問題点が排除されました。

- マスター・サイトでのリモート・プロシージャ・コール（RPC）の受信者に必要なオブジェクトの権限は自動的に管理されるようになりました。信頼されないサイトには、必要なオブジェクト権限しか付与されません。
- プロキシ・スナップショット管理者は、余分な権限が付与されていなくても、オブジェクト・グループ内のオブジェクトにアクセスできるようになりました。

関連項目： 『Oracle8*i* レプリケーション・マネージメント API リファレンス』の付録 A「セキュリティ・オプション」を参照してください。

Replication Manager

Oracle Replication Manager の機能が大幅に改善されました。主な改善点は、新しい Oracle Enterprise Manager（OEM）Java インタフェースに準拠するように作成し直されたことです。Oracle Replication Manager は、ネットワーク上のどこからでも、また Windows プラットフォーム以外でも実行できます。また、OEM にはイベントのレプリケーション・クラスがあり、たとえばエラーや遅延スナップショット・リフレッシュの監視に使用できます。

Oracle8*i* の Replication Manager には、新たに次のウィザードが追加されました。

- 「Deployment Template Wizard」。配置テンプレートを作成するときに使用します。
- 「Offline Instantiation Wizard」。オフライン・インスタンシエーション用の配置テンプレートをパッケージ化するときに使用します。

関連項目： 第 8 章「Replication Manager の概要」および Replication Manager のオンライン・ヘルプを参照してください。

Oracle8*i* Lite の統合の改善

Oracle8*i* Server と Oracle8*i* Lite の間の統合が改善され、パフォーマンスと機能性が向上しました。前述の新しいレプリケーション機能はすべてサポートされており、特にリモート・サイトでラップトップを使用する Oracle8*i* Lite ユーザーにとって、接続時間の短縮による利点は大きいものがあります。

Java RepAPI

Java RepAPI は、クライアントが Oracle Server のデータをレプリケートできるようにするランタイム・ライブラリです。Java RepAPI は、Oracle8*i* Lite クライアントやフロント・オフィス自動アプリケーションを実行するデータベースなど、シン・クライアント・データベースにレプリケーション・サービスを提供します。Oracle8*i* の配置テンプレートの新機能を基盤として、Java RepAPI では、CORBA の Internet Inter-ORB Protocol (IIOP) を通信に使用して、LOB や更新可能スナップショットの垂直サブセット化をサポートしています。Oracle8*i* は、サーバーと非接続状態での Oracle8*i* Lite サイトのインスタンシエーションをサポートしています。Java RepAPI は、Oracle リリース 8.1.6 以降のバージョンのみで使用できます。

関連項目： 詳細は、付録 C「RepAPI 用の Oracle8i Server の構成」を参照してください。

Oracle8 の新しいレプリケーション機能

以降に示す Oracle8（リリース 8.0）のレプリケーション機能と拡張機能は、マルチマスター・レプリケーションのパフォーマンスを最適化し、特定のタイプのスナップショットのサブセットを高速リフレッシュすることを目的としています。Oracle8 では、LOB のサポートが追加されました。また、Oracle8 では管理性とセキュリティも向上しました。次の項で説明する機能はすべて Oracle8*i* にも適用されます。

パフォーマンスの強化

Oracle8 では、次の新機能によってパフォーマンスが大幅に改善されました。

遅延トランザクションのパラレル伝播

データの整合性とトランザクション依存性を維持しながら、レプリケーション・トランザクション・ストリームの伝播をパラレル化することによって、スループット・パフォーマンスが大幅に向上しました。

関連項目： 7-10 ページの「スケジュール・リンクのガイドライン」

レプリケーション・トリガーの内部化

Oracle8 の古いバージョンでは外部化されていたレプリケーション・トリガーが内部化されました。内部トリガーには次の利点があります。

- 応答時間パフォーマンスの改善
- 処理オーバーヘッドの削減
- 管理作業の低減

関連項目： 第 2 章「マスター・レプリケーションの概念およびアーキテクチャ」

データ伝播の削減

Oracle8 では、ネットワークを介して伝播されるレプリケート・データの量が削減されました。伝播されるのは次の値のみです。

- 更新された列の新しい値
- 競合の検出および解消のために必要な列の古い値
- 主キー値

この機能は、LOB をレプリケートするときのパフォーマンスを改善するのに特に重要です。

関連項目： 2-38 ページの「最小限の通信」

副問合せに基づくデータのサブセット化

特定のタイプの副問合せを使用して定義されているスナップショットを高速リフレッシュできるようになりました。これにより、データのサブセットの定義とメンテナンスが容易になりました。この機能は、営業支援システムや支部の自動化などのような大量配布アプリケーションでは重要です。

関連項目： 3-8 ページの「スナップショットでのデータのサブセット化」

ラージ・オブジェクト型（LOB）のサポート

Oracle8 では、次のラージ・オブジェクト型のレプリケーションをサポートしています。

- バイナリ型の LOB（BLOB）
- 文字型の LOB（CLOB）
- 各国語サポート型の LOB（NCLOB）

関連項目： マスター・サイトへのオブジェクト定義のレプリケーションの詳細は、Replication Manager のオンライン・ヘルプを参照してください。また、3-22 ページの「スナップショットのデータ型に関する考慮事項」を参照してください。

管理方法の改善と使用方法の簡易化

Oracle8 では、次の機能によってデータベースの管理が簡単になりました。

きめ細かな停止

レプリケーション・マスター・グループは、他のレプリケーション・グループに影響を与えずに個々に停止できます。マスター・グループは、他のマスター・グループが停止している間も更新処理を継続できます。

関連項目： 2-20 ページの「レプリケーションの操作モード」

主キー・スナップショット

主キー・スナップショットを使用すると、高速リフレッシュ機能を保持しながら、マスター表を再編成できます。Oracle8 では、デフォルトで主キー・スナップショットを追加するとともに、引き続き ROWID スナップショットをサポートしています。

関連項目： 3-4 ページの「使用可能なスナップショット」を参照してください。また、Oracle 8 での ROWID スナップショットの使用方法の詳細は、『Oracle8*i* 移行ガイド』の付録 G「レプリケーション環境の移行および互換性」を参照してください。

マスター・サイトでのスナップショットの登録

Oracle8 では、スナップショットと対応付けられたマスター・サイトにスナップショットの情報が自動的に登録されます。これは、監視および分散管理に役立ちます。

> **関連項目：** 3-16 ページの「マスター・サイトでのスナップショットの登録」

高速リフレッシュ機能での表の再編成

Oracle8 には、マスター・スナップショット・ログの整合性を保持しながらマスター表を再編成できるユーティリティが用意されています。

> **関連項目：** マスター表の再編成の詳細は、『Oracle8*i* レプリケーション・マネージメント API リファレンス』を参照してください。

エクスポート / インポートを使用したオフライン・インスタンシエーションのサポート

エクスポート / インポートを使用したスキーマおよびデータベースのオフライン・インスタンシエーションがさらに自動化されました。

> **関連項目：** 『Oracle8*i* レプリケーション・マネージメント API リファレンス』の第 7 章の「エクスポート / インポートを使用したオフラインのインスタンシエーションの実行」の項を参照してください。

更新可能スナップショットの遅延制約

更新可能スナップショットで、宣言参照制約と一意性制約がサポートされるようになりました。

パーティション表と索引

Oracle8 では、パーティション表および索引のレプリケーションをサポートしています。マスター定義サイトの表と同一のパーティションをレプリケート表に作成する場合にこの機能を使用します。

システムベースのセキュリティ・モデルの強化

Oracle8 のシステムベースのセキュリティ・モデルでは、次の点が強化されました。

- 1 つのシステムレベル・モデルが同期環境でも非同期環境でも同一の動作をすることにより、整合性が向上しました。
- 受信サイトでの権限の不足によるトランザクションの失敗の可能性が低くなったため、信頼性が向上しました。

- 1 人のユーザーが REPADMIN としても REPSYS としても動作できるようになり、リンクが単純化されました。
- ユーザーレベル・モデルが必要なくなりました。
- 1 人以上のスナップショット所有者がスナップショットのリフレッシュを実行できるようになりました。

関連項目：『Oracle8*i* レプリケーション・マネージメント API リファレンス』の付録 A「セキュリティ・オプション」を参照してください。

Replication Manager の新機能

新しい Replication Manager には、システムを迅速に構成するのに役立つウィザードがいくつか用意されています。

- アカウント、スキーマおよびリンクの作成を含め、マルチマスター構成を設定するための「Setup Wizard」
- アカウント、スキーマおよびリンクの作成を含め、スナップショット・サイト構成を設定するための「Setup Wizard」
- スナップショット・グループを設定するための「Snapshot Group Wizard」

Oracle Replication Manager は、次のような Oracle8 レプリケーションの新機能の大部分をサポートしています。

- 各サイトのデータベースのバージョンの自動認識、およびリリース 8.x とリリース 7.3 の両方のデータベースを管理する動的構成
- スナップショット・ログの管理のサポート
- ジョブ、リフレッシュ・グループおよびスケジュール・リンクの日付式および間隔式の簡単な入力
- レプリケーション・サポートの再生成が必要な表を示すフラグ（Oracle8 のデータベースの場合）
- サイトのレプリケーション管理者およびプロパゲータの登録のサポート
- マスター・グループのデータベース・オブジェクトを表示するデータベース・オブジェクト・フォルダ
- すべてのマスター・サイトのマスター・グループの妥当性チェックのサポート

関連項目： 第 8 章「Replication Manager の概要」および Replication Manager のオンライン・ヘルプを参照してください。

B

レプリケーションに関する問題のトラブルシューティング

この付録では、レプリケーション環境を管理するためのトラブルシューティングのガイドライン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。

- データベース・リンクに関する問題の診断方法
- マスター・サイトに関する問題の診断方法
- 遅延トランザクション・キューに関する問題の診断方法
- スナップショットに関する問題の診断方法

データベース・リンクに関する問題の診断方法

データベース・リンクが正常に機能していないと考えられる場合、Oracle Enterprise Manager、SQL*Plus または別のツールを使用して、データベース・リンクを削除し、再作成できます。

- スケジュールした間隔が意図したとおりであることを確認します。
- スケジュールした間隔が、必要な実行時間よりも短くないことを確認します。

指定サイトへのデータベース・リンクで接続修飾子を使用した場合は、そのサイトにリンクする他のサイトでもまったく同じ接続修飾子を使用する必要があります。たとえば、次のようにデータベース・リンクを作成するとします。

```
CREATE DATABASE LINK dbs1.world@myethernet CONNECT TO repadmin
  IDENTIFIED BY secret USING 'connect_string_myethernet'
```

マスター・サイトかスナップショット・サイトかを問わず、dbs1.world@myethernet と対応付けられたすべてのサイトで、接続修飾子 myethernet を使用する必要があります。

マスター・サイトに関する問題の診断方法

マルチマスター・レプリケーション・システムでは、様々な問題が発生します。以降の項では、これらの問題とトラブルシューティングの方法をいくつか説明します。

新しいマスター・サイトでレプリケート・オブジェクトが作成されない場合

マスター・グループに新しいマスター・サイトを追加しても、新しいマスター・サイトで適切なオブジェクトが作成されない場合は、次の処理を実行します。

- 必要なプライベート・データベース・リンクが新しいマスター・サイトと既存のマスター・サイトの間に存在することを確認します。Replication Manager の「Setup Wizard」を使用してサイトを設定した場合は、何も問題はありません。リンクは、既存の各サイトから新しいサイトへ、および新しいサイトから既存の各サイトへ相互に存在している必要があります。
- すべてのサイトで管理要求が正常に終了していることを確認します。管理要求がまだ実行されていない場合、保留になっている管理要求を手動で実行し、即時に操作を完了させることができます。

DDL による変更がマスター・サイトに伝播されない場合

新しいマスター・グループ・オブジェクトを作成しても、あるいはマスター定義サイトでマスター・グループ・オブジェクトの定義を変更しても、変更がマスター・サイトに伝播されない場合は、すべてのサイトで管理要求が正常に完了していることを確認します。管理要求が保留になっている場合は、手動で実行して即時に操作を完了させることができます。

レプリケーション API を通して DDL 文を実行すると、DDL を送信するユーザー名において DDL 文が実行されます。発行者のスキーマ以外のスキーマ内のオブジェクトに DDL 文を適用するときは、発行者に文を実行する適切な権限が必要です。また、文の中でスキーマを明示的に指定する必要があります。たとえば、レプリケーション管理者のユーザーが DBMS_REPCAT.CREATE_MASTER_ REPOBJECT プロシージャの DDL_TEXT パラメータとして次のように記述するとします。

```
CREATE TABLE scott.new_emp AS SELECT * FROM hr.emp WHERE ...;
```

それぞれの表名にはスキーマ名が含まれるので、レプリケーション管理者が SCOTT や HR または別のユーザーであっても、必要な権限があれば、この文は正しく機能します。

注意： 各スキーマ・オブジェクトの名前を適切なスキーマで修飾してください。

DML による変更が他のサイトに非同期で伝播されない場合

マスター・サイトのデータを更新し、その変更がレプリケート環境の他のサイトに非同期で正しく伝播されない場合は、次の処理を実行します。

- Replication Manager を使用して、対応する遅延トランザクションが接続先にプッシュされているかどうかをチェックします。プッシュされていない場合、スケジュール・リンクがキューを接続先サイトにプッシュするまでの待ち時間をチェックできます。リンクによるスケジュール・プッシュを待機できない場合、遅延トランザクションを手動で実行できます。
- スケジュール・リンクのプッシュ間隔が経過しても対応する遅延トランザクションがプッシュされない場合は、リンクの対応するジョブをチェックします。
- 遅延トランザクションが接続先に伝播されても、エラーのために実行できない場合もあります。

DML をレプリケート表に適用できない場合

レプリケート表を修正するときに DEFERRED_RPC_QUIESCE 例外を受け取った場合、ローカル・サイトにある 1 つ以上のマスター・グループは休止または停止になっています。先に進むには、レプリケーション管理者がグループのレプリケーション・アクティビティを再開する必要があります。

バルク更新および制約違反

1 つの UPDATE 文をレプリケート表に適用すると、ゼロ以上の行を更新できます。UPDATE 文を実行すると、ゼロ以上の更新要求が、更新される行ごとに 1 つずつ遅延実行用のキューに入れられます。Oracle では文末で制約のチェックをまとめて行うので、制約が存在する場合はこの機能が重要になります。たとえば、バルク更新では一意性制約違反が発生しなくても、同じ順序で 1 行ずつ更新すると一意性制約違反が発生する場合があります。

更新順序が重要である場合は、適切な順序で 1 行ずつ更新します。これによって、遅延 RPC キューにおける更新要求の順序を定義できます。

レプリケート・オブジェクトの再作成

パッケージ、プロシージャまたはビューなどのオブジェクトをマスター・グループに追加する場合、オブジェクトの状態は VALID になっている必要があります。オブジェクトの状態が INVALID の場合は、マスター・グループにオブジェクトを追加する前に、オブジェクトを再コンパイルするか、またはオブジェクトを削除して再作成してください。

表のレプリケーション・サポートを生成できない場合

表のレプリケーション・サポートを生成すると、ローカル・サイトで内部トリガーがアクティブになります。レプリケーションに関連する大部分のパッケージ（DBMS_REPCAT や DBMS_DEFER など）に対する EXECUTE 権限を、レプリケート・オブジェクトを所有するレプリケーション管理者およびユーザーに付与する必要があります。レプリケーション管理者が多くの一般的なレプリケーション・シナリオで必要とする権限は、Replication Manager の「Setup Wizard」および DBMS_REPCAT_ADMIN パッケージを使用して付与します。ただし、レプリケート・オブジェクトの所有者がレプリケーション管理者でない場合は、DBMS_DEFER に対する EXECUTE 権限をオブジェクトの所有者に対して明示的に付与する必要があります。

レプリケート・プロシージャまたはレプリケート・トリガーに関する問題

予期しない未解消の競合を発見した場合で、プロシージャ・レプリケーションと行レベルのレプリケーションが表に混在しているときは、プロシージャを注意深く検証し、レプリケート・プロシージャが競合の原因かどうかを確認します。プロシージャでの更新と行レベルの更新の実行順序による競合がかどうかを確認してください。レプリケート・プロシージャで、更新の実行前に表を EXCLUSIVE モードにロックしているかどうか（または、行レベルの更新との競合を回避するための他のメカニズムを使用しているかどうか）をチェックします。レプリケート・プロシージャの最初で行レベルのレプリケーションを使用禁止にし、最後に再び使用可能にしていることをチェックします。プロシージャの実行時に例外が発生した場合も、行レベルのレプリケーションを再び使用可能にしていることを確認します。さらに、レプリケート・プロシージャがすべてのマスター・サイトで実行されたことをチェックします。レプリケート表で定義したレプリケート・トリガーのすべてについて、同様のチェックを行う必要があります。

ON DELETE CASCADE および整合性制約に関する問題

ON DELETE CASCADE は、次の例と類似した構成のレプリケート表をサポートしていません。この例では、A、B、C の 3 つの表があると想定します。

- 表 B に、外部キー制約のために表 A のレコードに依存するレコードがあります。
- 表 C に、外部キー制約のために表 B のレコードに依存するレコードがあります。
- 表 C には、外部キー制約のために表 A のレコードに依存するレコードもあります。

このような構成で ON DELETE CASCADE を使用すると、レコードを削除した順序とは異なる順序で伝播される可能性があります。この場合、次のエラーが戻されます。

```
ORA-02292: 整合性制約 (SCOTT.FK_CB) に違反しました。子レコードがあります。
```

このようなエラーが発生した場合は、ON DELETE CASCADE を使用せず、レプリケーションを認識してこの機能を実行するトリガーを作成します。

遅延トランザクション・キューに関する問題の診断方法

サイトの遅延トランザクションが接続先にプッシュされない場合、いくつかの理由が考えられます。次の項では、考えられる原因のいくつかを説明します。

スケジュール・リンクのジョブのチェック

スケジュール・リンクを作成すると、対応するジョブがサイトのジョブ・キューに追加されます。スケジュール・リンクが定期的な間隔で遅延トランザクションをプッシュするように設定しているときに問題が検出された場合は、まずジョブ・キューに関する問題かどうかを確認する必要があります。

分散トランザクションに関する問題

シリアル伝播を使用してリモート・サイトに遅延トランザクションをプッシュする場合は、分散トランザクションが使用されるので、トランザクションは必ずリモート・サイトで正しくコミットされた後にローカル・サイトのキューから削除されます。分散トランザクション（2 フェーズ・コミット）に関する問題の診断方法は、『Oracle8*i* 分散システム』を参照してください。

データベース・リンク指定の不備

指定したリモート・サイトにトランザクションがプッシュされていない場合、トランザクションの接続先の指定方法に問題がある場合があります。スケジュール・リンクを作成するときは、完全なデータベース・リンク名を指定する必要があります。Replication Manager を使用している場合は、問題はありません。

不適切なレプリケーション・カタログ・ビュー

ビューの定義が不適切であると、遅延トランザクションが正しく動作しない場合があります。DEFCALLDEST ビューと DEFTRANDEST ビューは、CATDEFER.SQL と CATREPC.SQL とで定義が異なります。レプリケーションを使用するときは常に、CATREPC.SQL の定義を使用する必要があります。CATDEFER.SQL を（再）ロードする場合は、その後で CATREPC.SQL のビュー定義がロードされるようにしてください。

スナップショットに関する問題の診断方法

レプリケーション・システムのスナップショット・サイトでは、様々な問題が発生します。以降の項では、これらの問題とトラブルシューティング方法をいくつか説明します。

スナップショット・サイトでのレプリケート・オブジェクトの作成に関する問題

スナップショット・サイトで新しいオブジェクトの作成に失敗した場合は、次の処理を実行します。

- 更新可能スナップショットの場合、対応付けられたマスター表にスナップショット・ログがあるかどうかをチェックします。
- オブジェクトを作成する権限があるかどうかを確認します。スナップショットの場合は、マスター表およびそのスナップショット・ログに対する SELECT 権限が必要です。詳細は、3-33 ページの「権限」を参照してください。
- 既存のスナップショットをスナップショット・グループに追加する場合は、スナップショットを再作成してグループに追加します。
- 高速リフレッシュを実行する、主キー・スナップショットや副問合せスナップショットを作成する場合は、マスター表のスナップショット・ログに主キーがロギングされていることを確認します。
- 高速リフレッシュを実行する ROWID スナップショットを作成する場合は、マスター表のスナップショット・ログに ROWID がロギングされていることを確認します。
- スナップショット・ログに副問合せスナップショットに必要なフィルタ列があるかどうかをチェックします。
- 高速リフレッシュを実行するスナップショットと関係があるすべての表に、スナップショット・ログが存在することを確認します。スナップショットに副問合せが含まれる場合は、副問合せで参照する表ごとにスナップショット・ログが存在している必要があります。

リフレッシュに関する問題のトラブルシューティング

次の項では、スナップショットのリフレッシュに関するいくつかの一般的な問題について説明します。

一般的な問題

次のようないくつかの一般的な要因で、スナップショットのグループの自動リフレッシュが実行されない場合があります。

- スナップショット・データベースの SNP バックグラウンド・プロセスの不足

- ネットワークまたはサーバー障害の影響
- サーバー・シャットダウンの影響

スナップショットのリフレッシュ・グループで問題が発生したときは、前述の状況が原因かどうかを確認します。

自動リフレッシュの再試行

グループの自動リフレッシュが失敗すると、グループはリフレッシュ実行予定のままになります。次の条件に基づいて、グループの自動リフレッシュが再試行されます。

- グループのリフレッシュを初回は 1 分後、次は 2 分後、その次は 4 分後というように、失敗するたびに再試行間隔を倍にします。
- 再試行間隔はリフレッシュ間隔自体より長い時間にはなりません。
- 最高 16 回まで自動リフレッシュを再試行します。

リフレッシュ・グループのリフレッシュを 16 回再試行してもエラーが発生する場合は、グループは破損しているとみなされます。Schema Manager の「Refresh Group」プロパティ・シートの「General」ページを参照すると、いつリフレッシュ・グループが破損したかがわかります。USER_REFRESH および USER_REFRESH_CHILDREN データ・ディクショナリ・ビューの BROKEN 列を問い合せて、リフレッシュ・グループの現在の状態を知ることもできます。

スナップショットのリフレッシュ・グループの破損によるエラーは、トレース・ファイルに記録されます。リフレッシュ・グループが正常にリフレッシュされない原因となった問題を修正し、グループを手動でリフレッシュする必要があります。この結果、破損フラグがリセットされ、自動リフレッシュが実行できるようになります。

> **関連項目：** スナップショット・トレース・ファイルの名前の形式は SNP*n* で、*n* はオペレーティング・システム固有です。各システムにおける名前は、オペレーティング・システム固有の Oracle ドキュメントを参照してください。

スナップショットのリフレッシュの継続

スナップショット・グループのリフレッシュが継続的に繰り返される場合、グループのリフレッシュ間隔をチェックします。グループの自動リフレッシュ間隔は、リフレッシュを開始する前に評価されます。グループのリフレッシュ間隔がグループのすべてのスナップショットのリフレッシュに要する時間よりも短い場合、SNP バックグラウンド・プロセスで未処理ジョブのキューがチェックされるたびに、グループ・リフレッシュが繰り返し開始されることになります。

無制限に増大するスナップショット・ログ

スナップショット・ログが無制限に増大する場合は、ネットワークまたはサイトの障害が原因で、スナップショットが削除されたことをマスターが認識できなくなっていないかを

チェックします。場合によっては、スナップショット・ログを部分的にパージするか、またはスナップショットの登録を解除する必要があります。

リフレッシュに関する問題の高度なトラブルシューティング

スナップショットのリフレッシュに関する問題が発生している場合、次の処理を実行します。

- リフレッシュがスケジュールされているかどうかを判断するには、DBA_SNAPSHOTS ビューの NEXT 値をチェックします。
- リフレッシュ間隔が経過している場合は、DBA_REFRESH ビューで、スナップショットのリフレッシュが対応付けられたジョブ番号をチェックして、ジョブ・キューに関する問題の診断を行います。
- 実行中のジョブ・キュー・プロセスがあるかどうかをチェックします。JOB_QUEUE_PROCESSES 初期化パラメータをチェックし、DBA_JOBS_RUNNING ビューを問い合せ、オペレーティング・システムを使用して SNP バックグラウンド・プロセスがまだ稼動中かどうかをチェックします。
- 2 つのスナップショット間のマスター・ディテール関係を定義すると、エラーが発生することもあります。宣言参照整合性制約を使用して、マスター表にのみマスター・ディテール関係を定義する必要があり、この関係を保つため、関連のあるスナップショットは同じリフレッシュ・グループに入れる必要があります。
- スナップショットのリフレッシュが継続的に繰り返されている場合、指定したリフレッシュ間隔をチェックします。この間隔は、スナップショットがリフレッシュされる前に評価されます。指定した間隔がスナップショットのリフレッシュに要する時間より短い場合は、SNP バックグラウンド・プロセスで未解決ジョブのキューがチェックされるたびに、スナップショットがリフレッシュされます。
- スナップショットのマスター・サイトに未解消の競合が記録されている場合、REFRESH_AFTER_ERRORS を TRUE に設定しないとスナップショットをリフレッシュできません。このパラメータは、スナップショットのリフレッシュ・グループの作成または変更時に設定できます。（Replication Manager のプロパティ・シートに、対応するパラメータがあります。）
- スナップショット・ログが大きくなりすぎる場合は、『Oracle8*i* レプリケーション・マネージメント API リファレンス』でスナップショット・ログの管理方法を参照してください。
- 同じリフレッシュ・グループ内のスナップショットでは、行は 1 つのトランザクションで更新されます。このようなトランザクションは非常に大きくなる可能性があるので、大きなロールバック・セグメント（リフレッシュ中に使用するように指定したロールバック・セグメントを含む）がスナップショット・サイトで必要になります。あるいは、リフレッシュをより頻繁に実行してトランザクションのサイズを小さくする必要があります。

- Oracle エラー ORA-12004 が発生した場合は、スナップショット・ログを管理するためにマスター・サイトでロールバック・セグメントが不足したか、スナップショット・ログの情報が古い可能性があります。たとえば、スナップショット・ログがパージまたは再作成された可能性があります。
- 1 つの表の完全リフレッシュでは、内部的に TRUNCATE 機能が使用されるので、処理速度が向上し、ロールバック・セグメントの必要量が少なくなります。ただし、スナップショットのリフレッシュが完了するまで、ユーザーは一時的にスナップショット内のデータが確認できない場合があります。複数のスナップショット（リフレッシュ・グループなど）をリフレッシュするときは、TRUNCATE 機能は使用されません。
- マスター表を再編成するとき（たとえば、システム・リソースの再生などの目的で）は、マスター表を切り捨て、ROWID スナップショットの完全リフレッシュを実行する必要があります。そうしないと、スナップショットがマスター表への誤った ROWID を持つことになります。
- リフレッシュの実行中にエラー ORA-942（表またはビューが存在しないことを示すエラー）が発生した場合は、データベース・リンクをチェックし、マスター表とスナップショット・ログに対して必要な権限を持っていることを確認します。
- 高速リフレッシュが成功した後で障害が発生した場合、次の点をチェックします。
  - スナップショット・ログの切捨て、パージまたは削除が実行されたかどうか
  - スナップショット・ログに対する必要な権限を持っているかどうか
- 強制リフレッシュに非常に長い時間を要する場合は、リフレッシュ時に使用するスナップショット・ログが削除されていないかチェックします。
- BUILD DEFERRED を使用して作成したスナップショットの最初の高速リフレッシュが失敗した場合は、その他の問題をチェックする前に、前の完全リフレッシュが成功していたことを確認します。

関連項目： スナップショット・ログの管理の詳細は、『Oracle8*i* レプリケーション・マネージメント API リファレンス』を参照してください。

C

RepAPI 用の Oracle8*i* Server の構成

この付録では、RepAPI クライアントからアクセスする Oracle8*i* Server の構成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。

- RepAPI とは
- RepAPI の概念およびアーキテクチャ
- RepAPI サーバー・オブジェクトのロードおよび公開
- オフライン・インスタンシエーション・ファイルの生成
- RepAPI の管理
- クライアントのサンプル・プログラム

RepAPI とは

RepAPI は、クライアントが Oracle Server のデータをレプリケートできるようにするランタイム・ライブラリです。RepAPI は、Oracle8*i* Lite クライアントやフロント・オフィス・オートメーション・アプリケーションを使用するクライアントなどの、シン・クライアント・データベースに対してレプリケーション・サービスを提供します。RepAPI サーバー・ランタイム・ライブラリは、*ORACLE_HOME*/rdbms/jlib ディレクトリにインストールされます。Oracle8*i* Lite クライアントにインストールされている RepAPI ファイルの場所の詳細は、Oracle8*i* Lite のドキュメントを参照してください。

RepAPI は、古いバージョンでは C 言語で記述されていましたが、Oracle8*i* では完全に Java で書き直されました。Java ベースの RepAPI 機能は、C 言語で記述されていた以前のものに比べて、必要とするリソースが少なく、使用しやすくなっています。また、Oracle8*i* では、クライアント側で保持するメタデータの量を減らし、SQL によるスナップショットの問合せを完全にマスター側で行うことで、クライアントとサーバー間でのネットワーク通信のラウンドトリップを最小限に抑えています。

Oracle8*i* の新しい配置テンプレート（リフレッシュ・グループをベースとしています）を、スナップショットを操作するための唯一のメカニズムとして使用することで、クライアント・サイトの管理と構成が簡素化されています。さらに、Oracle8*i* の RepAPI 機能は、LOB や、配置テンプレートのオフライン・インスタンシエーションおよびオンライン・インスタンシエーションをサポートしています。

オフライン・インスタンシエーションは、パッケージ化とインスタンシエーションという 2 つの作業から構成されます。通常、マスター・サイトの DBA が、特定のサイトでのオフライン・インスタンシエーション用に配置テンプレートをパッケージ化します。パッケージ化によって、各クライアント・サイト用のオフライン・インスタンシエーション・ファイルが生成されます。DBA がパッケージ化を完了すると、関連するスナップショット・ログに、テンプレートにパッケージ化されたスナップショットの更新のロギングが開始されます。オフライン・インスタンシエーション・ファイルは各クライアントで一意です。また、1 つのクライアントで複数のテンプレートをインスタンシエートできます。

ファイル転送プロトコル（FTP）、コンパクト・ディスクまたはその他の配布メカニズムによってオフライン・インスタンシエーション・ファイルがクライアント・サイトで使用できる状態になった後、クライアント・サイトのユーザーは、Oracle Client Replication Tool を使用して Oracle8*i* Lite データベースにファイルをロードします。オフライン・インスタンシエーションは大規模な高速配置に非常に便利な方法で、クライアントとサーバー間のネットワーク接続を必要としません。

オンライン・インスタンシエーションは、Oracle Client Replication Tool を使用すると 1 つのステップで完了できます。Oracle Client Replication Tool では、パッケージ化とインスタンシエーションが 1 つのステップに結合されています。オンライン・インスタンシエーションを実行するには、クライアントとサーバー間をネットワークで接続する必要があります。

> **関連項目：** 配置テンプレートの詳細は、第 4 章「配置テンプレートの概念およびアーキテクチャ」を参照してください。また、Oracle Client Replication Tool の詳細は、Oracle8*i* Lite のドキュメントを参照してください。

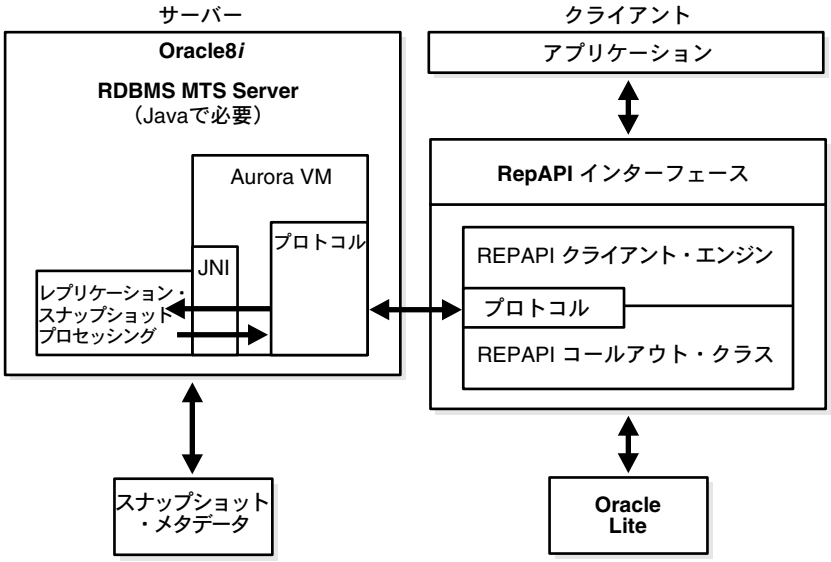
RepAPI の概念およびアーキテクチャ

RepAPI は、次の 3 つのコンポーネントから構成されています。

- Oracle8*i* の Java 仮想マシンに常駐する小規模サーバー・コンポーネント
- Oracle8*i* Lite でインプリメントされ、Java Access クラスに統合されたコールアウト・クラスのセット
- クライアント・エンジン

これらのコンポーネントはすべて Java で記述されており、クライアントおよびサーバー・コンポーネントは、Common Object Request Broker Architecture (CORBA) のスタブを介して通信を行います。Java RepAPI プロトコルは、本来はバイト・ストリームである行データを行セットにカプセル化します。各行セットは、少量の記述データを送信します。クライアントおよびサーバーによって交換された行セットはすべて、スナップショットと対応付けられます。RepAPI クライアントからマスター・サイトにデータがプッシュされると、各行セットはトランザクションにバインドされます。マスター・サイトから RepAPI クライアントにデータが取り出されると、各行セットは特定のスナップショットおよび DML 操作（INSERT、UPDATE または DELETE。これらは行セットの中で、SQL ではなくフラグによって示されます）にバインドされます。

図 C-1 RepAPI のアーキテクチャ



次の項では、RepAPI アーキテクチャの各コンポーネントの詳細を説明します。

サーバー・コンポーネント

RepAPI のサーバー・コンポーネントは、次の基本的な分散操作をサポートする CORBA インタフェースです。

- 使用可能なリフレッシュ・グループのリスト表示
- リフレッシュ・グループのインスタンシエーション用の配置テンプレートのパッケージ化
- リフレッシュ・グループの同期の開始
- リフレッシュ・グループの削除

これらの操作はすべて完全にリフレッシュ・グループ・レベルで実行されます。リフレッシュ・グループの名前は、これらの操作要求のパラメータとして送られます。個々のスナップショットに対する分散操作は、サポートしていません。リフレッシュ・グループの名前と記述は、DBA が配置テンプレートの形式で事前に設定します。リフレッシュ・グループの名前は、配置テンプレートの名前と同じです。

DBA は、Oracle8*i* で提供されている loadjava ユーティリティを使用して、サーバーの Java オブジェクトを仮想マシン（VM）にロードします。DBA は、Oracle 8*i* の公開ユーティリティを使用して、クライアントにサーバー・オブジェクトを公開します。具体的には、DBA は、oracle.repapi.server.RServer という名前の Java オブジェクトをロード、および公開して、サービス URL でアクティブにできるようにします。このオブジェクトを参照する URL の一般的な例を次に示します。

```
sess_iiop://myserver:2481:RNDM/etc/repapi
```

この URL の各要素は次のとおりです。

- myserver は、ホスト名を表します。
- 2481 は、Internet Inter-ORB Protocol（IIOP）のリスナー・ポート番号を表します。
- RNDM は、データベースの SID を表します。
- /etc/repapi は、オブジェクトの公開された名前を表します。

DBA は、このオブジェクトの公開された名前を設定できます。デフォルトでは、この名前は /etc/repapi です。エンドユーザーはアプリケーションで URL を指定するのみで、その他のクライアント設定は必要ありません。RepAPI クライアントが接続すると、Object Request Broker（ORB）によって、サーバーの一時オブジェクトがインスタンシエートされます。このオブジェクトは、スナップショット・プロセッシング・コードと対話します。セッションが切断されると、一時オブジェクトは削除されます。

サーバー・オブジェクトは、Java Native Interface（JNI）を介して Oracle8*i* Server と対話します。RepAPI のスナップショット・プロセッシングの大部分は、サーバー上で発生し、JNI のコールアウト後に実行されます。サーバーは、適切な配置テンプレートで定義されている、事前作成済みのスナップショットの問合せを実行します。サーバーにクライアントに送信する行データがあるときは、サーバーは JNI を介して VM にコールバックし、メッセージを

RepAPI クライアント・エンジンに戻します。したがって、RepAPI サーバー・コンポーネントは、CORBA の全多重メッセージ機能を十分に利用できます。

関連項目： Oracle8*i* の Java 仮想マシンの詳細は、『Oracle8*i* Java 開発者ガイド』を参照してください。

コールアウト・クラス

コールアウト・クラスは、RepAPI クライアントと Oracle8*i* Lite を結び付けるクラスです。このクラスには、クライアント・エンジンと対話できる、データベースおよび表操作の汎用セットが用意されています。コールアウト・クラスは、Oracle8*i* Lite から直接コールされるのではなく、Oracle8*i* Lite 固有の RepAPI クライアント・エンジンによって起動されます。したがって、RepAPI コールアウト・クラスによって外部データ・ストア固有の動作がクライアント・エンジンから隠され、RepAPI クライアント・エンジンは汎用性と柔軟性が高いものとなります。

クライアント・エンジン

RepAPI クライアント・エンジンは、RepAPI の基盤であり、サーバー・コンポーネントおよびコールアウト・クラスへの接続を行います。Rep API クライアント・エンジンは、次の処理を行います。

- CORBA IIOP ベースのプロトコルによるサーバーへの要求のディスパッチ
- クライアントのスナップショット・リフレッシュ・グループの管理
- 同期（リフレッシュ）操作の駆動
- マスターへの変更のプッシュ
- リフレッシュによる変更のクライアントでの適用
- アプリケーション開発用インタフェースの提供

2 つのオブジェクト（RepAPI とリフレッシュ・グループ）は、アプリケーション開発用のパブリック・メソッドを提供しています。RepAPI オブジェクトには、アプリケーション駆動でリフレッシュ・グループを操作する、次のようなパブリック・メソッドが含まれています。

- クライアントで使用可能なリフレッシュ・グループの表示
- リフレッシュ・グループの削除および作成
- オフライン・インスタンシエーション

Refresh Group オブジェクトは、パブリック同期メソッドを提供します。このメソッドには、高速、完全および強制といった標準のリフレッシュが含まれます。また、同期を使用してプッシュ専用の操作を実行できます。この操作を実行すると、クライアントからサーバーに対してのみ変更が伝播されます。

RepAPI クライアント・エンジンによって、メタデータ表の小さいセットが Oracle8*i* Lite データベースに保持されています。このメタデータには、次のものが含まれています。

- サーバー・オブジェクトの URL。次回接続を試みるときに使用します。
- 最後に正常にプッシュされたトランザクションのトランザクション ID（更新可能スナップショットが含まれていた場合）。
- リフレッシュ・グループのリスト。リフレッシュ・グループ名、識別番号およびリフレッシュ順序番号が含まれています。リフレッシュ順序番号は、2 つの側の妥当性チェックとして動作します。
- スナップショットのリスト。スナップショット名、識別番号、および更新可能かどうかを示すフラグが含まれています。

このメタデータの他に、RepAPI には、Oracle8*i* Lite クライアントの更新可能スナップショットのトランザクションのキュー表がメンテナンスされます。トランザクション・キューは、特定のマスター・サイトとバインドされ、トリガーによって更新可能スナップショットの更新が獲得されると、データ移入が実行されます。更新可能スナップショットを含まないグループに対してリフレッシュ要求が送られた場合でも、リフレッシュが開始されると、トランザクション・キューは常に、リフレッシュ・グループと対応付けられたマスター・サイトに完全にプッシュされます。 Oracle8*i* Server へキューをプッシュするとき、トランザクション境界は保持されます。

Oracle8*i* Server で RepAPI を使用する前の作業

Oracle8*i* Server で RepAPI を使用する前に、次の処理ステップを実行します。

1. 初期化パラメータ・ファイルの初期化パラメータを編集します。

   a. 次の初期化パラメータが次の値以上に設定されていることを確認します（必要に応じて値を大きくすることはできますが、この値より小さい値は設定しないでください）。

      SHARED_POOL_SIZE= 50000000

      JAVA_POOL_SIZE=50000000

      OPEN_LINKS=10

   b. 次の初期化パラメータが初期化パラメータ・ファイルに存在しない場合は、初期化パラメータを追加します。初期化パラメータが存在する場合は、次の値以上に値が設定されていることを確認します（必要に応じて値を大きくすることはできますが、この値より小さい値は設定しないでください）。

      ENQUEUE_RESOURCES=1000

      OPEN_CURSORS=100

      MAX_ENABLED_ROLES=30

   c. MTS_DISPATCHERS 初期化パラメータが次のように設定されていることを確認します。

      MTS_DISPATCHERS =
      "(PROTOCOL=TCP)(PRE=oracle.aurora.server.SGiopServer)"

2. *ORACLE_HOME*/network/admin ディレクトリの listener.ora ファイルに、次のような LISTENER 構成の行が含まれていることを確認します。ホスト名は、myserver です。

```
(DESCRIPTION =
     (PROTOCOL_STACK =
        (PRESENTATION = GIOP)
        (SESSION = RAW))
     (ADDRESS =
        (PROTOCOL = TCP)
        (HOST = myserver)
         (PORT = 2481)))
```

注意： Secure Sockets Layer（SSL）を使用している場合は、PROTOCOL を TCPS に、PORT を 2482 に設定します。2481 と 2482 は、標準のインストールのデフォルトの PORT 設定です。ただし、別の値を設定することもできます。

---

3. Oracle8*i* Server をいったんシャットダウンしてから再起動し、初期化パラメータの変更を有効にします。

RepAPI サーバー・オブジェクトのロードおよび公開

RepAPI クライアントから RepAPI のサーバー・コンポーネントにアクセスできるようにするには、loadjava ユーティリティを使用して、適切な RepAPI サーバー・オブジェクトを Oracle8*i* の仮想マシン（VM）にロードします。そして、公開ユーティリティを使用してオブジェクトを公開します。

インストール時にロードおよび公開される RepAPI サーバー・オブジェクト： Oracle8*i* の Java オプションをインストールした場合は、インストール時に RepAPI サーバー・オブジェクトが自動的にロードおよび公開されています。次に示すステップを参考にしてください。RepAPI サーバー・オブジェクトが正常にロードおよび公開されているかどうかを確認するには、C-12 ページのステップ 3 に進みます。

関連項目： この項では、RepAPI をサポートする Oracle8*i* Server の構成方法を説明します。

- Oracle8*i* Lite クライアントでの RepAPI の使用方法の詳細は、Oracle8*i* Lite のドキュメントを参照してください。
- loadjava ユーティリティおよび公開ユーティリティの詳細は、『Oracle8*i* Enterprise JavaBeans と CORBA 開発者ガイド』を参照してください。

RepAPI サーバー・オブジェクトをロードおよび公開するには、次のステップを実行します。

1. loadjava ユーティリティを実行して、`repapi_s.jar` ファイルをデータベースにロードします。

   コマンド・プロンプトから `repapi_s.jar` ファイルをロードするための構文は、次のようになります。

```
loadjava
   -synonym
   -resolve
   -thin
   -user sys/SYS_PASSWORD@HOST:LISTENER_PORT:ORACLE_SID
   -grant public
   ORACLE_HOME/rdbms/jlib/repapi_s.jar
```

この構文に含まれている変数は次のとおりです。

- *SYS_PASSWORD*
  SYS ユーザーのパスワードを指定します。
- *HOST*
  Oracle8*i* Server がインストールされているホストの名前を指定します。
- *LISTENER_PORT*
  リスナーのポートを指定します。デフォルトは 1521 です。
- *ORACLE_SID*
  Oracle8*i* Server に指定されたシステム識別子（SID）を指定します。
- *ORACLE_HOME*
  Oracle8*i* Server がインストールされているディレクトリを指定します。

注意： この loadjava 構文は、UNIX システムでも Windows プラットフォームでも使用できます。適切なファイル・パス指定に変更してください。

たとえば、次のような環境を想定します。

- *SYS_PASSWORD* は change_on_install
- *HOST* は myserver
- *LISTENER_PORT* は 1521
- *ORACLE_SID* は DB1
- *ORACLE_HOME* は `/oracle/product/8.1` ディレクトリ

この環境での、正しい loadjava ユーティリティ構文は次のとおりです。

```
loadjava -synonym -resolve -thin
   -user sys/change_on_install@myserver:1521:DB1
   -grant public /oracle/product/8.1/rdbms/jlib/repapi_s.jar
```

2. 公開ユーティリティを実行し、サービス URL を使用して RepAPI オブジェクトを公開します。

```
publish
   -u system
   -password SYSTEM_PASSWORD
   -s sess_iiop://HOST:ORB_PORT:ORACLE_SID/PUBLISHED_OBJECT
   oracle.repapi.server.RServer
   oracle.repapi.serverstub.RSHelper
   -schema SYS
   -grant public
```

この構文に含まれている変数は次のとおりです。

- *SYSTEM_PASSWORD*
  SYSTEM ユーザーのパスワードを指定します。
- *HOST*
  Oracle8*i* Server がインストールされているホストの名前を指定します。
- *ORB_PORT*
  IIOP のリスナー・ポート番号を指定します。デフォルトでは、TCP 接続の場合は 2481 に、Secure Sockets Layer（SSL）を使用した TCP 接続の場合は 2482 に設定されています。
- *ORACLE_SID*
  Oracle8*i* Server に指定されたシステム識別子（SID）を指定します。
- *PUBLISHED_OBJECT*
  公開オブジェクトの名前を指定します（デフォルトは、/etc/repapi です）。

注意： この公開構文は、UNIX システムでも Windows プラットフォームでも使用できます。適切なファイル・パス指定に変更してください。

たとえば、以下のような環境を想定します。

- SYSTEM_PASSWORD は manager
- *ORB_PORT* は 2481
- *HOST* は myserver
- *ORACLE_SID* は DB1
- *PUBLISHED_OBJECT* は /etc/repapi

この環境での、正しい公開ユーティリティ構文は次のとおりです。

```
publish -u system -password manager
   -s sess_iiop://myserver:2481:DB1/etc/repapi
   oracle.repapi.server.RServer
   oracle.repapi.serverstub.RSHelper
   -schema SYS -grant public
```

3. 次のステップを実行して、オブジェクトが正常に公開されたことを確認します。

   a. 次のコマンドを入力して、公開オブジェクトを表示するセッションを作成します。

```
sess_sh
   -user system
   -password SYSTEM_PASSWORD
   -service sess_iiop://HOST:AURORA_ORB_PORT:ORACLE_SID
```

この構文に含まれている変数は次のとおりです。

- *SYSTEM_PASSWORD*
  SYSTEM ユーザーのパスワードを指定します。
- *HOST*
  Oracle8*i* Server がインストールされているホストの名前を指定します。
- *AURORA_ORB_PORT*
  Java オブジェクトを公開したときに指定されたポート番号を指定します。デフォルトでは、TCP 接続の場合は 2481 に、SSL を使用した TCP 接続の場合は 2482 に設定されています。Aurora は、Oracle8*i* Server の Java 仮想マシンです。
- *ORACLE_SID*
  Oracle8*i* Server に指定されたシステム識別子（SID）を指定します。

たとえば、次のような環境を想定します。

- *SYSTEM_PASSWORD* は manager
- *HOST* は myserver
- *AURORA_ORB_PORT* は 2481
- *ORACLE_SID* は DB1

この環境での、正しい公開ユーティリティ構文は次のとおりです。

```
sess_sh -user system -password manager
   -service sess_iiop://myserver:2481:DB1
```

b. シェルが表示されるまで待機します。プロンプトが使用可能になると、画面に次のように表示されます。

```
AURORA ORB  shell
$
```

注意： プロンプトにこのように表示されるまで、操作を続行しないでください。これには、数分かかることもあります。

---

c. 次のように入力して、公開オブジェクト etc/repapi に対する許可をチェックします。

```
cd test
ls -l
```

Read、Write および Execute に対する許可が行の先頭に表示されます。次のように、Read 列と Execute 列の下に PUBLIC と表示されている必要があります。

```
Read     Write    Exec
PUBLIC   SYSTEM   PUBLIC
```

Read 列と Execute 列に PUBLIC と表示されていない場合は、次のコマンドを実行します。

```
chmod +r public repapi
chmod +e public repapi
```

許可を再度チェックし、repapi に対する READ および EXECUTE 許可が PUBLIC に付与されたことを確認します。

```
ls -l
```

d. 次のように入力して、セッションを終了します。

```
exit
```

オフライン・インスタンシエーション・ファイルの生成

RepAPI クライアント用のオフライン・インスタンシエーション・ファイルは、レプリケーション・マネージメント API の DBMS_REPCAT_INSTANTIATE.INSTANTIATE_OFFLINE_REPAPI ファンクションを使用して生成します。

オフライン・インスタンシエーション・ファイルの命名規則は、次のとおりです。

user_templatename_siteid.oli

したがって、ユーザー名が SCOTT、テンプレート名が MYTEMPLATE、サイト識別番号が 1234 の場合、ファイル名は次のようになります。

scott_mytemplate_1234.oli

サイト識別番号は、ファンクションを実行するときに SITE_ID パラメータを使用して指定できます。サイト識別番号を指定しない場合は、番号が自動的に生成されます。サイト識別番号は、一時プレース・ホルダです。Oracle8*i* Lite では、この値は Oracle8*i* Lite のグローバルに一意なサイト識別子と置き換えられます。

ユーザーは、次の例のように、オフライン・インスタンシエーション・ファイルを生成できます。

```
DECLARE
  i number;
BEGIN
  i := dbms_repcat_instantiate.instantiate_offline_repapi(
       refresh_template_name => 'PRODUCTS_DT',
       trace_vector => 0);
END;
/
```

デフォルトでは、オフライン・インスタンシエーション・ファイルはサーバーの *ORACLE_HOME*/dbs ディレクトリに生成されます。また、配置テンプレートにテンプレート・パラメータが含まれている場合は、接続しているユーザーのパラメータ値が適用されます。

DBA 権限を持つユーザーは、次の例のように、ユーザー名とオフライン・インスタンシエーション・ファイルのパスを指定できます。

```
DECLARE
  i number;
BEGIN
  i := dbms_repcat_rgt.instantiate_offline_repapi(
       refresh_template_name => 'PRODUCTS_DT',
       user_name => 'SCOTT',
       offline_dirpath => '/mydirectory',
       trace_vector => 0);
END;
/
```

この例では、配置テンプレートの名前は PRODUCTS_DT、オフライン・インスタンシエーション・ファイルのディレクトリ・パスは `/mydirectory`、デバッグのトレース・レベルは 0（ゼロ）です。また、配置テンプレートにテンプレートパラメータが含まれている場合は指定したユーザーのパラメータ値が適用されるので、この例では、ユーザー SCOTT のパラメータ値が適用されます。

関連項目： DBMS_REPCAT_INSTANTIATE.INSTANTIATE_OFFLINE_REPAPI ファンクションの使用方法の詳細は、『Oracle8*i* レプリケーション・マネージメント API リファレンス』を参照してください。

注意： RepAPI クライアント用のオフライン・インスタンシエーション・ファイルは、Replication Manager を使用して生成することもできます。詳細は、Replication Manager のオンライン・ヘルプを参照してください。

RepAPI の管理

RepAPI の管理および監視には、次のツールを使用できます。

Replication Manager

Replication Manager を使用すると、配置テンプレートおよび RepAPI クライアント・サイト用のオフライン・インスタンシエーション・ファイルを作成できます。また、ネットワークを介してクライアント・サイトを使用できるときは、Replication Manager を使用して、クライアント・サイトのスナップショットを監視し、必要に応じて、リフレッシュできます。

関連項目： Replication Manager の使用方法の詳細は、Replication Manager のオンライン・ヘルプを参照してください。

レプリケーション・マネージメント API

レプリケーション・マネージメント API には、次の PL/SQL ファンクションおよびプロシージャが含まれています。

- DBMS_REPCAT_INSTANTIATE.INSTANTIATE_OFFLINE_REPAPI
- DBMS_REPCAT_RGT.INSTANTIATE_OFFLINE_REPAPI
- DBMS_REPCAT_RGT.DROP_SITE_INSTANTIATION

関連項目： PL/SQL ファンクションおよびプロシージャを使用して RepAPI クライアント・サイトを管理する方法の詳細は、『Oracle8*i* レプリケーション・マネージメント API リファレンス』を参照してください。

レプリケーション・カタログ

DBA_REPCAT_TEMPLATE_SITES スタティック・データ・ディクショナリ・ビューを使用すると、RepAPI クライアント・サイトでのテンプレートのインスタンシエーションの状態を監視できます。

関連項目： DBA_REPCAT_TEMPLATE_SITES スタティック・データ・ディクショナリ・ビューの詳細は、『Oracle8*i* レプリケーション・マネージメント API リファレンス』を参照してください。

Oracle Client Replication Tool

Oracle Client Replication Tool は、グラフィカル・ユーザー・インタフェースを使用した Java アプリケーションで、Oracle8*i* Lite とともにインストールされます。ユーザーは、Oracle Client Replication Tool を使用して配置テンプレートをインスタンシエートできます。また、アプリケーション開発者は、Oracle Client Replication Tool をサンプル・コードとして使用してカスタム・アプリケーションを作成できます。

さらに、Oracle Server のインストール完了時にクライアントの Jar ファイルが組み込まれます。Jar ファイルは、サーバーの *ORACLE_HOME*/rdbms/jlib ディレクトリに置かれています。

関連項目： Oracle Client Replication Tool の使用方法の詳細は、Oracle8*i* Lite のドキュメントを参照してください。

クライアントのサンプル・プログラム

RepAPI の Java サンプル・プログラムが、Oracle8*i* Server 配布プログラムに含まれています。このサンプル・プログラムを使用すると、Oracle8*i* Lite のクライアント環境をテストできます。また、Oracle8*i* Lite クライアント環境で RepAPI を使用してアプリケーションを作成するための統合テクニックを学べるようになっています。

サンプル・プログラムのインストール

サンプル・プログラムをインストールするには、次のステップを実行します。

1. *ORACLE_HOME*/rdbms/demo ディレクトリの RepSample.java ファイルを任意のディレクトリにコピーします。
2. ステップ 1 でファイルをコピーしたディレクトリが、CLASSPATH 環境変数に含まれていることを確認します。
3. コマンド・プロンプトで次のコマンドを入力して、RepSample.java をコンパイルします。

   ```
   javac RepSample.java
   ```

   このコマンドを実行すると、RepSample.class という名前のクラス・ファイルが作成されます。

サンプル・プログラムの使用方法

サンプル・プログラムでは、Oracle の Java RepAPI レプリケーション・ライブラリを使用して、Oracle8*i* Server と Oracle8*i* Lite クライアント間でデータをレプリケートします。Oracle8*i* Lite サイトのユーザー名とパスワードは、SYSTEM/MANAGER と想定します。ユーザー名とパスワードが異なる場合は、サンプル・プログラムを適切に編集してください。サンプル・コードでは最低で 9 つの引数を使用する必要があります。引数は正しい順序で指定する必要があります。

サンプル・プログラムでは、オフライン（マスター・サイトに接続されていない状態）とオンライン（マスター・サイトに接続されている状態）の両方のモードで作業を実行できるようになっています。オフライン・モードでは、特定の配置テンプレート（リフレッシュ・グループ）、ユーザーおよびサイトに関するメタデータと初期データを含む既存のファイル（拡張子が *.oli のファイル）のインスタンシエーションのみを実行できます。オンライ

ン・モードでは、マスター・サイトとの接続中にリフレッシュ・グループを作成し、リフレッシュ・グループとマスター・サイトを同期化できます。

サンプル・プログラムの完全な構文を次に示します。

```
java RepSample ( -offline | -online )
   -user USER_NAME
   -pass USER_PASSWORD
   -master MASTER_DATABASE_NAME
   -dbid ORACLE_LITE_DATABASE
   [-url SERVICE_URL]
   [-offlineinstantiate OFFLINE_FILE_NAME]
   [-create TEMPLATE_NAME]
   [-sync | syncf TEMPLATE_NAME]
   [-drop TEMPLATE_NAME]
   [-list]
```

この構文に含まれている変数は次のとおりです。

- *USER_NAME*
  Oracle8*i* Server の認証ユーザーの識別子を指定します。
- *USER_PASSWORD*
  Oracle8*i* Server の *USER_NAME* のパスワードを指定します。
- *MASTER_DATABASE_NAME*
  Oracle8*i* Lite クライアントに格納されている、Oracle8*i* Server のラベルを指定します。このラベルは、個々の Oracle8*i* Server のトランザクション・キュー表の命名に使用します。

  注意： *MASTER_DATABASE_NAME* の値には、ハイフンやダッシュなどの特殊文字は使用できません。これは、Oracle8*i* Lite の表のネーミング制限です。

- *ORACLE_LITE_DATABASE*
  Oracle8*i* Lite データベースの完全修飾されたファイル名を指定します。
- *SERVICE_URL*
  次の形式で指定します。

  sess_iiop://*HOST:AURORA_ORB_PORT:ORACLE_SID/PUB_OBJECT*

この URL の各要素は次のとおりです。

- *HOST*
  Oracle8*i* Server がインストールされているホストの名前を指定します。
- *AURORA_ORB_PORT*
  Java オブジェクトを公開したときに指定されたポート番号を指定します。デフォルトでは、TCP 接続の場合は 2481 に、SSL を使用した TCP 接続の場合は 2482 に設定されています。
- *ORACLE_SID*
  Oracle8*i* Server に指定されたシステム識別子（SID）を指定します。
- *PUB_OBJECT*
  RepAPI サーバー構成の一部として公開コマンドで設定された公開オブジェクト名を指定します。デフォルトでは、/etc/repapi が公開オブジェクト名です。この名前は DBA が設定できます。
- *OFFLINE_FILE_NAME*
  オフライン・インスタンシエーション・ファイルの完全修飾されたファイル名を指定します。このファイルは、DBA が Replication Manager または DBMS_REPCAT_INSTANTIATE.INSTANTIATE_OFFLINE_REPAPI ファンクションを使用してマスター・サイトで生成したファイルで、シリアライズ化されています。*OFFLINE_FILE_NAME* には、次のファイル型の識別子が付いています。

  `.oli`

- *TEMPLATE_NAME*
  操作を行う配置テンプレート（およびリフレッシュ・グループ）の名前を指定します。配置テンプレートとリフレッシュ・グループの名前は同じである必要があります。エンドユーザーは、これらの名前を設定できません。配置テンプレート名は、DBA が指定します。

使用例

ここでは、サンプル・プログラムの使用例を示します。

例 1 以前にオフラインでインスタンシエートしたテンプレートからデータをロードするには、次のコマンドを実行します。

```
java RepSample -offline -user scott -pass tiger -master sales
   -dbid c:¥orant¥oldb40¥polite.odb -offlineinstantiate SCOTT_T1_CLNT2.oli
```

例 2 T2 という名前の配置テンプレートをオンラインでインスタンシエートするには、次のコマンドを実行します。

```
java RepSample -online -user scott -pass tiger -master sales
   -dbid c:¥orant¥oldb40¥polite.odb
   -url sess_iiop://salespc.us.oracle.com:2481:db1/etc/repapi
   -create T2
```

例 3 例 2 で作成したグループ T2 の完全リフレッシュを実行するには、次のコマンドを実行します。

```
java RepSample -online -user scott -pass tiger -master sales
   -dbid c:¥orant¥oldb40¥polite.odb
   -url sess_iiop://salespc.us.oracle.com:2481:db1/etc/repapi
   -sync T2
```

例 4 例 2 で作成したグループ T2 の強制リフレッシュを実行するには、次のコマンドを実行します。この場合は、sync パラメータではなく syncf パラメータを使用します。

```
java RepSample -online -user scott -pass tiger -master sales
   -dbid c:¥orant¥oldb40¥polite.odb
   -url sess_iiop://salespc.us.oracle.com:2481:db1/etc/repapi
   -syncf T2
```

例 5 T3 とうい名前の配置テンプレートをオンラインでインスタンシエートし、例 2 で作成したリフレッシュ・グループ T2 と同期化させるには、次のコマンドを実行します。

```
java RepSample -online -user scott -pass tiger -master sales
   -dbid c:¥orant¥oldb40¥polite.odb
   -url sess_iiop://salespc.us.oracle.com:2481:db1/etc/repapi
   -create T3  -sync T2
```

例 6 例 2 で作成したリフレッシュ・グループ T2 をオンラインで削除するには、次のコマンドを実行します。

```
java RepSample -online -user scott -pass tiger -master sales
   -dbid c:¥orant¥oldb40¥polite.odb
   -url sess_iiop://salespc.us.oracle.com:2481:db1/etc/repapi
   -drop T2
```

例 7 使用可能でかつインスタンシエートされているリフレッシュ・グループのリストを表示するには、次のコマンドを実行します。

```
java RepSample -online -user scott -pass tiger -master sales
   -dbid c:¥orant¥oldb40¥polite.odb
   -url sess_iiop://salespc.us.oracle.com:2481:db1/etc/repapi
   -list
```

索引

A

AND 式
　単純副問合せスナップショット，3-15

B

BLOB のサポート，A-7

C

CLOB のサポート，A-7
COMPATIBLE 初期化パラメータ，7-3
Copy Template Wizard，8-26

D

DBA_REGISTERED_SNAPSHOTS ビュー，3-16
DBA_REPCATLOG ビュー，2-22
DBA_SNAPSHOT_LOGS ビュー，3-16
DBMS_REFRESH パッケージ
　REFRESH_CHANGE プロシージャ，4-21
DBMS_REPCAT_INSTANTIATE パッケージ
　INSTANTIATE_OFFLINE_REPAPI ファンクション，C-14
DBMS_REPCAT パッケージ，2-22，2-24
　COMPARE_OLD_VALUES プロシージャ
　　競合解消，5-27
　DO_DEFERRED_REPCAT_ADMIN プロシージャ，2-22，2-23
　REPCAT_IMPORT_CHECK プロシージャ，6-11
　SEND_OLD_VALUES プロシージャ
　　競合解消，5-27
DBMS_REPUTIL パッケージ
　REPLICATION_OFF プロシージャ，6-4，6-18
　REPLICATION_ON プロシージャ，6-4，6-19
DBMS_SNAPSHOT パッケージ
　REGISTER_SNAPSHOT プロシージャ，3-17
　UNREGISTER_SNAPSHOT プロシージャ，3-17
Deployment Template Wizard，8-22

E

ENQUEUE_RESOURCES 初期化パラメータ，7-3
EXIST 句
　単純副問合せスナップショット，3-15

G

GLOBAL_NAMES 初期化パラメータ，7-3

I

INIT.ORA パラメータ，「初期化パラメータ」を参照

J

Java RepAPI，A-5，C-1
　CORBA，C-3
　Java オブジェクト，C-4
　Oracle Client Replication Tool，C-16
　アーキテクチャ，C-3
　概念，C-3
　管理，C-16
　クライアント・エンジン，C-5
　コールアウト・クラス，C-5
　サーバー・オブジェクト
　　公開，C-9
　　公開の確認，C-12
　　ロード，C-9

サーバー・コンポーネント，C-4
サーバーで構成する前，C-7
サーバーの構成，C-1
サービス URL，C-4
サンプル・プログラム，C-17
インストール，C-17
使用方法，C-17
例，C-20
事前構成作業，C-7
初期化パラメータ，C-7
定義済，C-2
配置テンプレート
パッケージ化，C-14
リスナー構成，C-8
JAVA_POOL_SIZE 初期化パラメータ，7-4
JOB_QUEUE_INTERVAL 初期化パラメータ，2-23，7-4
JOB_QUEUE_PROCESSES 初期化パラメータ，2-23，3-34，7-4

L

LOB
サポート，3-22，A-7
レプリケート表で許可されているデータ型，7-2

M

MAX_ENABLED_ROLES 初期化パラメータ，7-5
MTS_DISPATCHERS 初期化パラメータ，7-5

N

NCLOB のサポート，A-7
Net8
FAILOVER オプション，6-10
NULL リフレッシュ，A-2
n-way レプリケーション，「マルチマスター・レプリケーション」を参照，1-5

O

Offline Instantiation Wizard，8-24
OPEN_CURSORS 初期化パラメータ，7-6
OPEN_LINKS 初期化パラメータ，2-15，7-6
OPS，「Oracle Parallel Server」を参照
Oracle Client Replication Tool，C-16
Oracle Parallel Server
レプリケーションとの比較，2-5，6-8
Oracle Replication Manager，1-13，8-1
「Destination Map」，8-7
インタフェース，8-4
ウィザード，8-17
Copy Template，8-26
Deployment Template，8-22
Offline Instantiation Wizard，8-24
Setup，8-18
Snapshot Group，8-20
起動，8-3
最初のログイン，8-3
ツールバー，8-10
「Change Database Connection」ボタン，8-10
「Create Local Job」ボタン，8-12
「Create Master Group」ボタン，8-10
「Create Refresh Group」ボタン，8-11
「Create Scheduled Link」ボタン，8-12
「Create Snapshot Group」ボタン，8-11
「Create Snapshot Log」ボタン，8-11
「Create Snapshot」ボタン，8-11
「Create Template」ボタン，8-12
「Help」ボタン，8-13
「Offline Instantiation File Generation」ボタン，8-12
「Refresh」ボタン，8-10
「Setup Wizard」ボタン，8-10
ナビゲータ・シート，8-5
右ペイン，8-5
メニュー，8-13
「Edit」，8-14
「Group」，8-14
「Help」，8-17
「Object」，8-16
「Scheduling」，8-16
「Setup」，8-17
「Template」，8-15
「View」，8-14
「File」，8-13
用途，8-2
Oracle8*i*
新しいレプリケーション機能，A-1
Oracle8*i* Lite のサポート，C-1

P

Parallel Server，「Oracle Parallel Server」を参照，2-5
PARALLEL_MAX_SERVERS 初期化パラメータ，2-37，7-6
PARALLEL_MIN_SERVERS 初期化パラメータ，2-37，7-7
PCTFREE
　複合スナップショットの値，3-28
PCTUSED
　複合スナップショットの値，3-28
peer-to-peer レプリケーション，「マルチマスター・レプリケーション」を参照
PRIMARY KEY 制約
　単純副問合せスナップショット，3-15
PROCESSES 初期化パラメータ，7-7

R

RepAPI，「Java RepAPI」を参照
Replication Manager，「Oracle Replication Manager」を参照
REPLICATION_DEPENDENCY_TRACKING 初期化パラメータ，7-7
ROWID データ型
　ROWID スナップショット，3-5

S

Setup Wizard，8-18
SHARED_POOL_SIZE 初期化パラメータ，7-7
Snapshot Group Wizard，8-20
SNP バックグラウンド・プロセス，3-34

U

UNIQUE 制約
　単純副問合せスナップショット，3-15
USER_REFRESH_CHILDREN ビュー，B-8
USER_REFRESH ビュー，B-8

あ

アカウント
　スナップショット用の作成，3-32

い

依存性
　順序付け
　　レプリケート・トランザクション，6-21
　追跡
　　パラレル伝播，6-21
インポート
　スナップショット，3-20
　スナップショット・ログ，3-20

う

上書きによる競合解消方法，5-13

え

エラー
　エラー・キュー，2-20
　　競合，5-11

お

オブジェクト
　レプリケート
　　再作成，B-4
オフライン・インスタンシエーション
　サポート，A-8

か

加算による競合解消方法，5-15
完全リフレッシュ，3-28
管理キュー，2-24
管理要求，2-22
　状態，2-25
　　エラー，2-25
　　コールバック待機中，2-25
　　コールバック中，2-25
　　準備中，2-25

き

きめ細かな停止，A-7
休止，2-20
行
　競合検出時の識別，2-42

競合
一意性，5-3
回避，5-8
エラー・キュー，5-11
回避，5-7
動的所有権，6-11
検出，2-33，2-41，5-5
行の識別，2-42，5-6
更新，5-3
回避，5-8
削除，5-4
回避，5-8
順序
回避，5-8
トークン渡し，6-12
プロシージャ・レプリケーション，1-17
ワークフロー，6-11
競合解消，2-42，5-1
一意性競合，5-3
一意性競合の解消方法，5-24
上書きによる方法，5-13
概念，5-2
加算による方法，5-15
競合の回避，5-7
競合の検出，5-5
競合のタイプ，5-3
更新競合，5-3
更新競合の解消方法，5-11
最小値による方法，5-20
最大値による方法，5-18
サイト名の追加による方法，5-24
サイト優先順位による方法，5-22
削除競合，5-4
削除競合の解消方法，5-26
順序の追加による方法，5-25
データ要件，5-2
同期伝播，2-33
動的サイト所有権，5-7
トランザクションの順序，5-4
廃棄による方法，5-17，5-25
パフォーマンス，5-26
最小限の通信，5-26
古い値の比較，5-27
古い値のプッシュ，5-27
プライマリ・サイト所有権，5-7
プロシージャ・レプリケーション，6-3
平均による方法，5-16
最も新しいタイムスタンプによる方法，5-12
優先グループによる方法，5-21
列グループ，5-10
レプリケーション，2-10
強制リフレッシュ，3-29

く

グループ所有者
スナップショット・グループ，3-24

け

結合
単純副問合せスナップショット，3-15

こ

交差部表，3-15
更新可能スナップショット，1-8
高速リフレッシュ，3-28
ダイレクト・パス・ロード，3-29
表の再編成，A-8

さ

再使用可能にする
レプリケーション，6-19
最小限の通信，2-38
競合解消，5-26
最小値による競合解消方法，5-20
最大値による競合解消方法，5-18
サイト所有権
動的，5-7
プライマリ，5-7
サイト名の追加による競合解消方法，5-24
サイト優先順位
タイムスタンプによる競合解消時のバックアップ方式としての，5-12，5-18
サイト優先順位による競合解消方法，5-22
索引
パーティション表，A-8
レプリケーション，2-16
参照整合性
自己参照型制約，2-16

し

システムベースのセキュリティ，A-8
シノニム，2-19
　レプリケーション，2-19
支部自動化，A-6
シャドウ列グループ，5-10
主キー
　レプリケート表，7-2
主キー・スナップショット，3-5，A-7
受信者
　レプリケーション，2-12
順序
　レプリケーション，2-18
順序の追加による競合解消方法，5-25
使用禁止
　レプリケーション，6-17
初期化パラメータ
　JOB_QUEUE_INTERVAL，2-23
　JOB_QUEUE_PROCESSES，2-23
　OPEN_LINKS，2-15
　PARALLEL_MAX_SERVERS，2-37
　PARALLEL_MIN_SERVERS，2-37
　編集，3-36
　レプリケーション，7-3
ジョブ
　スケジュール・リンクのチェック，B-6
ジョブ・キュー，2-20
シリアライズ化
　トランザクション，6-3
新機能
　LOB のサポート，A-7
　Oracle Replication Manager の拡張機能，A-4
　Oracle8*i* Lite の統合の改善，A-5
　管理および使用，A-7
　スナップショットの副問合せ，A-6
　セキュリティの改善，A-4
　セキュリティの強化，A-8
　大量配布サポートの改善，A-2
　データのサブセット化と副問合せ，A-6
　パフォーマンスの強化，A-2，A-5
　副問合せスナップショット，A-6
　レプリケーション，A-1

す

スキーマ
　スナップショット用の作成，3-32
スケジュール・パージ
　ガイドライン，7-11
　定期的なパージ，7-12
スケジュール・リンク
　ガイドライン，7-10
　シリアル伝播，7-12
　定期的なプッシュ，7-10
　パラレル伝播，7-12
　連続的なプッシュ，7-11，7-12
スナップショット，1-7，3-1
　NULL リフレッシュ，A-2
　Oracle8*i* Lite のサポート，C-1
　ROWID，3-5
　アーキテクチャ，3-17
　インポート，3-20
　概念，3-2
　権限，3-33
　更新可能，1-8
　　遅延制約，A-8
　索引，3-23
　作成，3-38
　サポートされているデータ型，3-22
　主キー，3-5，A-7
　準備，3-31
　使用，3-3
　垂直パーティション化，3-8
　水平パーティション化，3-8
　スキーマの作成，3-32
　スナップショット・グループ，3-23
　スナップショット・ログ，1-10，3-20
　大規模リフレッシュ・グループのための最適化，A-2
　タイプ，3-4
　大量配布，3-4
　単純副問合せ
　　AND 式，3-15
　　EXISTS 句，3-15
　　結合，3-15
　　フィルタ列要件，3-15
　　マスター表の列数，3-15
　データのサブセット化，3-4，3-8
　トラブルシューティング，B-7
　トレース・ファイル，B-8

ネットワーク負荷，3-3
配置テンプレート，1-11，4-1
ビュー，3-22
複合，3-6
PCTFREE の値，3-28
PCTUSED の値，3-28
副問合せ，3-9
制限事項，3-14
ベース表，3-21
マスター・サイト，3-19
マスター・サイトでの登録，3-16
マスター表，3-19
モバイル・コンピューティング，3-4
読取り専用，1-7，3-8
登録，3-17
登録解除，3-17
副問合せを含む単純なスナップショット，A-6
リフレッシュ，1-10，3-28
開始，3-30
間隔，3-30
完全，3-28
強制，3-29
高速，3-28
再試行，B-8
最終リフレッシュ時刻の問合せ，3-16
障害，B-8
トラブルシューティング，B-7，B-8
必要に応じた，3-30
リフレッシュ・グループ，3-25
サイズ，3-27
列レベルでのサブセット化，A-3
レプリケーションを使用可能にする，6-19
レプリケーションを使用禁止にする，6-19
割当て表，3-11
スナップショット・グループ，1-5，3-23
グループ所有者，3-24
配置テンプレート，4-21
スナップショット・サイト
スキーマ，7-15
スケジュール・パージ
ガイドライン，7-11
スケジュール・リンク
ガイドライン，7-10
設定，7-14
遅延トランザクション・キュー
スケジュール・プッシュ，3-34
データベースのバージョン，7-14
データベース・リンク，3-32，7-15
ネットワーク接続性，7-14
配置テンプレートの準備，7-13
マスター・サイトとの比較，7-8
利点，7-9
ローカルでの作成，4-28
ロールバック・セグメント，7-16
スナップショット・ログ，1-10，3-20
ROWID，3-20
インポート，3-20
基礎となる表，3-20
作成，3-37
作成に必要な権限，3-37
主キー，3-20
トラブルシューティング，B-8
配置テンプレート，4-13

せ

制限事項
プロシージャ・レプリケーション，2-16
生成
レプリケーション・サポート
プロシージャ・レプリケーション，6-4
制約
参照
自己参照型，2-16
制約違反，B-4
セキュリティ，A-8
接続修飾子，2-14
問題の診断方法，B-2

た

耐障害性，6-7
Oracle Parallel Server，6-8
実現，6-9
設計上の考慮事項，6-8
大量配布，3-4，4-2，A-6
多対 1 参照
単純副問合せスナップショット，3-15
多対多参照
単純副問合せスナップショット，3-15
単純副問合せスナップショット
多対 1 参照，3-15
多対多参照，3-15

ち

遅延制約および更新可能スナップショット，A-8
遅延トランザクション，2-19
　問題の診断方法，B-6
遅延トランザクション・キュー
　スケジュール・パージ，3-34
　スケジュール・プッシュ，3-34
　プッシュ，2-20
蓄積交換レプリケーション，「非同期レプリケーション」を参照

て

定期的なパージ
　スケジューリング，7-12
定期的なプッシュ
　スケジューリング，7-10
停止
　きめ細かな停止，A-7
データ
　整合性
　　シリアル伝播，6-21
　　パラレル伝播，6-21
　　複数の列グループでの保証，5-10
データ型
　サポート，A-7
　スナップショットで使用可能なデータ型，3-22
　レプリケート表で許可されているデータ型，7-2
データ・ディクショナリ
　レプリケーション，1-14
データ伝播
　依存性の維持，6-21
　削減，A-6
　同期，2-31
データのサブセット化，A-6
データベース・リンク
　指定の不備，B-6
　スナップショット・サイト，3-32，7-15
　接続修飾子，2-14
　問題の診断方法，B-2
　レプリケーション，2-12
デッドロック
　解消
　　同期伝播，2-33
伝播，2-29
　開始，2-35
　削減，A-6
　パラレル，2-36
　　実現，2-37
　　チューニング，2-38
　プロパゲータのセキュリティ・コンテキスト，2-32
　モード，2-33

と

同期レプリケーション，1-16，2-31
　トランザクションの接続先，2-33
動的サイト
　所有権，5-7
動的所有権
　競合回避，6-11
　行の所有者の検索，6-15
　所有権の取得，6-15
　ワークフロー・パーティション分割，6-11
トークン渡し，6-12
　実現例，6-11
トラブルシューティング，B-1
トランザクション
　シリアライズ化，6-3
　伝播
　　保護メカニズム，6-20
トリガー
　内部化，A-6
　レプリケート，2-17，6-19
トレース・ファイル
　スナップショット，B-8

な

内部トリガー，A-6

は

パージ
　定期的
　　スケジューリング，7-12
パーティション表
　索引，A-8
廃棄による競合解消方法，5-17，5-25
配置テンプレート，1-11，4-1
　アーキテクチャ，4-14
　インスタンシエーション，1-11，4-9
　　オフライン，4-10，4-18，A-3

オンライン，4-9，4-17
シナリオ，4-11
プロセス，4-16
オブジェクト定義，4-6，4-14
概念，4-3
垂直パーティション化，4-22
水平パーティション化，4-24
スナップショット・グループ，4-21
スナップショット・サイト，4-9
スナップショット・サイトの準備，7-13
スナップショット・ログ，4-13
スナップショットの追加，4-14
設計，4-22
定義，4-14
データ・セット，4-26
テンプレートの一般情報，4-4
テンプレート・パラメータ，4-7
セキュリティ，4-8
パッケージ化，4-9
オフライン・インスタンシエーション（Oracle8*i* Lite）用，C-14
オフライン・インスタンシエーション用，4-16
オンライン・インスタンシエーション用，4-16
プロシージャ，4-13
プロセス，4-16
パラメータ，A-2
ユーザー認証，4-8
要素，4-4
リフレッシュ・グループ，4-21
列のサブセット化，4-22
バックアップ
レプリケーション用，6-10
パッケージ
内部化，A-2
レプリケーション，2-16
パフォーマンス
レプリケーション，2-36
パフォーマンスの強化，A-2，A-5
パラレル・サーバー・プロセス
レプリケーション環境用の構成，2-36
パラレル伝播，2-36，A-5
依存性
追跡，6-21
計画，7-12
実現，2-37
チューニング，2-38
レプリケーション環境，7-12
バルク更新，B-4

ひ

非同期レプリケーション，2-29
ビュー，2-19
レプリケーション，2-19
表
交差部，3-15
再編成，A-8
パーティション表と索引，A-8
変更
変更をレプリケートしない，6-17
レプリケーション，2-15
レプリケーション・サポートの生成に関する問題，B-4
表記
規則，xiii

ふ

ファンクション
レプリケート，2-17
フィルタ列，3-37
単純副問合せスナップショット，3-15
フェイルオーバー・サイト
FAILOVER オプション，6-10
実現，6-9
複合スナップショット，3-6
PCTFREE の値，3-28
PCTUSED の値，3-28
副問合せスナップショット，A-6
AND 式，3-15
EXIST 句，3-15
結合，3-15
多対 1 参照，3-15
多対多参照，3-15
フィルタ列要件，3-15
マスター表の列数，3-15
プッシュ
定期的
スケジューリング，7-10
連続的
スケジューリング，7-11，7-12
プライマリ・サイト
所有権，5-7

プロシージャ
　レプリケート，2-17
プロシージャ・レプリケーション，1-16
　競合，6-3
　競合の検出，1-17
　サポートの生成，6-4
　使用方法，6-2
　制限事項，2-16，6-2
　トランザクションのシリアライズ化，6-3
　ラッパー，1-16
プロパゲータ
　レプリケーション，2-12
分散スキーマ管理，1-14
分散トランザクション
　問題，B-6

へ

平均による競合解消方法，5-16
変更
　表
　　変更をレプリケートしない，6-17

ま

マスター・グループ，1-5，2-26
マスター・サイト，1-5
　DDL による変更が伝播されない場合，B-3
　DML による変更が伝播されない場合，B-3
　新しいマスター・サイトでレプリケート・オブジェクトが作成されない場合，B-2
　きめ細かな停止，A-7
　スケジュール・パージ，3-34
　　ガイドライン，7-11
　スケジュール・リンク
　　ガイドライン，7-10
　スナップショット，3-19
　スナップショット・サイトとの比較，7-8
　スナップショットの登録，3-16
　制約
　　違反，B-4
　内部トリガー，3-19
　バルク更新，B-4
　問題の診断方法，B-2
　ユーザー，2-11
　利点，7-9
　レプリケーション，2-6，2-11
　ロール，2-11
マスター定義サイト，1-5，2-2
マスター表
　スナップショット，3-19
　スナップショット・ログ，3-20
　列
　　単純副問合せスナップショットに関する数値制限，3-15
マルチマスター・レプリケーション，1-5，2-1
　アーキテクチャ，2-11
　概念，2-2
　使用，2-3
　同期，2-7，2-8
　トランザクション伝播保護，6-20
　非接続スナップショット，2-5
　非同期，2-7
　フェイルオーバー，2-3
　ロード・バランス，2-4

も

最も新しいタイムスタンプ
　競合解消方法，5-12

ゆ

優先グループによる競合解消方法，5-21

よ

読取り専用スナップショット，1-7，3-8
　登録
　　手動，3-17
　登録解除，3-17

ら

ラッパー
　プロシージャ・レプリケーション，1-16

り

リカバリ
　レプリケーション用，6-10
リフレッシュ，3-28
　DBMS_REFRESH パッケージ
　　REFRESH_CHANGE プロシージャ，4-21

NULL 最適化，3-27
開始，3-30
間隔，3-30
パラメータ制約，3-30
完全，3-28
強制，3-29
グループ，3-30
高速，3-28
高速リフレッシュおよび表の再編成，A-8
再試行，B-8
自動，3-30
手動，3-30
障害，B-8
スナップショット，1-10
必要に応じた，3-30
リフレッシュ・グループ，3-25
サイズに関する考慮事項，3-27
大規模リフレッシュ・グループのための最適化，A-2
トラブルシューティング，B-7
配置テンプレート，4-21

れ

列
列グループ，2-28，5-10
シャドウ，5-10
複数の列グループでのデータ整合性の保証，5-10
列レベルでのスナップショットのサブセット化，A-3
レプリケーション
Java RepAPI，C-1
Oracle Parallel Server との比較，2-5，6-8
Oracle8*i* Lite のサポート，C-1
REPADMIN ユーザー，2-12
SNP バックグラウンド・プロセス，3-34
アプリケーションでの使用，1-3
インポートされたデータのチェック，6-11
エラー・キュー，2-20
オブジェクト，2-15
オフまたはオンによるカレント・セッションへの影響，6-18
概要，1-2
可用性，1-2，6-7
環境の作成，8-28
環境の作成のためのフローチャート，8-28
管理，1-12，2-20
管理キュー，2-24
管理要求，2-22
状態，2-25
競合
検出，2-41
プロシージャ・レプリケーション，1-17
競合解消，1-15，2-10，2-42，5-1
グループ，1-4，2-26
計画，7-1
高度なテクニック，6-1
再開，2-22
再使用可能にする，6-19
最小限の通信，2-38
サイト，1-4
サポートされていないデータ型
BFILE，3-22
LONG，3-22
受信者，2-12
使用可能にする，6-4，6-17
使用禁止，6-4，6-17
使用禁止にする，6-18
使用方法，1-2
初期化パラメータ，7-3
ジョブ・キュー，2-20
新機能，A-1
Oracle Replication Manager の拡張機能，A-4
Oracle8*i* Lite の統合の改善，A-5
セキュリティの改善，A-4
大量配布サポートの改善，A-2
パフォーマンスの強化，A-2，A-5
シングル・マスター，2-6
スナップショット，1-7，3-1
スナップショット・グループ，1-5
接続修飾子，2-14
耐障害性，6-7
大量配布，1-3，4-2
遅延トランザクション，2-19
遅延トランザクション・キュー，2-20
問題の診断方法，B-6
中断，2-21
定義，1-2
停止，2-20
データベース・リンク，2-12
CONNECT TO 句，2-13
USING 句，2-13
データ要件，5-2
伝播，2-29

同期，1-16，2-31
トラブルシューティング，B-1
トランザクション伝播保護，6-20
トリガー，6-19
内部トリガー，2-19
内部プロシージャ，2-19
ネットワーク負荷の軽減，1-2
配置テンプレート，1-11，4-1
ハイブリッド構成，1-11
バックアップおよびリカバリ，6-10
パフォーマンス，1-2，2-36
非同期伝播，2-29
フィルタ列，3-37
フェイルオーバー，6-9
プロシージャ，1-16
プロシージャ・レプリケーション，6-2
プロパゲータ，2-12
分散スキーマ管理，1-14
マスター・グループ，1-5，2-26
マスター・サイト，1-5，2-6，2-11
マスター定義サイト，1-5
マルチマスター，1-5，2-1
モード，2-20
モバイル・コンピューティング，1-2
リアルタイム・レプリケーション，「同期レプリケーション」を参照
列グループ，2-28
レプリケーション管理者，2-12
レプリケーション・マネージメント API，1-14
レプリケーション，
「スナップショット」も参照
「マルチマスター・レプリケーション」も参照
レプリケーション・オブジェクト，1-4，2-19
再作成，B-4
索引，2-16
順序，2-18
スナップショット・サイト
作成に関する問題，B-7
トリガー，2-17
パッケージ，2-16
表，2-15，7-2
サポートを生成できない場合，B-4
主キー，7-2
変更，6-17
プロシージャ，2-17
レプリケーション・カタログ，1-14
DBA_REGISTERED_SNAPSHOTS，3-16
DBA_REPCATLOG，2-22
DBA_SNAPSHOT_LOGS，3-16
USER_REFRESH，B-8
USER_REFRESH_CHILDREN，B-8
不適切なビュー，B-6
レプリケーション・トリガー，A-6
レプリケーション・ファンクション，2-17
レプリケーション・マネージメント API，1-14，2-22
レプリケーションを使用可能にする，6-17
レプリケート・トランザクション
依存性の順序，6-21
レプリケート表
DML の非互換性，B-4
許可されているデータ型，7-2
連続的なプッシュ
スケジューリング，7-11，7-12

ろ

ロールバック・セグメント
スナップショット・サイト，7-16

わ

ワークフロー，6-11
割当て表，3-11