# Oracle® Database

# 異なるエンディアン・オペレーティング・システムで新しいリリースのための新しいハードウェアへの非 CDB の移行

# 19c

F16175-02(原本部品番号:F10900-02)

2019年5月

# タイトルおよび著作権情報

Oracle Database異なるエンディアン・オペレーティング・システムで新しいリリースのための新しいハードウェアへの非CDBの移行, 19c

F16175-02



原本著者: Sunil Surabhi、Nirmal Kumar

原本協力著者: Lance Ashdown、Padmaja Potineni、Rajesh Bhatiya、Prakash Jashnani、Douglas Williams、Mark Bauer

原本協力者: Roy Swonger、Byron Motta、Hector Vieyra Farfan、Carol Tagliaferri、Mike Dietrich、Marcus Doeringer、Umesh Aswathnarayana Rao、Rae Burns、Subrahmanyam Kodavaluru、Cindy Lim、Amar Mbaye、Akash Pathak、Thomas Zhang、Zhihai Zhang

このソフトウェアまたはハードウェア、そしてドキュメントは、第三者のコンテンツ、製品、サービスへのアクセス、あるいはそれらに関する情報を提供することがあります。お客様との間に適切な契約が定められている場合を除いて、オラクル社およびその関連会社は、第三者のコンテンツ、製品、サービスに関して一切の責任を負わず、いかなる保証もいたしません。お客様との間に適切な契約が定められている場合を除いて、オラクル社およびその関連会社は、第三者のコンテンツ、製品、サービスへのアクセスまたは使用によって損失、費用、あるいは損害が発生しても一切の責任を負いかねます。

# 目次

# はじめに

このガイドでは、特定のユースケース・シナリオの完了に役立てるために集められた、Oracle Databaseユーザー支援ドキュメントの各トピックについて説明します。

- このドキュメントのユースケース・シナリオ
- ドキュメントのアクセシビリティについて

## このドキュメントのユースケース・シナリオ

このシナリオ・ドキュメントを使用すると、RMANおよびトランスポータブル表領域を使用して別のエンディアン・オペレーティング・システムにOracle Databaseを移行できます。

このシナリオの前提条件

- ソース・データベースとターゲット・データベースのキャラクタ・セットが同一であること、または少なくとも宛先データベースのキャラクタ・セットがソース・データベースのスーパーセットであることを確認します。
- トランスポート・プロセスが完了するまで、トランスポートする表領域を読取り専用モードにします。
- 宛先サーバーでデータ・ポンプ・インポートを実行する前に、RMAN CONVERTコマンドを実行してデータを変換します。
- oracleユーザーとして移行プロシージャを実行します。

このシナリオの概要

- **トランスポータブル表領域を使用したデータの移行**。
- **ターゲット・サーバーの準備**。データベースを移行する準備として、ターゲット・データベースを作成します。
- **ターゲット・サーバーのendian形式へのデータの変換**。
- **SQL*Plusを使用したデータベースへの接続**。
- **Oracle Databaseの移行**。データベースを移行し、移行後タスクを実行します。

これらのステップは、このドキュメントの各章に対応しています。

**親トピック:** はじめに

## ドキュメントのアクセシビリティについて

Oracleのアクセシビリティについての詳細情報は、Oracle Accessibility ProgramのWebサイト(http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc)を参照してください。

Oracleサポートへのアクセス

サポートを購入したオラクル社のお客様は、My Oracle Supportを介して電子的なサポートにアクセスできます。詳細情報は(http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info)か、聴覚に障害のあるお客様は(http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs)を参照してください。

**親トピック:** はじめに

# 1 トランスポータブル表領域を使用したデータの移行

トランスポータブル表領域およびトランスポータブル表では、ユーザー定義表領域に存在するデータがトランスポートされます。

トランスポータブル表領域機能を使用して、表領域セットをデータベース間で移動できます。

- プラットフォーム間でのデータ・トランスポート
  プラットフォーム間でデータをトランスポートできます。
- データのトランスポートに関する一般的な制限事項
  データのトランスポートに関する一般的な制限事項があります。また、フル・トランスポータブル・エクスポート/インポート、トランスポータブル表領域またはトランスポータブル表に固有の制限事項もあります。
- データのトランスポートの互換性に関する注意事項
  データをトランスポートする場合、Oracle Databaseはターゲット・データベースが稼働する最低限の互換性レベルを計算します。
- トランスポータブル表領域に関する制限事項
  この項では、トランスポータブル表領域の制限事項を示します。

# プラットフォーム間でのデータ・トランスポート

プラットフォーム間でデータをトランスポートできます。

プラットフォーム間でデータをトランスポートする機能を使用すると、次のことが可能です。

- データベースをプラットフォーム間で移行できます。
- コンテンツ・プロバイダは、簡単にかつ効率的に構造化データを公開し、別のプラットフォームでOracle Databaseを実行している顧客に配布できます。
- データ・ウェアハウス環境からデータ・マート(多くの場合、小規模プラットフォームで実行されている)へのデータの配布を簡素化できます。
- 異なるオペレーティング・システムまたはプラットフォーム上のOracle Databaseインストール間で読取り専用表領域を共有できます。この場合、次の各項で説明するように、それらのプラットフォームおよびendiannessが同じプラットフォームからストレージ・システムにアクセスできることが前提となります。

多くのプラットフォーム(すべてではありません)では、クロス・プラットフォームでのデータのトランスポートがサポートされます。V$TRANSPORTABLE_PLATFORMビューを問い合せると、サポートされているプラットフォームを参照して、各プラットフォームのendian形式(バイトの並び順)を確認できます。次の問合せを実行すると、プラットフォーム間でのデータのトランスポートがサポートされているプラットフォームが表示されます。

```
COLUMN PLATFORM_NAME FORMAT A40
COLUMN ENDIAN_FORMAT A14

SELECT PLATFORM_ID, PLATFORM_NAME, ENDIAN_FORMAT
  FROM V$TRANSPORTABLE_PLATFORM
  ORDER BY PLATFORM_ID;

PLATFORM_ID PLATFORM_NAME                            ENDIAN_FORMAT
----------- ---------------------------------------- --------------
          1 Solaris[tm] OE (32-bit)                  Big
          2 Solaris[tm] OE (64-bit)                  Big
          3 HP-UX (64-bit)                           Big
          4 HP-UX IA (64-bit)                        Big
          5 HP Tru64 UNIX                            Little
          6 AIX-Based Systems (64-bit)               Big
          7 Microsoft Windows IA (32-bit)            Little
          8 Microsoft Windows IA (64-bit)            Little
          9 IBM zSeries Based Linux                  Big
         10 Linux IA (32-bit)                        Little
         11 Linux IA (64-bit)                        Little
         12 Microsoft Windows x86 64-bit             Little
         13 Linux x86 64-bit                         Little
         15 HP Open VMS                              Little
         16 Apple Mac OS                             Big
         17 Solaris Operating System (x86)           Little
         18 IBM Power Based Linux                    Big
         19 HP IA Open VMS                           Little
         20 Solaris Operating System (x86-64)        Little
         21 Apple Mac OS (x86-64)                    Little
```

ソース・プラットフォームとターゲット・プラットフォームが同じendiannessの場合、データはデータ変換なしでソース・プラットフォームからターゲット・プラットフォームにトランスポートされます。

ソース・プラットフォームとターゲット・プラットフォームでendiannessが異なる場合は、トランスポートするデータをターゲット・プラッ

トフォームの形式に変換する必要があります。次のいずれかの方法使用してデータを変換できます。

- DBMS_FILE_TRANSFERパッケージのGET_FILEまたはPUT_FILEプロシージャ

  これらのプロシージャのいずれかを使用して、ソース・プラットフォームとターゲット・プラットフォーム間でデータファイルを移動する場合、各データファイル内の各ブロックがターゲット・プラットフォームのendiannessに変換されます。変換はターゲット・プラットフォームで発生します。

- RMAN CONVERTコマンド

  ソース・プラットフォームまたはターゲット・プラットフォームでRMAN CONVERTコマンドを実行します。このコマンドにより、トランスポートするデータがターゲット・プラットフォームの形式に変換されます。

注意:

UNDO セグメントを含むデータファイルでは、異なる endian 形式間でのデータファイルの変換はサポートされていません。

データファイルのデータを別のプラットフォームにトランスポートする前に、そのデータが属しているプラットフォームをデータファイル・ヘッダーで識別する必要があります異なるプラットフォームのOracle Databaseインストール間で読取り専用表領域をトランスポートするには、少なくとも1回データファイルを読取り/書込みにします。

**関連項目:**

「プラットフォーム間でのデータの変換」

**親トピック:** トランスポータブル表領域を使用したデータの移行

# データのトランスポートに関する一般的な制限事項

データのトランスポートに関する一般的な制限事項があります。また、フル・トランスポータブル・エクスポート/インポート、トランスポータブル表領域またはトランスポータブル表に固有の制限事項もあります。

データをトランスポートする場合は、次の一般的な制限事項に注意してください。

- ソース・データベースとターゲット・データベースで、互換性のあるデータベース・キャラクタ・セットを使用している必要があります。具体的には、次の内容のいずれかを満たしている必要があります。
  - ソース・データベースとターゲット・データベースのデータベース・キャラクタ・セットが同じです。
  - ソース・データベース・キャラクタ・セットがターゲット・データベース・キャラクタ・セットの厳格な(バイナリ)サブセットであり、かつ、次の3つの条件が満たされています。
    - ソース・データベースは、Oracle Database 10*g* リリース1 (10.1.0.3)以上です。
    - トランスポート対象の表領域に、文字長セマンティクスが使用される表の列が含まれていないか、またはソースとターゲットの両方のデータベースにおいて、データベース・キャラクタ・セットの最大文字幅が同じです。
    - トランスポート対象のデータにはCLOBデータ型の列が含まれていないか、またはソース・データベースとターゲット・データベースにおいて、データベース・キャラクタ・セットが両方ともシングルバイトであるか、両方ともマルチバイトです。
  - ソース・データベース・キャラクタ・セットがターゲット・データベース・キャラクタ・セットの厳格な(バイナリ)サブセットであり、かつ、次の2つの条件が満たされています。
    - ソース・データベースは、Oracle Database 10*g* リリース1 (10.1.0.3)以前です。
    - ソース・データベースとターゲット・データベース・キャラクタ・セットにおいて、最大文字幅が同じです。

> **注意:**
>
> Oracle Database によって認識されるキャラクタ・セット間のサブセットとスーパーセットの関係は、『*Oracle Database グローバリゼーション・サポート・ガイド*』を参照してください。

- ソースとターゲットのデータベースで、互換性のある各国語キャラクタ・セットを使用している必要があります。具体的には、次の内容のいずれかを満たしている必要があります。
  - ソースとターゲットのデータベースの各国語キャラクタ・セットが同じです。
  - ソース・データベースはOracle Database 10*g* リリース1 (10.1.0.3)以上であり、トランスポート対象の表領域にはNCHAR、NVARCHAR2、NCLOBのデータ型の列が含まれていません。
- トランスポータブル・エクスポート操作を実行する場合は、次の制限が適用されます。
  - エクスポートを実行するユーザーのデフォルトの表領域を、転送対象となっている表領域のいずれかにすることはできません。
  - エクスポートを実行するユーザーのデフォルトの表領域を、書込み可能にする必要があります。
- 非CDBで、ターゲット・データベースに同じ名前の表領域が含まれている場合は、表領域をトランスポートできません。

CDBで、ターゲットのコンテナに同じ名前の表領域が含まれている場合は、表領域をトランスポートできません。ただし、異なるコンテナには同じ名前の表領域を格納できます。

REMAP_TABLESPACEインポート・パラメータを使用して、データベース・オブジェクトを異なる表領域にインポートできます。または、トランスポート操作を実行する前に、トランスポート対象の表領域またはターゲットの表領域のいずれかの名前を変更できます。

Oracle Database 12cリリース2 (12.2)以降では、Recovery Manager (RMAN)のRECOVERコマンドで、表領域を再マッピングしながら表を異なるスキーマに移動できます。詳細は、『Oracle Databaseバックアップおよびリカバリ・ユーザーズ・ガイド』を参照してください。

- CDBでは、デフォルトのデータ・ポンプ・ディレクトリ・オブジェクトDATA_PUMP_DIRはPDBでは機能しません。データ・ポンプ・エクスポートおよびインポートで使用する明示的なディレクトリ・オブジェクトをPDB内に定義する必要があります。
- XMLTypeを含むデータのトランスポートには、次の制限事項があります。
    - ターゲット・データベースにXML DBがインストールされている必要があります。
    - XMLType表が参照するスキーマをXML DB標準スキーマにすることはできません。
    - トランスポートされたXMLType表のスキーマがターゲット・データベース内に存在しない場合は、スキーマがインポートおよび登録されます。ターゲット・データベース内にすでにスキーマが存在する場合は、インポート中にメッセージが表示されます。
    - XMLTypeを含むデータのメタデータは、データ・ポンプのみを使用してエクスポートおよびインポートする必要があります。

次の問合せでは、XMLTypeを含む表領域のリストが返されます。

```
select distinct p.tablespace_name from dba_tablespaces p,
  dba_xml_tables x, dba_users u, all_all_tables t where
  t.table_name=x.table_name and t.tablespace_name=p.tablespace_name
  and x.owner=u.username;
```

XMLTypeの詳細は、『*Oracle XML DB開発者ガイド*』を参照してください。

- 解釈がアプリケーション固有で、データベースに対して不透明なタイプ(RAW、BFILEなど)は、トランスポートできますが、クロス・プラットフォームのトランスポート操作では変換されません。このタイプの実際の構造はアプリケーションのみが認識するため、このタイプが新規プラットフォームに移動した後、アプリケーションではendiannessの問題に対処する必要があります。OPAQUE型を使用するタイプとオブジェクトも、直接的または間接的にこの制限の影響を受けます。
- TIMESTAMP WITH LOCAL TIME ZONE (TSLTZ)データを含む表がある表領域をタイム・ゾーンが異なるデータベース間でトランスポートする場合、TSLTZデータを含む表はトランスポートされません。エラー・メッセージで、トランスポートされなかった表の説明が示されます。ただし、表領域内のTSLTZデータを含まない表はトランスポートされます。

  データベースのタイム・ゾーンを確認するには、次の問合せを使用します。

  ```
  SELECT DBTIMEZONE FROM DUAL;
  ```

  ALTER DATABASE SQL文を使用すると、データベースのタイム・ゾーンを変更できます。

  データ・ポンプを使用すると、トランスポート操作の完了後に、TSLTZデータを含む表の従来のエクスポート/インポートを実行できます。

- プラットフォーム間でのトランスポート操作の一部として、アナリティック・ワークスペースは使用できません。ソース・プラットフォームおよびターゲット・プラットフォームが別な場合、Data Pumpエクスポート/インポートを使用し、アナリティック・ワークスペースをエクスポートおよびインポートします。アナリティック・ワークスペースの詳細は、『*Oracle OLAP DMLリ*

ファレンス』を参照してください。

> **注意:**
>
> データ・ポンプ・エクスポート・ユーティリティ expdp またはインポート・ユーティリティ impdp は、Oracle サポート・サービスから要求された場合以外、SYSDBA として起動しないでください。SYSDBA は内部的に使用され、一般ユーザーとは異なる特別な機能を持ちます。

**親トピック:** トランスポータブル表領域を使用したデータの移行

# データのトランスポートの互換性に関する注意事項

データをトランスポートする場合、Oracle Databaseはターゲット・データベースが稼働する最低限の互換性レベルを計算します。

ターゲット・データベースが同じプラットフォームにある場合も、別のプラットフォームにある場合も、トランスポータブル表領域を使用して、同じ互換性または高い互換性が設定されているターゲット・データベースにソース・データベースから表領域または表をトランスポートできます。ソース・データベースの互換性レベルがターゲット・データベースの互換性レベルよりも高い場合、データ・トランスポート操作が失敗します。

次の表に、様々な使用例でのソース・データベースとターゲット・データベースの互換性の最低要件を示します。ソース・データベースとターゲット・データベースの互換性設定は同一である必要はありません。

表1-1 互換性の最低要件

| | **互換性の最低設定** | |
|---|---|---|
| **トランスポートの使用例** | **ソース・データベース** | **ターゲット・データベース** |
| フル・トランスポータブル・エクスポート/インポートを使用したデータベースのトランスポート | 12.0 (Oracle Database 12*c* 以降のデータベースの COMPATIBLE 初期化パラメータの設定)<br><br>12 (11.2.0.3 以上のデータベースの VERSION データ・ポンプ・エクスポート・パラメータの設定) | 12.0 (COMPATIBLE 初期化パラメータの設定) |
| トランスポータブル表領域を使用した、同じプラットフォーム上のデータベース間での表領域のトランスポート | 8.0 (COMPATIBLE 初期化パラメータの設定) | 8.0 (COMPATIBLE 初期化パラメータの設定) |
| トランスポータブル表領域を使用した、ターゲット・データベースとデータベース・ブロック・サイズが異なる表領域のトランスポート | 9.0 (COMPATIBLE 初期化パラメータの設定) | 9.0 (COMPATIBLE 初期化パラメータの設定) |
| トランスポータブル表領域を使用した、異なるプラットフォーム上のデータベース間での表領域のトランスポート | 10.0 (COMPATIBLE 初期化パラメータの設定) | 10.0 (COMPATIBLE 初期化パラメータの設定) |
| データベース間での表のトランスポート | 11.2.0 (Oracle Database 12*c* 以降のデータベースの COMPATIBLE 初期化パラメータの設定) | 11.2.0 (COMPATIBLE 初期化パラメータの設定) |

注意:

- フル・トランスポータブル・エクスポート/インポートを使用する場合、ソース・データベースは Oracle Database 11*g* リリース 2 (11.2.0.3)以降のデータベースであり、ターゲット・データベースは Oracle Database 12*c* 以降のデータベースである必要があります。
- Oracle Database 11*g* リリース 2 (11.2.0.3)以降のデータベースから Oracle Database 12*c* 以降のデータベースにトランスポートする場合は、VERSION データ・ポンプ・エクスポート・パラメータを 12 以上に設定する必要があります。
- Oracle Database 19c データベースから Oracle Database 19c データベースにトランスポートする場合は、COMPATIBLE 初期化パラメータを 19.0.0 以上に設定する必要があります。

**親トピック:** トランスポータブル表領域を使用したデータの移行

# トランスポータブル表領域に関する制限事項

この項では、トランスポータブル表領域の制限事項を示します。

トランスポータブル表領域に関する次の制限事項に注意してください。

- トランスポータブル表領域には、「データのトランスポートに関する一般的な制限事項」で説明されている一般的な制限事項が適用されます。
- 表領域セットをトランスポートする場合、基礎になるオブジェクトを持つオブジェクト(マテリアライズド・ビューなど)またはオブジェクトを含むオブジェクト(パーティション表など)は、それらのオブジェクトがすべて表領域セットに含まれている場合のみトランスポートできます。
- トランスポータブル表領域では、タイム・ゾーン・ファイルのバージョンが異なるプラットフォーム間で、TIMESTAMP WITH TIMEZONE (TSTZ)データを含む表をトランスポートできません。トランスポータブル表領域操作では、これらの表をスキップします。これらの表は、従来と同じようにエクスポートおよびインポートできます。

  詳細は、『*Oracle Databaseユーティリティ*』を参照してください。
- トランスポータブル表領域セットに、SYSTEMやSYSAUXなどの管理表領域を含めることはできません。

**親トピック:** トランスポータブル表領域を使用したデータの移行

# 2 ターゲット・サーバーの準備

この章では、Database Configuration Assistant (DBCA)を使用してターゲット・サーバーに宛先データベースを作成する方法について説明します。

ソース・サーバーとターゲット・サーバーでPINGコマンドを実行して、両方のサーバーにアクセスできるかどうかを確認します。

- ターゲット・データベースの作成
  ターゲット・システムに新しいデータベースを作成します。

## ターゲット・データベースの作成

新しいデータベースをターゲット・システムに作成します。

新しいターゲット・データベースは、最初はSYSTEM、SYSAUX、UNDO、一時表領域およびユーザー表領域のみで構成されます。DBCAを使用して、ターゲット・データベースを作成できます。

ターゲット・データベースを作成するときは、次の点を考慮してください。

- 元のソース・データベースのすべてのユーザー表領域はトランスポートされ、新しいターゲット・データベースにプラグインされますが、最初にターゲット・データベースに、トランスポートされるユーザー表領域のプレースホルダ表領域が含まれている必要があります。ターゲット・データベースに最初に作成するユーザー表領域のサイズは小さくすることができ、ターゲット表領域はソース・データベースのサイズと一致させる必要はありません。プレースホルダ表領域は、ソース・システムから表領域をトランスポートする前に、ターゲット・データベースから削除されます。
- SYSTEM、SYSAUX、UNDOおよび一時表領域のサイズは、ソース・データベースの表領域以上である必要があります。
- 新しいターゲット・データベース内の各ログ・ファイル・グループのログ・ファイルのサイズおよびメンバー数は、ソース・データベースと同じか、それより大きくする必要があります。
- ソース・データベースとターゲット・データベースで、同じキャラクタ・セットおよび同じ各国語キャラクタ・セットを使用する必要があります。次の問合せを発行して、ソース・データベースのキャラクタ・セットを確認します。

  ```
  SQL> select * from database_properties where property_name like '%CHARACTERSET';
  ```

- ソース・データベースで使用されるデータベース・オプションおよびコンポーネントを、ターゲット・データベースにインストールする必要があります。
  - V$OPTIONビューを問い合せて、現在インストールされているデータベース・オプションを取得します。
  - DBA_REGISTRYを問い合せて、現在インストールされているデータベース・コンポーネントを取得します。

**親トピック:** ターゲット・サーバーの準備

# 3 ターゲット・オペレーティング・システムのendian形式へのデータ変換

ターゲット・オペレーティング・システム・プラットフォームで使用されているendian形式にデータのendian形式を変換して、データ移行を準備します。

- プラットフォーム間でのデータの変換
  トランスポータブル操作を実行するときに、ソース・プラットフォームとターゲット・プラットフォームでendiannessが異なる場合は、トランスポートするデータをターゲット・プラットフォームの形式に変換する必要があります。ソース・プラットフォームとターゲット・プラットフォームでendiannessが同じ場合は、データ変換は必要ありません。データを変換するには、DBMS_FILE_TRANSFERパッケージまたはRMAN CONVERTコマンドを使用できます。
- DBMS_FILE_TRANSFERパッケージを使用したプラットフォーム間でのデータの変換
  DBMS_FILE_TRANSFERパッケージのGET_FILEまたはPUT_FILEプロシージャを使用すると、データファイルの転送中にプラットフォーム間でデータを変換できます。
- RMANを使用したプラットフォーム間でのデータの変換
  RMAN CONVERTコマンドを使用してデータを変換する場合は、データ・ポンプ・エクスポートを実行した後にソース・プラットフォームでデータを変換するか、またはデータ・ポンプ・インポートを実行する前にターゲット・プラットフォームでデータを変換できます。いずれの場合も、データファイルをソース・システムからターゲット・システムに転送する必要があります。

# プラットフォーム間でのデータの変換

トランスポータブル操作を実行するときに、ソース・プラットフォームとターゲット・プラットフォームでendiannessが異なる場合は、トランスポートするデータをターゲット・プラットフォームの形式に変換する必要があります。ソース・プラットフォームとターゲット・プラットフォームでendiannessが同じ場合は、データ変換は必要ありません。データを変換するには、DBMS_FILE_TRANSFERパッケージまたはRMAN CONVERTコマンドを使用できます。

注意:

これらの項には記載されていない制限事項が適用されることがあります。詳細は、次のドキュメントを参照してください。

- プラットフォームの endianness をチェックする方法の詳細は、「プラットフォーム間でのデータのトランスポート」を参照してください
- DBMS_FILE_TRANSFER パッケージに関する制限事項の詳細は、『*Oracle Database PL/SQL パッケージおよびタイプ・リファレンス*』を参照してください。
- RMAN CONVERT コマンドに関する制限事項の詳細は、『*Oracle Database バックアップおよびリカバリ・リファレンス*』を参照してください。

**親トピック:** ターゲット・オペレーティング・システムのendian形式へのデータ変換

# DBMS_FILE_TRANSFERパッケージを使用したプラットフォーム間でのデータの変換

DBMS_FILE_TRANSFERパッケージのGET_FILEまたはPUT_FILEプロシージャを使用すると、データファイルの転送中にプラットフォーム間でデータを変換できます。

これらのプロシージャのいずれかを使用して、ソース・プラットフォームとターゲット・プラットフォーム間でデータファイルを移動する場合、各データファイル内の各ブロックがターゲット・プラットフォームのendiannessに変換されます。

この項では、例を使用して、データファイルを異なるプラットフォームに変換するためにDBMS_FILE_TRANSFERパッケージを使用する方法を示します。この例では、次のことを想定しています。

- GET_FILEプロシージャでデータファイルを転送します。
- mytable.342.123456789データファイルを異なるプラットフォームに転送します。
- ソース・プラットフォームのendiannessはターゲット・プラットフォームのendiannessと異なります。
- ソース・データベースのグローバル名はdbsa.example.comです。
- ソース・データベースとターゲット・データベースの両方でOracle Automatic Storage Management(Oracle ASM)が使用されています。

> **注意:**
>
> DBMS_FILE_TRANSFER パッケージを使用して、endianness が同じプラットフォーム間でデータファイルを転送することもできます。

GET_FILEプロシージャを使用して転送することによってデータファイルを変換するには、次のステップを実行します。

1. SQL*Plusを使用して、ディレクトリ・オブジェクトを作成できる管理ユーザーとしてソース・データベースに接続します。
2. ターゲット・データベースに転送するデータファイルを格納するディレクトリ・オブジェクトを作成します。

   たとえば、+data/dbsa/datafileディレクトリに対してsales_dir_sourceというディレクトリ・オブジェクトを作成するには、次のSQL文を実行します。

   ```
   CREATE OR REPLACE DIRECTORY sales_dir_source
      AS '+data/dbsa/datafile';
   ```

   ディレクトリ・オブジェクトの作成時に、指定したファイル・システム・ディレクトリが存在している必要があります。
3. SQL*Plusを使用して、データベース・リンクの作成、ディレクトリ・オブジェクトの作成およびDBMS_FILE_TRANSFERパッケージのプロシージャの実行ができる管理ユーザーとしてターゲット・データベースに接続します。
4. ターゲット・データベースからソース・データベースへデータベース・リンクを作成します。

   ソース・データベースの接続されるユーザーには、ステップ2で作成したディレクトリ・オブジェクトについての読取り権限が必要です。
5. ソース・データベースから転送するデータファイルを格納するディレクトリ・オブジェクトを作成します。

   DBMS_FILE_TRANSFERパッケージのプロシージャを実行するローカル・データベースのユーザーには、ディレクトリ・オブジェ

クトについての書込み権限が必要です。

たとえば、+data/dbsb/datafileディレクトリに対してsales_dir_targetというディレクトリ・オブジェクトを作成するには、次のSQL文を実行します。

```
CREATE OR REPLACE DIRECTORY sales_dir_target
  AS '+data/dbsb/datafile';
```

6. DBMS_FILE_TRANSFERパッケージのGET_FILEプロシージャを実行して、データファイルを転送します。

たとえば、次のプロシージャを実行し、ステップ4で作成したデータベース・リンクを使用して、mytable.342.123456789データファイルをソース・データベースからターゲット・データベースに転送します。

```
BEGIN
  DBMS_FILE_TRANSFER.GET_FILE(
    source_directory_object      => 'sales_dir_source',
    source_file_name             => 'mytable.342.123456789',
    source_database              => 'dbsa.example.com',
    destination_directory_object => 'sales_dir_target',
    destination_file_name        => 'mytable');
END;
/
```

**注意:**

この例では、宛先データファイル名はmytableです。Oracle ASMでは、GET_FILEプロシージャのdestination_file_nameパラメータで完全修飾ファイル名形式を使用できません。

**関連項目:**

- DBMS_FILE_TRANSFERパッケージの使用方法の詳細は、『*Oracle Database PL/SQLパッケージおよびタイプ・リファレンス*』を参照してください。
- ASMの完全修飾ファイル名形式の詳細は、『*Oracle Automatic Storage Management*管理者ガイド』を参照してください。
- データベース・リンクの作成方法は、『*Oracle Database SQL*言語リファレンス』を参照してください

**親トピック:** ターゲット・オペレーティング・システムのendian形式へのデータ変換

# RMANを使用したプラットフォーム間でのデータの変換

RMAN CONVERTコマンドを使用してデータを変換する場合は、データ・ポンプ・エクスポートを実行した後にソース・プラットフォームでデータを変換するか、またはデータ・ポンプ・インポートを実行する前にターゲット・プラットフォームでデータを変換できます。いずれの場合も、データファイルをソース・システムからターゲット・システムに転送する必要があります。

次のRMAN CONVERTコマンドを使用してデータを変換できます。

- CONVERT DATAFILE
- CONVERT TABLESPACE
- CONVERT DATABASE

注意:

- RMAN CONVERT コマンドにはデータ型の制約が適用されます。
- RMAN CONVERT コマンドでは、UNDO セグメントを含むデータファイルについて、異なる endian 形式間でのデータファイルの変換はサポートされていません。

- エクスポート後のソース・システムでの表領域の変換
  例を使用して、表領域を異なるプラットフォームに変換するために、RMAN CONVERT TABLESPACEコマンドを使用する方法を示します。
- インポート前のターゲット・システムでのデータファイルの変換
  例を使用して、データファイルを異なるプラットフォームに変換するために、RMAN CONVERT DATAFILEコマンドを使用する方法を示します。

**関連項目:**

- 『*Oracle Database*バックアップおよびリカバリ・リファレンス』
- 『*Oracle Database*バックアップおよびリカバリ・アドバンスト・ユーザーズ・ガイド』

**親トピック:** ターゲット・オペレーティング・システムのendian形式へのデータ変換

## エクスポート後のソース・システムでの表領域の変換

例を使用して、表領域を異なるプラットフォームに変換するために、RMAN CONVERT TABLESPACEコマンドを使用する方法を示します。

この例では、次のことを想定しています。

- sales_1およびsales_2表領域を異なるプラットフォームにトランスポートします。
- ソース・プラットフォームのendiannessはターゲット・プラットフォームのendiannessと異なります。
- 表領域セットをターゲット・システムにトランスポートする前に、ソース・システムでデータを変換します。

- ソース・データベースでデータ・ポンプ・エクスポートを完了しています。

ソース・システムで表領域を変換するには、次のステップを実行します。

1. コマンド・プロンプトで、RMANを起動してソース・データベースに接続します。

```
$ RMAN TARGET /

Recovery Manager: Release 12.1.0.1.0 - Production

Copyright (c) 1982, 2012, Oracle and/or its affiliates.  All rights reserved.

connected to target database: salesdb (DBID=3295731590)
```

2. RMAN CONVERT TABLESPACEコマンドを使用してデータファイルを変換し、ソース・プラットフォーム上の一時的な場所に格納します。

   この例では、一時的な場所はディレクトリ/tmpと想定し、すでに作成されているとします。変換されたデータファイルの名前は、システムによって割り当てられます。

```
RMAN> CONVERT TABLESPACE sales_1,sales_2
2> TO PLATFORM 'Microsoft Windows IA (32-bit)'
3> FORMAT '/tmp/%U';

Starting conversion at source at 30-SEP-08
using channel ORA_DISK_1
channel ORA_DISK_1: starting datafile conversion
input datafile file number=00007 name=/u01/app/oracle/oradata/salesdb/sales_101.dbf
converted datafile=/tmp/data_D-SALESDB_I-1192614013_TS-SALES_1_FNO-7_03jru08s
channel ORA_DISK_1: datafile conversion complete, elapsed time: 00:00:45
channel ORA_DISK_1: starting datafile conversion
input datafile file number=00008 name=/u01/app/oracle/oradata/salesdb/sales_201.dbf
converted datafile=/tmp/data_D-SALESDB_I-1192614013_TS-SALES_2_FNO-8_04jru0aa
channel ORA_DISK_1: datafile conversion complete, elapsed time: 00:00:25
Finished conversion at source at 30-SEP-08
```

   **関連項目:**

   RMANのCONVERTコマンドの詳細は、『*Oracle Database*バックアップおよびリカバリ・リファレンス』を参照してください。

3. Recovery Managerを終了します。

```
RMAN> exit
Recovery Manager complete.
```

4. データファイルをターゲット・システムに転送します。

**親トピック:** RMANを使用したプラットフォーム間でのデータの変換

## インポート前のターゲット・システムでのデータファイルの変換

例を使用して、データファイルを異なるプラットフォームに変換するために、RMAN CONVERT DATAFILEコマンドを使用する方法を示します。

変換時、データファイルは、表領域名ではなく、ファイル名で指定します。表領域のメタデータがインポートされるまで、ターゲット・インスタンスでは対象の表領域名を認識できません。

この例では、次のことを想定しています。

- トランスポートする表領域のデータファイルはまだ変換していません。

  DBMS_FILE_TRANSFERパッケージを使用してデータファイルをターゲット・システムに転送した場合、データファイルはファイル転送中に自動的に変換されています。「DBMS_FILE_TRANSFERパッケージを使用したプラットフォーム間でのデータの変換」を参照してください。

- 次のデータファイルを異なるプラットフォームにトランスポートします。
  - C:¥Temp¥sales_101.dbf
  - C:¥Temp¥sales_201.dbf
- ソース・プラットフォームのendiannessはターゲット・プラットフォームのendiannessと異なります。
- データ・ポンプ・インポートを実行する前に、ターゲット・システムでデータを変換します。
- 変換されたデータファイルは、C:¥app¥orauser¥oradata¥orawin¥(ターゲット・システムの既存データファイルの場所)に配置されます。

ターゲット・システムで表領域を変換するには、次のステップを実行します。

1. SQL*Plusを実行している場合は、ホスト・システムに戻ります。

```
SQL> HOST
```

2. RMAN CONVERT DATAFILEコマンドを使用して、データファイルをターゲット・プラットフォームに変換します。

```
C:¥>RMAN TARGET /

Recovery Manager: Release 12.1.0.1.0 - Production

Copyright (c) 1982, 2012, Oracle and/or its affiliates.  All rights reserved.

connected to target database: ORAWIN (DBID=3462152886)

RMAN> CONVERT DATAFILE
2>'C:¥Temp¥sales_101.dbf',
3>'C:¥Temp¥sales_201.dbf'
4>TO PLATFORM="Microsoft Windows IA (32-bit)"
5>FROM PLATFORM="Solaris[tm] OE (32-bit)"
6>DB_FILE_NAME_CONVERT=
7>'C:¥Temp¥', 'C:¥app¥orauser¥oradata¥orawin¥'
8> PARALLELISM=4;
```

ソースの場所、ターゲットの場所、あるいはその両方でOracle Automatic Storage Management(Oracle ASM)を使用しない場合、ソース・プラットフォームとターゲット・プラットフォームはオプションです。RMANでは、データファイルを調べてソース・プラットフォームを判別し、デフォルトのターゲット・プラットフォームは、変換を実行するホストのプラットフォームになります。

ソースおよびターゲットの場所の両方でOracle ASMを使用する場合は、DB_FILE_NAME_CONVERT句でソース・プラットフォームとターゲット・プラットフォームを指定する必要があります。

**関連項目:**

RMANのCONVERTコマンドの詳細は、『*Oracle Database*バックアップおよびリカバリ・リファレンス』を参照してください。

3. Recovery Managerを終了します。

```
RMAN> exit
Recovery Manager complete.
```

**親トピック:** RMANを使用したプラットフォーム間でのデータの変換

# 4 SQL*Plusを使用したデータベースへの接続

SQL*Plusを使用してOracle Databaseインスタンスに接続します。

- ステップ1: コマンド・ウィンドウのオープン
  プラットフォームで必要な処理を実行して、オペレーティング・システムのコマンドを入力できるウィンドウを開きます。
- ステップ2: オペレーティング・システムの環境変数の設定
  プラットフォームによっては、SQL*Plusを起動する前に、環境変数を設定するか少なくとも正しく設定されていることを確認することが必要な場合があります。
- ステップ3: SQL*Plusの起動
  SQL*Plusを起動します。
- ステップ4: SQL*PlusのCONNECTコマンドの発行
  最初にOracle Databaseインスタンスに接続する、または任意の時点で別のユーザーとして再接続するには、SQL*PlusのCONNECTコマンドを発行します。

# ステップ1: コマンド・ウィンドウのオープン

プラットフォームで必要な処理を実行して、オペレーティング・システムのコマンドを入力できるウィンドウを開きます。

- コマンド・ウィンドウを開きます。

**親トピック:** SQL*Plusを使用したデータベースへの接続

# ステップ2: オペレーティング・システムの環境変数の設定

プラットフォームによっては、SQL*Plusを起動する前に、環境変数を設定するか少なくとも正しく設定されていることを確認することが必要な場合があります。

たとえば、ほとんどのプラットフォームでは、環境変数ORACLE_SIDおよびORACLE_HOMEを設定する必要があります。さらに、ORACLE_HOME/binディレクトリを含めるようにPATH環境変数を構成する必要があります。一部のプラットフォームでは、さらに環境変数の設定が必要な場合があります。

- UNIXおよびLinuxでは、オペレーティング・システムのコマンドを入力して環境変数を設定します。
- Windowsでは、Windowsレジストリ内のORACLE_HOMEとORACLE_SIDにOracle Universal Installer (OUI)によって値が自動的に割り当てられます。

インストール時にデータベースを作成しなかった場合は、レジストリ内のORACLE_SIDがOUIによって設定されないため、後でデータベースを作成するときに、ORACLE_SID環境変数をコマンド・ウィンドウから設定する必要があります。

UNIXおよびLinuxのインストールには、oraenvとcoraenvの2つのスクリプトが含まれています。これらのスクリプトを使用すると、環境変数を容易に設定できます。詳細は、*Oracle Database*管理者リファレンス *for Linux and UNIX-Based Operating Systems*を参照してください。

いずれのプラットフォームでも、異なるOracleホームを使用するインスタンス間で切り替える場合は、ORACLE_HOME環境変数を変更する必要があります。同じOracleホームを複数のインスタンスで共有している場合は、インスタンスを切り替えるときにORACLE_SIDのみを変更する必要があります。

例4-1 UNIXの環境変数の設定(Cシェル)

```
setenv ORACLE_SID orcl
setenv ORACLE_HOME /u01/app/oracle/product/database_release_number/dbhome_1
setenv LD_LIBRARY_PATH $ORACLE_HOME/lib:/usr/lib:/usr/dt/lib:/usr/openwin/lib:/usr/ccs/lib
```

例4-2 UNIXの環境変数の設定(Bashシェル)

```
export ORACLE_SID=orcl
export ORACLE_HOME=/u01/app/oracle/product/database_release_number/dbhome_1
export LD_LIBRARY_PATH=$ORACLE_HOME/lib:/usr/lib:/usr/dt/lib:/usr/openwin/lib:/usr/ccs/lib
```

例4-3 Windowsの環境変数の設定

```
SET ORACLE_SID=orawin2
```

例4-3では、ORACLE_HOMEおよびORACLE_SIDがレジストリに設定されているが、別のインスタンスに接続するにはORACLE_SIDのレジストリ値を上書きする必要があると仮定しています。

Windowsでは、コマンド・プロンプト・ウィンドウで設定した環境変数値でレジストリの値が上書きされます。

**親トピック:** SQL*Plusを使用したデータベースへの接続

# ステップ3: SQL*Plusの起動

SQL*Plusを起動します。

1. 次のいずれかを行います:
   - PATH環境変数に*ORACLE_HOME*/binが含まれていることを確認します。
   - *ORACLE_HOME*/binディレクトリに移動します。PATH環境変数にドット(".")が含まれていることを確認します。
2. 次のコマンドを入力します(UNIXとLinuxでは大/小文字が区別されます)。

   ```
   sqlplus /nolog
   ```

   完全なパスを指定してsqlplusコマンドを実行することもできます。

   ```
   ORACLE_HOME/bin/sqlplus /nolog
   ```

**親トピック:** SQL*Plusを使用したデータベースへの接続

# ステップ4: SQL*PlusのCONNECTコマンドの発行

最初にOracle Databaseインスタンスに接続する、または任意の時点で別のユーザーとして再接続するには、SQL*PlusのCONNECTコマンドを発行します。

- SQL*Plusで、CONNECTコマンドを発行します。

例4-4 ローカル・データベース・ユーザーへの接続

次の簡単な例では、ユーザーSYSTEMでローカル・データベースに接続します。SQL*Plusによって、ユーザーSYSTEMのパスワードの入力が求められます。

```
connect system
```

例4-5 SYSDBA権限でのローカル・データベース・ユーザーへの接続

次の例では、SYSDBA権限のユーザーSYSでローカル・データベースに接続します。SQL*Plusによって、ユーザーSYSのパスワードの入力が求められます。

```
connect sys as sysdba
```

ユーザーSYSとして接続するときには、AS SYSDBAを指定して接続する必要があります。

例4-6 SYSBACKUP権限でのローカル・データベース・ユーザーへの接続

次の例では、SYSBACKUP権限のユーザーSYSBACKUPでローカル・データベースに接続します。SQL*Plusによって、ユーザーSYSBACKUPのパスワードの入力が求められます。

```
connect sysbackup as sysbackup
```

ユーザーSYSBACKUPとして接続するときには、AS SYSBACKUPを指定して接続する必要があります。

例4-7 オペレーティング・システム認証によるSYSDBA権限でのローカルな接続

次の例では、オペレーティング・システム認証を使用してSYSDBA権限でローカルに接続します。

```
connect / as sysdba
```

例4-8 簡易接続構文による接続

この例では、簡易接続の構文を使用し、ユーザーsalesadminとして、ホストdbhost.example.comで動作するリモート・データベースに接続します。Oracle Netリスナー(リスナー)は、デフォルト・ポート(1521)でリスニングしています。データベース・サービスは、sales.example.comです。SQL*Plusによって、ユーザーsalesadminのパスワードの入力が求められます。

```
connect salesadmin@"dbhost.example.com/sales.example.com"
```

例4-9 サービス・ハンドラ・タイプを指定した簡易接続構文による接続

次の例は、サービス・ハンドラのタイプが指定されていること以外は例4-8と同じです。

```
connect salesadmin@"dbhost.example.com/sales.example.com:dedicated"
```

例4-10 デフォルト以外のリスナー・ポートを使用した簡易接続構文による接続

次の例は、デフォルト・ポート番号ではない1522でリスナーがリスニングしていること以外は例4-8と同じです。

```
connect salesadmin@"dbhost.example.com:1522/sales.example.com"
```

例4-11 ホストIPアドレスを使用した簡易接続構文による接続

この例は例4-8とほぼ同じで、ホスト名のかわりにホストのIPアドレスが使用されている点のみが異なります。

```
connect salesadmin@"192.0.2.5/sales.example.com"
```

例4-12 IPv6アドレスによる接続

この例では、IPv6アドレスを使用して接続します。角括弧で囲まれている点に注意してください。

```
connect salesadmin@"[2001:0DB8:0:0::200C:417A]/sales.example.com"
```

例4-13 インスタンスの指定による接続

次の例では、接続先のインスタンスを指定し、データベース・サービス名を省略します。インスタンスのみを指定する場合はサービス・ハンドラのタイプを指定できないことに注意してください。

```
connect salesadmin@"dbhost.example.com//orcl"
```

例4-14 ネット・サービス名による接続

次の例では、ユーザーsalesadminとして、ネット・サービス名sales1によって指定されたデータベース・サービスにリモート接続します。SQL*Plusによって、ユーザーsalesadminのパスワードの入力が求められます。

```
connect salesadmin@sales1
```

例4-15 外部認証による接続

次の例では、外部認証を使用して、ネット・サービス名sales1によって指定されたデータベース・サービスにリモート接続します。

```
connect /@sales1
```

例4-16 SYSDBA権限と外部認証による接続

次の例では、外部認証を使用してSYSDBA権限で、ネット・サービス名sales1によって指定されたデータベース・サービスにリモート接続します。

```
connect /@sales1 as sysdba
```

例4-17 サービス名を使用したユーザーとしての接続

次の例では、ユーザーsalesadminとして、ネット・サービス名sales1によって指定されたデータベース・サービスにリモート接続します。データベース・セッションは、rev21エディションで開始されます。SQL*Plusによって、ユーザーsalesadminのパスワードの入力が求められます。

```
connect salesadmin@sales1 edition=rev21
```

注意:

SYSDBA 権限を持つユーザーとしてデータベースに接続しているときに問題が発生した場合は、My Oracle Support ノート 69642.1、233223.1、18089.1 および 747456.1 を参照してください。

**親トピック:** SQL*Plusを使用したデータベースへの接続

# 5 Oracle Databaseの移行

トランスポータブル表領域機能を使用して、表領域セットを別のOracle Databaseにコピーします。

- キャラクタ・セットの互換性の確認
  ソース・データベースおよび宛先データベースでこれらのコマンドを実行し、互換性のあるキャラクタ・セットを見つけます。
- データベース間での表領域のトランスポート
  表領域または表領域セットをデータベース間でトランスポートできます。
- タスク1: 自己完結型の表領域セットの選択
  トランスポータブル・セットのデータベース・オブジェクトとトランスポータブル・セット外のデータベース・オブジェクトの間に、論理的または物理的な依存関係がある場合があります。トランスポートできるのは、自己完結型である表領域セットのみです。つまり、表領域セット内のデータベース・オブジェクトは、その表領域セット外のデータベース・オブジェクトのいずれにも依存しません。
- タスク2: トランスポータブル表領域セットの生成
  トランスポートする表領域セットが自己完結型であることを確認した後で、トランスポータブル表領域セットを生成します。
- タスク3: エクスポート・ダンプ・ファイルのトランスポート
  ダンプ・ファイルを、DATA_PUMP_DIRディレクトリ・オブジェクトで指し示されているディレクトリ、または他の任意のディレクトリにトランスポートします。新しい場所はターゲット・データベースへアクセス可能であることが必要です。
- タスク4: 表領域セットのトランスポート
  表領域のデータファイルをターゲット・データベースへアクセス可能なディレクトリにトランスポートします。
- タスク5: (オプション)表領域を読取り/書込みモードに戻す
  トランスポートした表領域をソース・データベースで再び読取り/書込みモードにします。
- タスク6: 表領域セットのインポート
  トランスポータブル表領域に対する操作を完了するために、表領域セットをインポートします。
- 移行後タスク
  これらのタスクを完了して、使用するターゲットのOracle Databaseを準備します。

# キャラクタ・セットの互換性の確認

ソース・データベースおよび宛先データベースでこれらのコマンドを実行し、互換性のあるキャラクタ・セットを見つけます。

```
SQL> show parameter CHARACTER

NAME                    TYPE        VALUE
---------               ---------   ----------
nls_numeric_characters  string

SQL> select * from database_properties where PROPERTY_NAME in
('NLS_CHARACTERSET','NLS_NCHAR_CHARACTERSET');

PROPERTY_NAME            PROPERTY_VALUE DESCRIPTION
-----------------------  -------------  ------------------
NLS_NCHAR_CHARACTERSET   AL16UTF16      NCHAR Character set
NLS_CHARACTERSET         WE8MSWIN1252   Character set

SQL> select * from nls_database_parameters where parameter like '%SET%';

PROPERTY_NAME              VALUE
-------------------------  ---------
NLS_NCHAR_CHARACTERSET     AL16UTF16
NLS_CHARACTERSET           WE8MSWIN1252
```

**親トピック:** Oracle Databaseの移行

# データベース間での表領域のトランスポート

データベース間で表領域または表領域のセットをトランスポートできます。

次のタスクのリストでは、表領域のトランスポート処理の概要を示します。各タスクの詳細は、後続の例で示します。

1. 自己完結型の表領域セットの選択
2. ソース・データベースで、表領域セットを読取り専用モードに構成してから、トランスポータブル表領域セットを生成します。

   トランスポータブル表領域セット(トランスポータブル・セット)は、トランスポートされる表領域セットのデータファイルと、そのセットの構造情報(メタデータ)を含むエクスポート・ダンプ・ファイルから構成されます。データ・ポンプを使用してエクスポートを実行します。
3. エクスポート・ダンプ・ファイルをトランスポートします。

   エクスポート・ダンプ・ファイルをターゲット・データベースへアクセス可能な場所にコピーします。
4. 表領域セットのトランスポート

   データファイルをターゲット・データベースへアクセス可能なディレクトリにコピーします。

   ソース・プラットフォームとターゲット・プラットフォームが異なる場合は、V$TRANSPORTABLE_PLATFORMビューに対して問合せを実行して、各プラットフォームのendian形式をチェックできます。

   ソース・プラットフォームのendian形式がターゲット・プラットフォームのendian形式と異なる場合は、次のいずれかの方法を使用してデータファイルを変換します。

   - DBMS_FILE_TRANSFERパッケージのGET_FILEまたはPUT_FILEプロシージャを使用して、データファイルを転送します。これらのプロシージャを使用すると、データファイルがターゲット・プラットフォームのendian形式に自動的に変換されます。
   - RMAN CONVERTコマンドを使用して、データファイルをターゲット・プラットフォームのendian形式に変換します。

   

   注意:

   UNDO セグメントを含むデータファイルでは、異なる endian 形式間でのデータファイルの変換はサポートされていません。
5. (オプション)ソース・データベースで表領域を読取り/書込みモードに戻します。
6. ターゲット・データベースで、データベース・セットをインポートします。

   データ・ポンプ・ユーティリティを実行して、表領域セットのメタデータをインポートします。

例5-1 例

表領域をトランスポートするタスクについては、次のデータファイルと表領域を想定した例で詳細に説明します。

| 表領域 | データファイル |
| --- | --- |
| sales_1 | /u01/app/oracle/oradata/salesdb/sales_101.dbf |

| 表領域 | データファイル |
| --- | --- |
| sales_2 | /u01/app/oracle/oradata/salesdb/sales_201.dbf |

**親トピック:** Oracle Databaseの移行

# タスク1: 自己完結型の表領域セットの選択

トランスポータブル・セットのデータベース・オブジェクトとトランスポータブル・セット外のデータベース・オブジェクトの間に、論理的または物理的な依存関係がある場合があります。トランスポートできるのは、自己完結型である表領域セットのみです。つまり、表領域セット内のデータベース・オブジェクトは、その表領域セット外のデータベース・オブジェクトのいずれにも依存しません。

次に、自己完結した表領域に違反する例を示します。

- 表領域セット内に、そのセットに含まれない表に関する索引が含まれている場合。

注意:

表に対応する索引が表領域セットの外部にある場合は、違反になりません。

- パーティション表の一部が表領域セットに含まれている場合。

  コピーする表領域セットは、パーティション化した表のすべてのパーティションが含まれている状態、またはまったく含まれていない状態にしてください。パーティション表のサブセットをトランスポートするには、パーティションを表に変換する必要があります。

  パーティションの変換の詳細は、『*Oracle Database VLDBおよびパーティショニング・ガイド*』を参照してください。

- 参照整合性制約がセット境界を越えて別の表を指している場合。

  表領域セットをトランスポートするときには、参照整合性制約を含めるかどうかを選択できます。ただし、そうすることによって、表領域セットの自己完結性に影響を与える場合があります。制約をトランスポートしなければ、その制約はポインタとはみなされません。

- 表領域セット内の表に、そのセットに含まれないLOBを指すLOB列が含まれている場合

- ユーザーAが登録されたXML DBスキーマ(*.xsd)にユーザーBが登録されたグローバル・スキーマをインポートする際、ユーザーAのデフォルト表領域が表領域A、ユーザーBのデフォルト表領域が表領域Bで、表領域Aのみが表領域セットに含まれている場合。

表領域セットが自己完結型かどうかを判別するには、オラクル社が提供するDBMS_TTSパッケージのTRANSPORT_SET_CHECKプロシージャを実行します。このプロシージャを実行するには、EXECUTE_CATALOG_ROLEロール(最初はSYSに付与されている)を付与されている必要があります。

DBMS_TTS.TRANSPORT_SET_CHECKプロシージャを実行するときは、自己完結かどうかを調べるトランスポータブル・セットの表領域のリストを指定します。制約を含むかどうかを指定することもできます。厳密または完全な完結であるかを調べる場合は、TTS_FULL_CHECKパラメータをTRUEに設定する必要があります。

厳密または完全な完結のチェックは、トランスポータブル・セットから外部への参照のみではなく、外部からトランスポータブル・セットへの参照も捕捉する必要がある場合に実行します。依存オブジェクトがトランスポータブル・セットに完全に含まれているか、またはトランスポータブル・セットの外部にのみ存在することが必要な場合は、表領域のPoint-in-Timeリカバリ(TSPITR)を実行します。

たとえば、表tを含んでいるが、その索引iを含んでいない表領域に対してTSPITRを実行すると、トランスポート後に索引とデータの整合性がなくなるため、これは違反になります。完全完結チェックを実行することにより、トランスポータブル・セットからの依存関係またはトランスポータブル・セットへの依存関係がないことが保証されます。詳細は、『*Oracle Databaseバックアップおよび*

リカバリ・ユーザーズ・ガイド』のTSPITRの例を参照してください。

> **注意:**
>
> デフォルトでは、トランスポータブル表領域は、完全完結しているかどうかではなく自己完結しているかどうかがチェックされます。

次の文を使用して、表領域sales_1およびsales_2が自己完結しているかどうかを、参照整合性制約を考慮して(TRUEを指定して)調べます。

```
EXECUTE DBMS_TTS.TRANSPORT_SET_CHECK('sales_1,sales_2', TRUE);
```

DBMS_TTS.TRANSPORT_SET_CHECKプロシージャを実行した後に、TRANSPORT_SET_VIOLATIONSビューからすべての違反を選択して表示できます。表領域セットが自己完結している場合、このビューは空になります。次の例は、表領域セットの境界を超えている外部キー定数dept_fkと、表領域セットに部分的に含まれているパーティション表jim.salesという、2つの違反がある場合を示しています。

```
SELECT * FROM TRANSPORT_SET_VIOLATIONS;

VIOLATIONS
--------------------------------------------------------------------------------
Constraint DEPT_FK between table JIM.EMP in tablespace SALES_1 and table
JIM.DEPT in tablespace OTHER
Partitioned table JIM.SALES is partially contained in the transportable set
```

sales_1およびsales_2をトランスポータブルにする前に、これらの違反を解決する必要があります。次のタスクで説明するように、整合性制約違反を回避するための選択肢の1つとして、整合性制約をエクスポートしない方法があります。

**関連項目:**

- DBMS_TTSパッケージの詳細は、『*Oracle Database PL/SQLパッケージおよびタイプ・リファレンス*』を参照してください。
- TSPITRにおけるDBMS_TTSパッケージの使用の詳細は、『*Oracle Databaseバックアップおよびリカバリ・ユーザーズ・ガイド*』を参照してください。

**親トピック:** Oracle Databaseの移行

# タスク2: トランスポータブル表領域セットの生成

トランスポートする表領域セットが自己完結型であることを確認した後で、トランスポータブル表領域セットを生成します。

トランスポータブル表領域セットを生成する手順は、次のとおりです。

1. SQL*Plusを起動し、管理者として、あるいはALTER TABLESPACEまたはMANAGE TABLESPACEシステム権限を持つユーザーとしてデータベースに接続します。
2. セット内のすべての表領域を読取り専用にします。

```
ALTER TABLESPACE sales_1 READ ONLY;

ALTER TABLESPACE sales_2 READ ONLY;
```

3. DATAPUMP_EXP_FULL_DATABASEロールを持つユーザーとしてデータ・ポンプ・エクスポート・ユーティリティを実行し、トランスポータブル・セット内の表領域を指定します。

```
SQL> HOST

$ expdp user_name dumpfile=expdat.dmp directory=data_pump_dir
        transport_tablespaces=sales_1,sales_2 logfile=tts_export.log

Password: password
```

トランスポータブル・オプションを使用することを指定するTRANSPORT_TABLESPACESを常に指定する必要があります。この例では、次の追加のデータ・ポンプ・パラメータを指定します。

- DUMPFILEパラメータでは、作成する構造情報エクスポート・ダンプ・ファイルの名前をexpdat.dmpと指定します。
- DIRECTORYパラメータでは、オペレーティング・システムまたはOracle Automatic Storage Managementのダンプ・ファイルの場所を示すディレクトリ・オブジェクトを指定します。DIRECTORYオブジェクトはデータ・ポンプを起動する前に作成し、ディレクトリに対するREADおよびWRITEオブジェクト権限をエクスポート・ユーティリティを実行するユーザーに付与する必要があります。

  非CDBで、ディレクトリ・オブジェクトDATA_PUMP_DIRが自動的に作成されます。このディレクトリへの読取りおよび書込みアクセス権がDBAロールに(したがって、ユーザーSYSおよびSYSTEMに)自動的に付与されます。

  ただし、ディレクトリ・オブジェクトDATA_PUMP_DIRは、PDBでは自動的には作成されません。このため、PDBにインポートする場合は、PDBにディレクトリ・オブジェクトを作成し、データ・ポンプを実行するときにそのディレクトリ・オブジェクトを指定します。
- LOGFILEパラメータでは、エクスポート・ユーティリティ用に作成するログ・ファイルを指定します。この例では、ログ・ファイルはダンプ・ファイルと同じディレクトリに作成されますが、ログ・ファイルの格納には他のディレクトリを指定できます。
- トリガーと索引は、デフォルトでエクスポート操作に含まれています。

表領域のトランスポート操作を厳密完結チェック付きで実行するには、次の例に示すように、TRANSPORT_FULL_CHECKパラメータを使用します。

```
expdp use_name dumpfile=expdat.dmp directory=data_pump_dir
      transport_tablespaces=sales_1,sales_2 transport_full_check=y
      logfile=tts_export.log
```

この場合は、データ・ポンプ・エクスポート・ユーティリティによって、トランスポータブル・セット内のオブジェクトとトランスポー

タブル・セット外のオブジェクトとの間に依存性がないことを検証します。トランスポートする表領域セットが自己完結していない場合、エクスポートは失敗し、トランスポータブル・セットが自己完結していないことがわかります。これらの違反を解決してから、このタスクを再度実行する必要があります。

> **注意:**
>
> この例では、データ・ポンプ・ユーティリティを使用してエクスポートするのは、表領域のデータ・ディクショナリの構造情報(メタデータ)のみです。実際のデータはアンロードされないため、この操作は大規模な表領域セットの場合でも比較的早く完了します。

4. expdpユーティリティは、次の例に示すように、ダンプ・ファイルおよびデータファイルの名前およびパスをコマンドラインに表示します。これらは、ターゲット・データベースへのトランスポートに必要なファイルです。また、エラーがないかログ・ファイルを確認します。

```
******************************************************************************
Dump file set for SYSTEM.SYS_EXPORT_TRANSPORTABLE_01 is:
  /u01/app/oracle/admin/salesdb/dpdump/expdat.dmp
******************************************************************************
Datafiles required for transportable tablespace SALES_1:
  /u01/app/oracle/oradata/salesdb/sales_101.dbf
Datafiles required for transportable tablespace SALES_2:
  /u01/app/oracle/oradata/salesdb/sales_201.dbf
```

5. データ・ポンプ・エクスポート操作が完了したら、expdpユーティリティを終了してSQL*Plusに戻ります。

```
$ EXIT
```

**関連項目:**

- CREATE DIRECTORYコマンドの詳細は、『*Oracle Database SQL言語リファレンス*』を参照してください
- DIRECTORYパラメータを省略する場合のデフォルト・ディレクトリの詳細は、『*Oracle Databaseユーティリティ*』を参照してください。
- データ・ポンプ・ユーティリティの使用方法は、『*Oracle Databaseユーティリティ*』を参照してください
- PDBの詳細は、『Oracle Multitenant管理者ガイド』を参照してください

**親トピック:** Oracle Databaseの移行

# タスク3: エクスポート・ダンプ・ファイルのトランスポート

ダンプ・ファイルを、DATA_PUMP_DIRディレクトリ・オブジェクトで指し示されているディレクトリ、または他の任意のディレクトリにトランスポートします。新しい場所はターゲット・データベースへアクセス可能であることが必要です。

ターゲット・データベースで、次の問合せを実行してDATA_PUMP_DIRの場所を確認します。

```
SELECT * FROM DBA_DIRECTORIES WHERE DIRECTORY_NAME = 'DATA_PUMP_DIR';

OWNER      DIRECTORY_NAME   DIRECTORY_PATH
---------- ---------------- ----------------------------------
SYS        DATA_PUMP_DIR    C:¥app¥orauser¥admin¥orawin¥dpdump¥
```

**親トピック:** Oracle Databaseの移行

# タスク4: 表領域セットのトランスポート

表領域のデータファイルをターゲット・データベースへアクセス可能なディレクトリにトランスポートします。

この例では、次のファイルをソース・データベースからターゲット・データベースに転送します。

- sales_101.dbf
- sales_201.dbf

ソース・プラットフォームとは異なるプラットフォームに表領域セットをトランスポートする場合は、ソースおよびターゲット・プラットフォームの両方でプラットフォーム間の表領域トランスポートがサポートされているかどうかを確認し、それぞれのプラットフォームのendiannessを判別します。両方のプラットフォームのendiannessが同じ場合、変換は必要ありません。同じでない場合は、ソース・データベースまたはターゲット・データベースでデータ変換を実行する必要があります。

sales_1およびsales_2を異なるプラットフォームにトランスポートする場合は、各プラットフォームで次の問合せを実行できます。問合せで行が返される場合、そのプラットフォームではプラットフォーム間の表領域トランスポートがサポートされています。

```
SELECT d.PLATFORM_NAME, ENDIAN_FORMAT
     FROM V$TRANSPORTABLE_PLATFORM tp, V$DATABASE d
     WHERE tp.PLATFORM_NAME = d.PLATFORM_NAME;
```

ソース・プラットフォームでの問合せの結果は次のとおりです。

```
PLATFORM_NAME                          ENDIAN_FORMAT
-------------------------------------- --------------
Solaris[tm] OE (32-bit)                Big
```

ターゲット・プラットフォームからの結果は、次のとおりです。

```
PLATFORM_NAME                          ENDIAN_FORMAT
-------------------------------------- --------------
Microsoft Windows IA (32-bit)          Little
```

この例では、endian形式が異なることがわかります。したがって、この場合、データベースをトランスポートするには変換が必要です。DBMS_FILE_TRANSFERパッケージのGET_FILEまたはPUT_FILEプロシージャを使用して、データファイルを転送します。これらのプロシージャを使用すると、データファイルがターゲット・プラットフォームのendian形式に自動的に変換されます。データファイルを、ターゲット・データベースの既存データファイルの場所にトランスポートします。UNIXおよびLinuxプラットフォームでは、この場所は通常、/u01/app/oracle/oradata/*dbname*/または+*DISKGROUP*/*dbname*/datafile/です。または、データファイルを変換するために使用することもできます。

注意:

- RMAN CONVERT コマンドを使用する場合、UNDO セグメントを含むデータファイルでは、異なる endian 形式間でのデータファイルの変換はサポートされていません。
- 表領域の endianness を変換する必要がない場合は、任意のファイル転送方法を使用してファイルを転送できます。

**親トピック:** Oracle Databaseの移行

# タスク5: (オプション)表領域を読取り/書込みモードに戻す

トランスポートした表領域をソース・データベースで再び読取り/書込みモードにします。

次の文で、sales_1およびsales_2表領域を読取り/書込みモードにします。

```
ALTER TABLESPACE sales_1 READ WRITE;
ALTER TABLESPACE sales_2 READ WRITE;
```

インポート・プロセスが成功したことを先に確認するために、このタスクを延期することができます。

**親トピック:** Oracle Databaseの移行

# タスク6: 表領域セットのインポート

トランスポータブル表領域に対する操作を完了するために、表領域セットをインポートします。

表領域セットのインポート手順は、次のとおりです。

1. DATAPUMP_IMP_FULL_DATABASEロールを持つユーザーとしてデータ・ポンプ・インポート・ユーティリティを実行し、表領域メタデータをインポートします。

```
impdp user_name dumpfile=expdat.dmp directory=data_pump_dir
   transport_datafiles=
   'c:¥app¥orauser¥oradata¥orawin¥sales_101.dbf',
   'c:¥app¥orauser¥oradata¥orawin¥sales_201.dbf'
   remap_schema=sales1:crm1  remap_schema=sales2:crm2
   logfile=tts_import.log

Password: password
```

この例では、次のデータ・ポンプ・パラメータを指定します。

- DUMPFILEパラメータでは、インポートされる表領域のメタデータが含まれるエクスポート・ファイルを指定します。
- DIRECTORYパラメータでは、エクスポート・ダンプ・ファイルの場所を識別するディレクトリ・オブジェクトを指定します。DIRECTORYオブジェクトはデータ・ポンプを実行する前に作成し、ディレクトリに対するREADおよびWRITEオブジェクト権限をインポート・ユーティリティを実行するユーザーに付与する必要があります。CREATE DIRECTORYコマンドの詳細は、『*Oracle Database SQL言語リファレンス*』を参照してください。

  非CDBで、ディレクトリ・オブジェクトDATA_PUMP_DIRがデータベースにより自動的に作成されます。このディレクトリへの読取りおよび書込みアクセス権がDBAロールに(したがって、ユーザーSYSおよびSYSTEMに)自動的に付与されます。

  ただし、ディレクトリ・オブジェクトDATA_PUMP_DIRは、PDBではデータベースにより自動的には作成されません。このため、PDBにインポートする場合は、PDBにディレクトリ・オブジェクトを作成し、データ・ポンプを実行するときにそのディレクトリ・オブジェクトを指定します。

  **関連項目:**

  - DIRECTORYパラメータを省略する場合のデフォルト・ディレクトリの詳細は、『*Oracle Databaseユーティリティ*』を参照してください。
  - PDBの詳細は、『Oracle Multitenant管理者ガイド』を参照してください

- TRANSPORT_DATAFILESパラメータによって、インポートする表領域が含まれるすべてのデータファイルを識別します。

  多くのデータファイルがある場合は、PARFILEパラメータで指定されたパラメータ・ファイルでTRANSPORT_DATAFILESパラメータを複数回指定できます。

- REMAP_SCHEMAパラメータによって、データベース・オブジェクトの所有権を変更します。REMAP_SCHEMAを指定しない場合、すべてのデータベース・オブジェクト(表や索引など)はソース・データベースと同じユーザー・スキーマ内で作成され、そのユーザーはターゲット・データベース内にすでに存在している必要があります。ユーザーが存在しない場合は、インポート・ユーティリティによってエラーが返されます。この例では、ソース・データベース内で

sales1が所有している表領域セット内のオブジェクトは、表領域セットをインポートした後のターゲット・データベース内ではcrm1の所有となります。同様に、ソース・データベース内でsales2が所有しているオブジェクトは、ターゲット・データベース内ではcrm2の所有となります。この例では、ターゲット・データベース内にユーザーsales1およびsales2は存在する必要はありませんが、ユーザーcrm1およびcrm2が存在する必要があります。

Oracle Database 12cリリース2 (12.2)以降では、Recovery Manager (RMAN)のRECOVERコマンドで、表を再マッピングしながら表を異なるスキーマに移動できます。詳細は、『Oracle Databaseバックアップおよびリカバリ・ユーザーズ・ガイド』を参照してください。

- LOGFILEパラメータでは、インポート・ユーティリティによって書き込まれるログ・ファイルのファイル名を指定します。この例では、ログ・ファイルの書込み先はダンプ・ファイルの読取り元と同じディレクトリですが、ログ・ファイルは別の場所に書き込むことができます。

この文が正常に実行されると、コピーするセット内のすべての表領域は読取り専用モードのままになります。インポート・ログ・ファイルを確認して、エラーが発生しなかったことを確認します。

多数のデータファイルを処理する場合は、データファイル・リストが文の行の制限を超えることがあるため、データファイル名のリストを文の行で指定することは煩雑です。このような場合には、インポート・パラメータ・ファイルを使用できます。たとえば、次のようにしてデータ・ポンプ・インポート・ユーティリティを実行できます。

```
impdp user_name parfile='par.f'
```

par.fパラメータ・ファイルには、次の情報が含まれています。

```
DUMPFILE=expdat.dmp
DIRECTORY=data_pump_dir
TRANSPORT_DATAFILES=
'C:¥app¥orauser¥oradata¥orawin¥sales_101.dbf',
'C:¥app¥orauser¥oradata¥orawin¥sales_201.dbf'
REMAP_SCHEMA=sales1:crm1  REMAP_SCHEMA=sales2:crm2
LOGFILE=tts_import.log
```

**関連項目:**

インポート・ユーティリティの使用方法は、『*Oracle Database*ユーティリティ』を参照してください。

2. 必要に応じて、ターゲット・データベースで表領域を読取り/書込みモードにします。

**親トピック:** Oracle Databaseの移行

# 移行後タスク

これらのタスクを完了して、使用するターゲットOracle Databaseを準備します。

1. データがターゲット・データベースにインポートされたかどうかを確認します。

   ソース・データベースとターゲット・データベースで次の問合せを実行し、エラーなしでデータが完全にエクスポートおよびインポートされたかどうかを確認します。

   データベースに存在するすべてのユーザーを表示するには、次のようにします。

```
SQL> SELECT count(*) FROM dba_users;
SQL> SELECT username, account_status FROM dba_users;
```

   データベース内のオブジェクトの合計数を表示するには、次のようにします。

```
SQL> SELECT count(*) FROM dba_objects;
SQL> SELECT count(*), owner FROM dba_objects group by owner;
```

   現在のユーザーが所有しているすべての表のリストを表示するには、次のようにします。

```
SQL> SELECT count(*) FROM user_tables;
SQL> SELECT count(*), tablespace_name FROM user_tables group by tablespace_name;
```

   表領域でオブジェクトによって占有された正確なサイズ(MB)を表示するには、次のようにします。

```
SELECT owner, segment_name, segment_type, partition_name, ROUND(bytes/(1024*1024),2) SIZE_MB,
tablespace_name
FROM DBA_SEGMENTS
WHERE SEGMENT_TYPE IN ('TABLE', 'TABLE PARTITION', 'TABLE SUBPARTITION',
'INDEX', 'INDEX PARTITION', 'INDEX SUBPARTITION', 'TEMPORARY', 'LOBINDEX', 'LOBSEGMENT', 'LOB
PARTITION')
--AND TABLESPACE_NAME LIKE 'COSTE%'
--AND SEGMENT_NAME LIKE 'P2010201%'
--AND partition_name LIKE 'P20100201%'
--AND segment_type = 'TABLE'
--AND OWNER = 'TARGET_POC'
--AND ROUND(bytes/(1024*1024),2) > 1000
ORDER BY bytes DESC;
```

   占有された合計領域(MB)を表示するには、次のようにします。

```
SELECT tablespace_name, owner, segment_type "Object Type",
       COUNT(owner) "Number of Objects",
       ROUND(SUM(bytes) / 1024 / 1024, 2) "Total Size in MB"
FROM   sys.dba_segments
WHERE  tablespace_name IN ('MPIS')
GROUP BY tablespace_name, owner, segment_type
ORDER BY tablespace_name, owner, segment_type;
```

   データベースのサイズを表示するには、次のようにします。

```
SQL> SELECT a.data_size+b.temp_size+c.redo_size+d.controlfile_size "total_size in MB" FROM
( select
        sum(bytes)/1024/1024 data_size
        FROM dba_data_files ) a,
        (select nvl(sum(bytes),0)/1024/1024 temp_size
        FROM dba_temp_files) b,
        (select sum(bytes)/1024/1024 redo_size
```

```
        FROM sys.v_$log) c,
        (select sum(BLOCK_SIZE*FILE_SIZE_BLKS)/1024/1024 controlfile_size from v$controlfile)
d;
```

2. トランスポートされた表領域を宛先でREAD WRITEモードに切り替えます。

```
SQL> ALTER TABLESPACE tablespace name READ WRITE;
```

3. 表領域をソースでREAD WRITEモードに戻します。

```
SQL> ALTER TABLESPACE tablespace name READ WRITE;
```

4. アプリケーションを宛先データベースにリダイレクトします。

   新しい宛先データベースで、適切なデータベース・サービスまたはネットワーク接続(あるいはその両方)を作成して起動します。

5. ステージング・ディレクトリをクリーン・アップします。

   不要なファイルをソース・ホストおよび宛先ホストから削除します。

**親トピック:** Oracle Databaseの移行