ORACLE

フル・トランスポータブル・エクスポート/インポート

Oracle Data Pumpのフル・トランスポータブル機能を使用した
Oracleデータベースの移行

2021年8月 | バージョン2.00

本書の目的

本書では、Oracle Data Pumpのフル・トランスポータブル・エクスポート/インポートの概要を説明します。

免責事項

本文書には、ソフトウェアや印刷物など、いかなる形式のものも含め、オラクルの独占的な所有物である占有情報が含まれます。この機密文書へのアクセスと使用は、締結および遵守に同意したOracle Software License and Service Agreementの諸条件に従うものとします。本文書と本文書に含まれる情報は、オラクルの事前の書面による同意なしに、公開、複製、再作成、またはオラクルの外部に配布することはできません。本文書は、ライセンス契約の一部ではありません。また、オラクル、オラクルの子会社または関連会社との契約に組み込むことはできません。

本書は情報提供のみを目的としており、記載した製品機能の実装およびアップグレードの計画を支援することのみを意図しています。マテリアルやコード、機能の提供をコミットメント（確約）するものではなく、購買を決定する際の判断材料にするものでもありません。本書に記載されている機能の開発、リリースおよび時期については、オラクルの裁量により決定されます。

製品アーキテクチャの性質上、コードが大幅に不安定化するリスクなしに、本書に記載されているすべての機能を安全に含めることができない場合があります。

目次

はじめに

フル・トランスポータブル・エクスポート/インポートは、以前のバージョンのOracleデータベースから非PDBデータベースおよびマルチテナント・データベースへの移行に使用でき、これまでになく高速かつ容易で効率的なOracleデータベース移行を可能にします。フル・トランスポータブル・エクスポートは、Oracle Database 11g Release 2（11.2.0.3）以降で使用でき、フル・トランスポータブル・インポートは、Oracle Database 12c（12.1.0.1）以降で利用できます。

この技術概要では、Oracle Data Pumpのフル・トランスポータブル・エクスポート/インポート機能について説明します。まず、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メリットの概要を述べ、その後この機能の仕組みについて説明し、この機能を使用する場合の構文とプロセス・フローを示すフル・トランスポータブル・エクスポート/インポートの詳細な例を示します。本書では、プラガブル・データベース環境でのフル・トランスポータブル・エクスポート/インポートの使用に焦点を合わせていますが、この機能は、非PDBデータベースへの移行でも使用できます。

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メリット

フル・トランスポータブル・エクスポート/インポートは、Data Pumpエクスポート/インポートのユーザーが使い慣れている操作性と、トランスポータブル表領域によって実現される物理的な移行スピードを兼ね備えています。これらの移行手法の要点をまとめた上で、フル・トランスポータブル・エクスポート/インポートと比較対照すると、フル・トランスポータブル・エクスポート/インポートがより高速かつシンプルで、効率的な理由を理解できるでしょう。

Oracle Data Pumpのエクスポートおよびインポートは、Oracle Database 10gで導入され、大量のデータを処理するように設計されています。これまでに、1時間あたり10 TBを軽く超えるデータ転送速度を実現してきました。パラレル・ワーカー・プロセス、複数ダンプ・ファイルなどの手法を採用して、移行対象データに最適なアクセス方法を選択します。Oracle Data Pumpのコマンドラインは、DBAにとって違和感がなく、全般的に使いやすいと見なされています。論理的な移行手法であるData Pumpは、最大限の柔軟性を備えています。データおよびメタデータの圧縮、暗号化、IOTからヒープ表への変換、オブジェクトの包含/除外、キャラクタ・セットの変更、表領域とスキーマの再マッピング、BasicFileからSecureFileへの透過的な変換などの機能を含み、ソース・データベース/ターゲット・データベース間のデータベース・リンクを介してエクスポートとインポートを1つの操作に集約することも可能です。また、非CDBデータベースからマルチテナントにも簡単に移行できます。

場合によっては、トランスポータブル表領域を使用する物理的な移行方法の方が、ダンプ・ファイルを使用するエクスポート/インポートよりも高速になるケースがあります。たとえば、再作成なしの大量のデータ移行や大量の索引の移行では、トランスポータブル表領域の方が高速になり得ます。

トランスポータブル表領域は通常、データベース間でユーザー・データとアプリケーション・データを移動する方法としては最速です。これは、ソース・データベースからターゲット・データベースに表領域データファイルがまとめて移動されるからです。一般には、データファイル全体を移動する方が、データの個々の行あるいはブロックを論理的にエクスポートしてインポートするよりもはるかに高速です。ただし、従来のトランスポータブル表領域では、移行先データベースでこれらの表領域データファイルを使用するのに必要なユーザーとアプリケーションのメタデータを移動するため、非常に複雑な一連の手順が必要になることがあります。そのため、トランスポータブル表領域を使用する移行方法は、非常に高速ではあってもより複雑なものと特徴付けることができます。

Oracle Database 12c以降のフル・トランスポータブル・エクスポート/インポートは、論理的な移行方法と物理的な移行方法の両方の長所を兼ね備えています。フル・トランスポータブル・エクスポート/インポートのコマンドラインは、Oracle Data Pumpのコマンドラインです。フル・トランスポータブル・エクスポート/インポートでは、データベース・リンク経由でメタデータを移動できる機能など、Data Pumpオプションの利点を活用して、1つのインポート・コマンドでデータベース全体を移行することができます。
同時に、フル・トランスポータブル・エクスポート/インポートでは、トランスポータブル表領域のメカニズムを使用してユーザー・データとアプリケーション・データを移動します。これにより、データが非常に大量であっても、非常に高速に移行処理が行われます。もっとも重要なのは、フル・トランスポータブル・エクスポート/インポートでは、従来のトランスポータブル表領域による処理で必要とされる一連の複雑な手順を行うことなく、データベース移行に必要なシステム、ユーザー、およびアプリケーションのすべてのメタデータが移動されることです。

このようにフル・トランスポータブル・エクスポート/インポートには、Oracle Data Pumpの使いやすさとトランスポータブル表領域のパフォーマンスが組み合わされており、データベース移行を迅速かつ簡単に実行できることが特長となっています。

フル・トランスポータブル・エクスポート/インポートによるプラガブル・データベースのサポート

フル・トランスポータブル・エクスポート/インポートは、マルチテナントとプラガブル・データベース（PDB）をサポートしているので、非CDBデータベースからPDBへ、またPDBから非CDBへの双方向の移行に使用できます。さらに、あるPDBを別のPDBに移行することもできます。

非CDBからプラガブル・データベースへの移行プロセスの概要

1. CREATE PLUGGABLE DATABASEコマンドを使用して、移行先CDB内に新しいPDBを作成します。
2. ソース・データベースのユーザー表領域とアプリケーション表領域をREAD ONLYに設定します。
3. 表領域データファイルを移行先にコピーします。
4. DATAPUMP_IMP_FULL_DATABASE権限を持つアカウントでログインして、次のいずれかの移行方法を使用します。
   - コマンドラインでexpdpクライアントを使用し、FULL=Y TRANSPORTABLE=ALWAYSパラメータを指定して、ソース・データベースからエクスポート/インポートする。次に、impdpクライアントを使用して、ターゲットのPDBにインポートする。
   - impdpを使用して、データベース・リンク経由でソースからターゲットにインポートする。
5. 移行後のデータベース検証とアプリケーション・テストを実行します。

フル・トランスポータブル・エクスポート/インポートの内部の仕組み

このセクションでは、フル・トランスポータブル・エクスポート/インポートで使用される仕組みについて簡単に説明します。これにより、フル・トランスポータブル・エクスポート/インポートの動作の理解が深まるでしょう。フル・トランスポータブルの動作を理解することで、どのようにして最適な利便性とパフォーマンスを両立させているのかという問いに答えられるようになります。

フル・トランスポータブル・エクスポート/インポートの概念

フル・トランスポータブル・エクスポート/インポートについて理解するには、従来の方法でデータを移動する場合とトランスポータブルを使用してデータを移動する場合の違い、および管理表領域とユーザー表領域の違いを理解している必要があります。

従来の方法を使用してデータを移動する場合、Oracle Data Pumpでは、外部表かダイレクト・パス・アンロードのいずれかを使用してデータを抽出します。どちらのアクセス方法を選択するかは、アップロードされるデータの構造とタイプによって決まりますが、どちらの方法を使用するとしても、データの論理サブセットがOracleデータベースから効率よく抽出されます。

それとは対照的に、トランスポータブルによるデータと索引の移動には、1つ以上の表領域データファイルを物理的に移動することが含まれます。表領域データファイル内部のデータ・セグメントは、個別に読み取られません。代わりに、エクスポート操作により、各データファイル内に格納されているオブジェクトについて記述したメタデータが抽出され、各ファイルが単一エンティティとして移動されます。トランスポータブル表領域を使用して大量のデータを移動する方が、従来式のデータ移動方法よりも速く処理できますが、これは、データまたは索引エントリの個々の行を解釈および抽出する必要がないからです。個々の表またはパーティションをトランスポータブル方式で移動することは可能ですが、これらの場合には、表領域データファイルも全体が移動されることになります。

従来の方法とトランスポータブルによるデータ移動の違いについて理解していることは、管理表領域とユーザー表領域の違いについて考慮する場合に役立ちます。フル・トランスポータブル・エクスポートを目的とする場合、管理表領域は、SYSTEM、SYSAUX、TEMP、およびUNDOなどのオラクルが提供する表領域です。これらの表領域には、Oracleデータベース・コア機能のプロシージャ、パッケージ、シード・データ、およびOracle Spatial、Oracle Text、OLAP、JAVAVM、およびOracle XML Databaseなどのオラクルによって提供されるデータベース・コンポーネントが存在します。これとは対照的に、ユーザー表領域は、データベースのユーザーまたはアプリケーションによって定義される表領域です。ユーザー表領域には、ユーザー・データ、アプリケーション・データ、およびデータベースのユーザーによって定義される他のすべての情報が保存されます。

フル・トランスポータブル・エクスポートの最初のステップは、エクスポート先データベースを作成することです。新規作成されるデータベースにはターゲット環境に適した管理表領域のセットが組み込まれており、オラクルが提供するコンポーネントとパッケージを備えています。ユーザー表領域はこの移行先データベース内には存在せず、トランスポータブル・インポートの一環として作成されます。

Oracle Database 12c以降、オラクルが提供するオブジェクトがOracle Data Pumpによってエクスポートまたはインポートされることはなくなりました。ソース・データベースの管理表領域に、ユーザーやアプリケーションが定義したデータ/メタデータが含まれる場合、フル・トランスポータブル・エクスポートによる移行中に、従来のデータ移動手法を使用してこれらが抽出されます。

一方、ユーザー表領域は、フル・エクスポートによってソースから移行先のシステムにユーザーとアプリケーションのすべてのデータおよびメタデータが移行されることを前提に、全表領域データファイルとして移行先データベースに移動されます。ユーザー表領域はこのようにトランスポータブルな方法で移動されるため、ユーザー・データ移行時に最大のパフォーマンスが得られます。

注：*1*つのデータベース・オブジェクトがユーザー表領域と管理表領域の両方にわたって格納されている場合（パーティション表など）、フル・トランスポータブル・エクスポート/インポートに特有の事項を考慮する必要が生じます。この方法でオブジェクトを格納することは、一般には勧められませんが、そうすることは可能です。管理表領域とユーザー表領域の両方にストレージがあるオブジェクトが存在する場合は、データを転送する前にそのオブジェクトを再定義するか、または従来の*Data Pump*エクスポート/インポートを使用することができます。この技術概要の後半部分に挙げる例には、フル・トランスポータブル・エクスポートを開始する前にこの状態を検出する方法が示されています。

フル・トランスポータブル・エクスポート

フル・トランスポータブル・エクスポート/インポートの優れたユーザビリティの秘訣は、ユーザーとアプリケーションが定義したすべてのオブジェクトのメタデータが抽出されることです。Oracleコンポーネントが内部で使用するコールアウト・メカニズムとAPIが提供されているため、インポート・プロセス中にデータベースのフル・コピーを作成するために必要になるメタデータがすべて移行されます。

管理表領域に存在するすべてのユーザーおよびアプリケーション・オブジェクトは、メタデータ（表定義など）とデータ（その表の行など）の両方が従来式の方法でアンロードされます。それとは対照的に、ユーザー表領域に格納されているオブジェクトは、Data Pumpによってメタデータのみがアンロードされ、それらのオブジェクトのデータは表領域データファイルのコピーによりトランスポータブルな方法で移動されます。

フル・トランスポータブル・エクスポートの起動

Oracle Data Pumpのエクスポート・クライアント（expdp）は、フル・トランスポータブル・エクスポート用のコマンドライン・インタフェースです。フル・トランスポータブル・エクスポートは、パラメータ・ファイルまたはコマンドラインにおいてTRANSPORTABLE=ALWAYSとFULL=Yの2つのパラメータを指定することによって開始できます。これらのパラメータ値により、従来のエクスポート方法ではなくフル・トランスポータブル・エクスポートを使用するようData Pumpに指示します。

注：ソース・データベースのCOMPATIBLEデータベース初期化パラメータの値が最低でも12.0に設定されていない場合には、特別な考慮事項があります。これは、たとえばOracle Database 11g Release 2（11.2.0.3）からフル・トランスポータブル・エクスポートを実行する場合が該当します。この場合、Data PumpのパラメータをVERSION=12.0またはそれ以上に指定する必要があります。これは、エクスポートの結果がOracle Database 12c Release 1（12.1）以降のデータベースにインポートされることを示しています。

フル・トランスポータブル・インポート

従来式のData Pumpインポートの場合と同様に、フル・トランスポータブル・インポートによってダンプ・ファイルをインポートしたり、データベース・リンク経由でソース・データベースから移行先データベースに直接インポートしたりすることができます。ダンプ・ファイルを使用せずにインポート可能なため、フル・トランスポータブル・インポートは、データベースの移行処理に欠かせないツールとなっています。

フル・トランスポータブル・インポートの起動

ユーザーがダンプ・ファイルをインポートしようとすると、Oracle Data Pumpは、そのダンプ・ファイルが従来のエクスポートとフル・トランスポータブル・エクスポートのどちらによって生成されたものなのかを判別することができます。したがって、ファイルベースのフル・トランスポータブル・インポートで必要なのは、TRANSPORT_DATAFILES=<datafile_names>パラメータを使用して、トランスポートされる1）ダンプ・ファイルの名前と、2）読取り専用のユーザー表領域データファイルのコピー・リストを指定することだけです。このパラメータでは、ファイルのカンマ区切りリストを使用することも、データファイルごとに複数回指定することもできます。

注：オラクルは、ファイル名を囲む引用符などの構文の問題を回避するため、パラメータ・ファイルを使用することを推奨します。

ダンプ・ファイルを使用せずにソース・データベースからターゲット・データベースに直接移行する場合は、追加のNETWORK_LINK=<dblink_name>パラメータを指定して、フル・トランスポータブル・モードで処理を開始する必要があります。これは、ネットワーク・モードのインポートの場合は、実際には、ソース・データベースからのエクスポートおよびターゲット・データベースへのインポートが一度の処理として実行されるからです。ネットワークベースのフル・トランスポータブル・インポートでは、以下のパラメータが必要とされます。

- FULL=Y, TRANSPORTABLE=ALWAYS：フル・トランスポータブル・エクスポートで指定する
- TRANSPORT_DATAFILES=<datafile_names>：ファイルベースのフル・トランスポータブル・インポートで使用する

- VERSION=12以上：ソース・データベース・バージョンのCOMPATIBLE設定が12.0未満の場合。フル・トランスポータブル・インポートの場合、ソース・データベースは、Oracle Database 11g Release 2（11.2.0.3）以降でなければならない
- NETWORK_LINK=<dblink_name>：データおよびメタデータがソース・データベースから送信されるときに経由するデータベース・リンクを指定する

注：インポート・ジョブが正常に完了すると、すべてのユーザー表領域が自動的に読取り/書込み状態に設定されます。元のソース・データファイルが移行先データベースに接続されている場合、ソース・データベースからは使用できなくなります。これらのデータファイルは移行先データベースに属します。

エンディアン変換

ソースとターゲットの両方のプラットフォームで同じエンディアン形式が使用されている場合、データファイルは同じプラットフォーム上に存在するかのような感覚で転送できます。ソース・プラットフォームとターゲット・プラットフォームのエンディアンネスが異なる場合は、RMAN CONVERTコマンドを使用するか、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、データファイルをターゲット・プラットフォームでの形式に変換する必要があります。

注：暗号化されている表領域は、エンディアンネスが異なるプラットフォームに転送できません。

例：フル・トランスポータブルを使用した非CDBからCDBへの移動

この例では、フル・トランスポータブルを使用して、Oracle Solaris x86サーバーからOracle Linux上で稼働するCDB内のOracle Database 12cのPDBに、Oracle Database 11g Release 2（11.2.0.3）データベースを移行します。この例ではダンプ・ファイルを使用しているため、ソース・データベースからのフル・トランスポータブル・エクスポートに続いて、移行先データベースでフル・トランスポータブル・インポートを実行する必要があります。ソース・データベースには、hr_1とhr_2の2つのユーザー表領域があります。この例の表領域HR_1は、暗号化されています。

表1：ソース・データベースの表領域

表領域名	暗号化の有無	データファイル名
HR_1	あり	/u01/app/oracle/oradata/hr_db/hr_101.dbf
HR_2	なし	/u01/app/oracle/oradata/hr_db/hr_201.dbf

この例では、ターゲット・データベースのOracle SIDがHR_PDBであり、ターゲットPDBのデータファイルが移行先サーバーの/u01/app/oracle/oradata/hr_pdb/ディレクトリに格納されることを前提としています。

注：透過的データ暗号化（TDE）の設定および管理に使用されるコマンドは、Oracle Database 11g Release 2より後に変更されています。使用するデータベースへのTDEの実装について詳しくは、『Oracle Database Advanced Securityガイド』を参照してください。

手順1：両方のプラットフォームのエンディアンネスを確認する

プラットフォームのエンディアンネスを確認するには、各プラットフォームで次の問合せを実行します。

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

この場合は、Oracle Solaris x86とOracle Enterprise Linuxの両方でリトル・エンディアン形式が使用されているため、エンディアン変換は不要です。

手順2：転送される表領域のセットが自己完結型であることを確認する

表領域のセットを転送する前に、転送される表領域に格納されているオブジェクトと転送されない表領域に格納されているオブジェクトの間に、論理的または物理的な依存関係が存在しないことを検証する必要があります。たとえば、1つのオブジェクトがユーザー表領域と管理表領域の両方に格納されている状況があれば、検出する必要があります。これは完結チェックと呼ばれます。

転送後に参照整合性制約が有効になることの検証を含め、表領域hr_1およびhr_2が自己完結型かどうかを判別するには、ソース・データベースで次のコマンドを実行します。

```
SQL> EXECUTE DBMS_TTS.TRANSPORT_SET_CHECK('hr_1,hr_2', TRUE);
```

フル・トランスポータブル・エクスポート/インポートでこのチェック機能を実行する場合は、データベース内のすべてのユーザー表領域を含める必要があります。

PL/SQLプロシージャを起動した後は、次のようにTRANSPORT_SET_VIOLATIONSビューから選択することによって、すべての違反を表示できます。

```
SQL> SELECT * FROM TRANSPORT_SET_VIOLATIONS;
```

表領域のセットが自己完結型になっている場合、このビューは空になります。いずれかの違反が表示された場合は、フル・トランスポータブルの処理を続行する前に、これらの問題に対処する必要があります。

手順3：ソース・データベースでディレクトリ・オブジェクトを作成する

フル・トランスポータブル・エクスポートを開始する前に、ダンプ・ファイルの書込み先にディレクトリ・オブジェクトを作成します。このディレクトリへの読取り/書込みアクセス権は、DBAロールとSYSおよびSYSTEMユーザーに自動的に付与されます。

```
SQL> CREATE DIRECTORY dp_dir AS '/u01/app/datafiles';
```

このディレクトリ・オブジェクトは、フル・トランスポータブル・エクスポートの終了後にデータベースから削除できます。

手順4：表領域hr_1およびhr_2を読取り専用モードにする

転送される表領域は、エクスポートの期間中は読取り専用モードになっている必要があります。この場合は、ソース・データベースで2つのコマンドを発行する必要があります。

```
SQL> ALTER TABLESPACE hr_1 READ ONLY;
SQL> ALTER TABLESPACE hr_2 READ ONLY;
```

表領域は、フル・トランスポータブル・エクスポートが終了し、表領域データファイルが移行先システムにコピーされたら、読取り/書込みステータスに戻すことができます。

手順5：ソース・データベースでフル・トランスポータブル・エクスポートを起動する

DATAPUMP_EXP_FULL_DATABASEロールのユーザーとして、Data Pumpエクスポート・ユーティリティを起動します。

```
$ expdp system/manager full=y transportable=always version=12 \
        directory=dp_dir dumpfile=full_tts.dmp \
        metrics=y exclude=statistics \
        encryption_password=secret123word456 \
        logfile=full_tts_export.log
```

注：Data Pumpエクスポートで、METRICS=Yと統計情報に対するEXCLUDEを指定することは一般に推奨されています。Oracle Database 12g以降では、LOGTIME=ALLの使用も推奨されています。

ソース・データベースがOracle Database 11g Release 2（11.2.0.3）になっているため、VERSION=12パラメータは必須です。これは、expdpコマンドによって、現行バージョンより大きいバージョン番号が許可される唯一の場合です。ソース・データベースがOracle Database 12c以上であり、COMPATIBLE=12.0以上に指定されている場合、VERSIONパラメータは不要です。

エクスポート・コマンドが完了すると、エクスポート・ログ・ファイルには、ターゲットに移動する必要がある表領域データファイルのすべてが一覧表示されます。

手順6：ソースからターゲットに表領域データファイルとエクスポート・ダンプ・ファイルを転送する

このデータベースを新しいサーバーに移行しようとしているため、表領域データファイルとエクスポート・ダンプ・ファイルは、移行先データベースからアクセス可能な場所にコピーする必要があります。たとえば、移行先システムで次のコマンドを発行できます。

```
$ cd /u01/app/oracle/oradata/hr_pdb/

$ cp /net/<source-server>/u01/app/oracle/oradata/hr_db/hr_101.dbf .

$ cp /net/<source-server>/u01/app/oracle/oradata/hr_db/hr_201.dbf .

$ cp /net/<source-server>/u01/app/datafiles/full_tts.dmp .
```

手順7：移行先データベースでディレクトリ・オブジェクトを作成する

Data Pumpのダンプ・ファイルをHR_PDBのoradataディレクトリにコピーしたため、このインポートでその同じディレクトリを指すようにディレクトリ・オブジェクトを作成します。このディレクトリ・オブジェクトは、PDBコンテナに接続されているユーザーが作成する必要があります。

```
SQL> CREATE DIRECTORY dp_dir AS '/u01/app/oracle/oradata/hr_pdb';
SQL> GRANT read, write on directory dp_dir to system;
```

このディレクトリ・オブジェクトは、フル・トランスポータブル・インポートの終了後にデータベースから削除できます。

手順8：移行先データベースでフル・トランスポータブル・インポートを起動する

Data Pumpインポート・ユーティリティをDATAPUMP_IMP_FULL_DATABASEロールのユーザーとして起動します。

```
$ impdp system/manager@hr_pdb directory=dp_dir \
        dumpfile=full_tts.dmp logfile=full_tts_imp.log \
        metrics=y \
        logtime=all \
        encryption_password=secret123word456 \
        transport_datafiles='/u01/app/oracle/oradata/hr_pdb/hr_101.dbf',\
                            '/u01/app/oracle/oradata/hr_pdb/hr_201.dbf'
```

注：impdpコマンドの接続文字列で、PDBのサービス名を明示的に指定する必要があります。

この例のコマンドラインでは、いくつかのパラメータが指定されていますが、ほとんどの場合、コマンドラインでの空白文字と引用符に関する問題を回避するため、Data Pumpのパラメータ・ファイルを使用することを推奨します。

この文が正常に実行された後、ユーザー表領域は移行先データベースで自動的に読取り/書込みモードになります。インポート・ログ・ファイルを調べて予期せぬエラーが発生していないことを確認し、通常の移行後の検証およびテストを実施します。

手順9：（オプション）ソース・データベースでユーザー表領域を読取り/書込みモードに戻す

フル・トランスポータブル・エクスポートが終了したら、必要に応じて、ソース・データベースにおいて、ユーザー定義表領域を読取り/書込みモードに戻すことができます。

```
SQL> ALTER TABLESPACE hr_101 READ WRITE;
SQL> ALTER TABLESPACE hr_201 READ WRITE;
```

移行先データベースでは、フル・トランスポータブル・インポートの完了時に、すべての表領域が自動的に読取り/書込みモードに設定されます。

結論

フル・トランスポータブル・エクスポート/インポートにより、データベースの移行プロセスが大幅に簡素化されて迅速になります。フル・トランスポータブル・エクスポート/インポートでは、Oracle Data Pumpの使いやすさとトランスポータブル表領域のパフォーマンスが組み合わされており、ソース・データベースがOracle Database 11g Release 2（11.2.03）以降であれば、一度の処理でOracle Databaseにアップグレードまたは移行することができます。また、フル・トランスポータブル・エクスポート/インポートは、プラガブル・データベースへの移行のための価値あるツールであり、マルチテナント・アーキテクチャへの移行に付随するコストの節約とスケール・メリットの恩恵に与ることができます。

付録：フル・トランスポータブル・エクスポート/インポートの制限事項

フル・トランスポータブル・エクスポート/インポートにより、従来のトランスポータブル表領域のユーザビリティが大幅に強化されていますが、データ転送には制限事項がいくつかあり、それらに留意する必要があります。

- 従来のトランスポータブル表領域と同様、フル・トランスポータブル・エクスポート/インポートでは、エクスポート中は、移行されるユーザー表領域が読取り専用モードになっている必要があります。この制限が望ましくない場合（特に、本番環境で使用する前にこの方法をテストする場合）は、Oracle RMANの機能を使用して、バックアップから表領域を転送することができます。バックアップからの表領域の転送について詳しくは、『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。
- オブジェクトのストレージがユーザー表領域と管理表領域の両方にわたっている場合、そのオブジェクトをエクスポート対象として選択することはできません。すべてのストレージ・セグメントが、完全にユーザー定義のトランスポータブル表領域に含まれるか、または完全に非トランスポータブルの管理表領域に含まれている必要があります。ストレージが複数の表領域に存在していると、トランスポータブル・モードにすることができなくなり、エクスポートに失敗します。この場合は、従来の方法でエクスポートを実行するか、この基準を満たすようにオブジェクトのストレージを再定義する必要があります。
- ソースとターゲットのプラットフォームでエンディアンネスが異なる場合は、転送され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。
- データの変換には、RMAN CONVERT DATAFILEコマンドを使用することができます。
- あるいは、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送信することにより、データファイルをターゲット・プラットフォームの形式に変換できます。
- 暗号化されている表領域をエンディアンネスが異なるプラットフォームに転送することはできません。
- 暗号化されている表領域を同じエンディアンネスのプラットフォームに転送するには、エクスポート時にexpdpのENCRYPTION_PASSWORDパラメータを使用します。インポート時には、impdpのENCRYPTION_PASSWORDパラメータに同じ値を指定します。
- XDBリポジトリは、フル・トランスポータブル・エクスポート/インポートではサポートされていません。ただし、ユーザー定義のXMLスキーマは、すべての形式のエクスポート/インポート（Data Pumpエクスポート/インポート、トランスポータブル表領域、およびフル・トランスポータブル・エクスポート/インポート）でサポートされています。
- 自動ワークロード・リポジトリ（AWR）は、フル・トランスポータブル・エクスポート/インポートではサポートされていません。Oracleデータベース間でAWRデータを移動するには、awrextr.sqlおよびawrload.sqlスクリプトを使用してください。
- ソース・データベースとターゲット・データベースでは、互換性のあるデータベース文字セットを使用する必要があります。この点について、およびデータ転送における一般的な制限事項について詳しくは、『Oracle Database管理者ガイド』を参照してください。

CONNECT WITH US

+1.800.ORACLE1までご連絡いただくか、oracle.comをご覧ください。

北米以外の地域では、oracle.com/contactで最寄りの営業所をご確認いただけます。

blogs.oracle.com facebook.com/oracle twitter.com/oracle

Oracle Databaseのフル・トランスポータブル・エクスポート/インポート
2021年8月
著者：Row Swonger
共著者：William Beauregard