



EDB Postgres Migration Toolkit Guide

リリース **53.0.0**

EnterpriseDB

2020 年 01 月 05 日

目次

第 1 章	新機能	3
第 2 章	サポートされているバージョン	5
第 3 章	移行方法	7
3.1	移行プロセス	7
3.2	アプリケーションを Postgres に接続する	9
第 4 章	機能概要	11
第 5 章	移行ツールキットのインストール	15
5.1	RPM パッケージを使用して Migration Toolkit をインストールする	15
5.2	Stack Builder を使用して移行ツールキットをインストールする	18
5.3	ソース固有のドライバのインストール	25
第 6 章	toolkit.properties ファイルの作成	27
6.1	Advanced ServerURL の定義	29
6.2	PostgreSQL URL の定義	31
6.3	Oracle URL の定義	33
6.4	MySQL URL の定義	35
6.5	Sybase URL の定義	36
6.6	SQL Server URL の定義	37
第 7 章	移行ツールキットの呼び出し	39
7.1	Oracle からのスキーマの移行	41
7.2	Oracle 以外のソースデータベースからの移行	42
第 8 章	Migration Toolkit コマンドオプション	45
8.1	オフライン移行オプション	46
8.1.1	オフライン移行スクリプトの実行	47
8.2	インポートオプション	48
8.3	スキーマ作成オプション	49
8.4	スキーマオブジェクト選択オプション	50
8.5	移行オプション	53

8.6	Oracle 固有のオプション	57
8.7	その他のオプション	62
8.8	例	63
第 9 章	移行エラー	67
9.1	接続エラー	67
9.1.1	無効なユーザー名/パスワード	67
9.1.2	接続が拒否されました:致命的:パスワード	68
9.1.3	例外:ORA-28000:アカウントはロックされています	68
9.1.4	例外:oracle.jdbc.driver.OracleDriver	68
9.1.5	I/O 例外:ネットワークアダプターは接続を確立できませんでした	69
9.1.6	例外:“ターゲット”データベースに指定された URL は無効です	69
9.2	移行エラー	70
9.2.1	エラー:予想される最後の列の後の追加データ	70
9.2.2	テーブルへのデータ読み込みエラー:* TABLE_NAME *	70
9.2.3	制約 CONS_NAME_FK の作成エラー	70
9.2.4	テーブルへのデータの読み込みエラー	71
第 10 章	エラーコード	75
10.1	エラーコードの概要	76
10.1.1	クラス 02-警告	76
10.1.2	クラス 10-無効なユーザー入力	76
10.1.3	クラス 11-構成の問題	78
10.1.4	クラス 12-不十分な特権	79
10.1.5	クラス 13-サポートされていない機能	80
10.1.6	クラス 14-欠落オブジェクト	82
10.1.7	クラス 15-スキーマの移行	82
10.1.8	クラス 16-手続き言語の移行	83
10.1.9	クラス 17-データの読み込み	84
第 11 章	サポートされていない機能	85
11.1	サポートされていない Postgres 機能	86
11.2	よくある質問	87
索引		89

EDB Postgres™ Migration Toolkit (Migration Toolkit) は、移行プロセスをきめ細かく制御できる強力なコマンドラインツールです。 Migration Toolkit は、データベースオブジェクトとデータの EDB Postgres™ Advanced Server (Advanced Server) または PostgreSQL® データベースへの以下からの移行を容易にします。

- Oracle®
- MySQL™
- Microsoft® SQL Server®

Migration Toolkit を使用して、データベースオブジェクトとデータを Sybase® Adaptive Server® Enterprise から Advanced Server に、または Advanced Server と PostgreSQL の間で移行することもできます。

Migration Server は、Advanced Server インストーラーまたは Stack Builder を使用してインストールできます。 Stack Builder は Advanced Server と PostgreSQL のワンクリックインストーラーの両方で配布され、EnterpriseDB Web サイトから入手できます:

<https://www.enterprisedb.com/advanced-downloads>

このガイドでは、移行プロセスに関連する手順の概要と、Migration Toolkit のインストールおよび使用手順について説明します。また、一般的な移行の問題の解決策、およびサポートされていない機能とその潜在的な回避策の詳細も含まれています。

ご注意ください:

このガイドでは、Postgres という用語を使用して、* Advanced Server または PostgreSQL のインストールのいずれかを指します。

このガイドでは、Stack Builder という用語を使用して、* StackBuilder Plus (Advanced Server で配布) または Stack Builder (EnterpriseDB の PostgreSQL ワンクリックインストーラーで配布) のいずれかを指します。

第 1 章

新機能

Migration Toolkit 53.0.0 に次の機能が追加されました。

- Migration Toolkit は、ソースおよびターゲットデータベースとして PostgreSQL バージョン 12 および Advanced Server バージョン 12 をサポートするようになりました。
- Migration Toolkit は、次の Oracle 機能の Advanced Server バージョン 12 への移行をサポートしています。
 - 複合トリガー
 - SYS_GUID
 - 中央値
 - LISTAGG
 - CAST_MULTISSET
 - Interval Partition
 - SELECT UNIQUE

詳細については、“EDB Postgres Advanced Server v12 の機能”を参照してください<https://www.enterprisedb.com/edb-docs/static/docs/epas/12/EPAS_Release_Notes_v12_Beta.pdf>’_ _。

第 2 章

サポートされているバージョン

以下は、Migration Toolkit で使用できるデータベース製品のバージョンです。

- PostgreSQL バージョン 9.5、9.6、10、11、および 12
- Advanced Server バージョン 9.5、9.6、10、11、および 12
- Oracle 10g リリース 2
- Oracle 11g リリース 2
- Oracle 12c リリース 1
- SQL Server 2008
- SQL Server 2012
- SQL Server 2014
- MySQL 5.5.36
- Sybase Adaptive Server Enterprise 15.7

注: EnterpriseDB は、Oracle Real Application Clusters (RAC) および Oracle Exadata での Migration Toolkit の使用をサポートしていません。前述の Oracle 製品は、この EnterpriseDB 製品では評価も認定もされていません。

他のプラットフォームのサポートが必要な場合は、EnterpriseDB アカウントマネージャーまたは sales@enterprisedb.com にお問い合わせください。

第 3 章

移行方法

あるデータベースから別のデータベースへの移行を検討する理由はたくさんあります。移行により、新しいまたはより優れた技術を活用できます。現在のデータベースがシステムのスケーリングを可能にする適切な機能セットを提供していない場合、必要な機能を提供するデータベースに移行することは、会社にとって最適な移行です。

移行も非常に費用対効果が高くなります。多大な保守コストでシステムを移行すると、システムの維持に費やされる費用を節約できます。使用中のデータベースの数を統合することにより、社内の管理コストも削減できます。使用するデータベースプラットフォームの数を減らす（またはデータベースの互換性を活用する）ことにより、IT 予算でより多くのことができます。

複数のデータベースプラットフォームを使用すると、ベンダーが価格を引き上げたり、会社の指示を変更した場合に、適切な移行パスを提供できます。EnterpriseDB は、企業が既存のデータベースシステムを長年 Postgres に移行するのを支援してきました。

移行プロセスで詳述されている方法に従うことをお勧めします。

3.1 移行プロセス

Postgres への移行パスには、次の主な手順が含まれます。

1. 移行に含めるデータベースオブジェクトとデータを決定して、移行プロセスを開始します。ソースシステムのアーキテクチャと実装に関する十分な知識を持つ人を含む移行チームを編成します。
2. 潜在的な移行の問題を特定します。Oracle から Advanced Server への移行の場合、“Oracle 開発者ガイドのデータベース互換性”を参照してください。<<https://www.enterprisedb.com/edb-docs/d/edb-postgres-advanced-server/user-guides/database-compatibility-for-oracle-developers-guide/11/toc.html>> ‘_Advanced Server でサポートされている互換性機能の詳細について。このレビューを支援するために、EnterpriseDB の移行評価サービスの使用を検討してください。
3. 移行環境を準備します。必要なソフトウェアを入手してインストールし、サーバー間の接続を確立します。
4. 移行に大量のデータが含まれる場合は、データを移動する前にスキーマ定義を移行することを検討してくだ

さい。DDL 移行の結果を確認し、移行の概要で報告された問題を解決します。:doc:`移行エラーセクション<mtk\_errors>`このドキュメントの `には、移行の問題の解決に関する情報が含まれています。

5. データを移行します。小さいデータセットの場合は、Migration Toolkit を使用します。（Advanced Server への）Oracle 移行であり、データセットが大きい場合、またはデータ転送が遅いことに気づいた場合は、他の利用可能なデータ移動方法のいずれかを利用してください。

- Oracle データベースと互換性のある Advanced Server データベースリンク機能を使用します。
- データに BLOB または CLOB データがある場合、Oracle スタイルのデータベースリンクの代わりに dblink_ora スタイルのデータベースリンクを使用します。

これらのメソッドはどちらも、Oracle Call Interface (OCI) を使用して Oracle に接続します。接続後、SQL ステートメントを使用して、`リンクされた` Oracle データベースからデータを選択し、そのデータを Advanced Server データベースに挿入します。

6. データ移行の結果を確認し、移行の概要で報告された問題を解決します。
 7. 新しく移行された Postgres データベースで動作するようにアプリケーションを変換します。JDBC や ODBC などのオープンスタンダード接続を使用するアプリケーションでは、通常、データベース接続文字列の変更と EnterpriseDB ドライバーの選択のみが必要です。 [Postgres へのアプリケーションの接続を参照してください](#) 詳細については。
 8. システムのパフォーマンスをテストし、新しいサーバーを調整します。Advanced Server データベースに移行する場合は、Advanced Server のパフォーマンスチューニングユーティリティを利用してください。
- Dynatune を使用して、データベース構成リソースを動的に調整します。
 - オプティマイザーヒントを使用して、クエリパスを指定します。
 - ANALYZE コマンドを使用して、データベース統計を取得します。

The *EDB Postgres Advanced Server Guide* and *Database Compatibility for Oracle Developer's Guide* (both available through the EnterpriseDB website) offer information about the performance tuning tools available with Advanced Server.

3.2 アプリケーションを Postgres に接続する

クライアントアプリケーションを変換して Postgres データベースを使用するには、接続プロパティを変更して新しいターゲットデータベースを指定する必要があります。Java アプリケーションの場合、JDBC ドライバー名 (Class.forName) と JDBC URL を変更します。

Oracle で実行されている Java アプリケーションには、次の接続プロパティがあります。

```
Class.forName("oracle.jdbc.driver.OracleDriver");
Connection con =
DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe",
"user",
"password")
```

接続文字列を変更して、Postgres サーバーに接続します。

```
Class.forName("com.edb.Driver")
Connection con = DriverManager.getConnection("jdbc:edb://localhost:5444/edb", "user",
↪ "password");
```

ODBC アプリケーションを変換して Postgres のインスタンスに接続するには、2 段階のプロセスが必要です。ODBC アプリケーションを接続するには、ODBC データソース管理者を使用して、新しいターゲットデータベースの接続プロパティを定義するデータソースを作成します。

ほとんどの Linux および Windows システムには、ODBC データソースを作成および編集できるグラフィカルツールが含まれています。ODBC をインストールしたら、[管理ツール] メニューで ODBC データソースアドミニストレーターへのリンクを確認します。[追加] ボタンをクリックして、新しいデータソースの作成ウィザードを開始します。ダイアログを完了して、新しいターゲットデータソースを定義します。

次に、新しいデータソースを使用するようにアプリケーションを変更します。

アプリケーションには SQLConnect (または場合によっては SQLDriverConnect) への呼び出しが含まれます。呼び出しを編集して、データソース名を変更します。次の例では、データソースの名前は OracleDSN です:

```
result = SQLConnect(conHandle, // Connection handle (returned)
"OracleDSN", SQL_NTS, // Data source name
username, SQL_NTS, // User name
password, SQL_NTS); // Password
```

PostgresDSN という名前のデータソースで定義された Postgres のインスタンスに接続するには、データソース名を変更します。

```
result = SQLConnect(conHandle, // Connection handle (returned)
"PostgresDSN", SQL_NTS, // Data source name
username, SQL_NTS, // User name
password, SQL_NTS); // Password
```

アプリケーションとサーバー間の接続を確立した後、アプリケーションをテストして、アプリケーションと移行されたスキーマ間の互換性の問題を見つけます。ほとんどの場合、簡単な変更により、アプリケーションで発生する非互換性が解決されます。機能がサポートされていない場合は、回避策またはサードパーティのツールを使用して、アプリケーションに必要な機能を提供してください。 [移行エラーを参照してください](#)、いくつかの一般的な問題とその回避策についての情報。

第 4 章

機能概要

Migration Toolkit は、移行プロセスをきめ細かく制御できる強力なコマンドラインツールです。Migration Toolkit の使用は、2 段階のプロセスです。

1. `toolkit.properties` ファイルを編集して、ソースデータベースとターゲットデータベースを指定します。
2. コマンドラインで Migration Toolkit を呼び出し、移行オプションを指定します。

Migration Toolkit は、次のものから Advanced Server または PostgreSQL データベースへのデータベースオブジェクトとデータの移行を容易にします。

- オラクル
- MySQL
- SQL サーバー

Migration Toolkit を使用すると、データベースオブジェクトとデータを Sybase から Advanced Server データベースに移行することもできます。Migration Toolkit を使用して、Advanced Server と PostgreSQL の間を移行することもできます。Migration Toolkit には多数のオプションが含まれており、移行プロセスをきめ細かく制御できます。

- 移行される各行をコミットするには、`-safeMode` オプションを使用します。
- 移行を最適化するために WAL ログをバイパスするには、`-fastCopy` オプションを使用します。
- 一括挿入のバッチサイズを制御するには、`-batchSize` オプションを使用します。
- COPY コマンドで使用するバッチサイズを指定するには、`-cpBatchSize` オプションを使用します。
- `-lobBatchSize` オプションを使用して、ラージオブジェクトデータ型に使用されるバッチサイズを指定します。
- ユーザー定義の条件を満たす行のみを移行するには、`-filterProp` オプションを使用します。

- 選択した列のデータ型を変更するには、`-customColTypeMapping` オプションを使用します。
- `-dropSchema` オプションを使用して、移行前に既存のスキーマを削除し、新しいスキーマを作成します。
- Advanced Server では、`-allDBLinks` オプションを使用してすべての Oracle データベースリンクを移行します。
- Advanced Server では、`-copyViaDBLinkOra` オプションを使用して `dblink_ora` モジュールを有効にします。

オブジェクト移行のサポート

Migration Toolkit は、オブジェクト定義 (DDL)、テーブルデータ、またはその両方を移行します。次の表には、Migration Toolkit が移行できるデータベースオブジェクトのタイプのプラットフォーム固有のリストが含まれています。

対象	オラクル	Sybase	SQL サーバー	MySQL
スキーマ	X	X	X	X
テーブル	X	X	X	X
リストパーティションテーブル	X			
範囲パーティションテーブル	X			
制約	X	X	X	X
インデックス	X	X	X	X
トリガー	X			
テーブルデータ	X	X	X	X
Views	X		X	
マテリアライズドビュー	X			
パッケージ	X			
手続き	X			
機能	X			
シーケンス	X			
ユーザー/ロール	X			
プロフィール	X			
オブジェクトタイプ	X			
オブジェクト型メソッド	X			
データベースリンク	X			
キュー	X			

インポートされたオブジェクトの詳細な制御を提供するコマンドの詳細については、[スキーマオブジェクト選択オプション](#)を参照してください。

オンライン移行とオフライン移行

Migration Toolkit は、すぐに直接 Postgres データベースに移行することができます (* online ** migration)、または *Postgres* データベースでオブジェクト定義を再作成するために後で使用するスクリプトを生成することもできます (offline ** migration *) 。

デフォルトでは、Migration Toolkit は Postgres データベースにオブジェクトを直接作成します。対照的に、-offlineMigration オプションを含めて、後で使用して新しいデータベースに移行したオブジェクトまたはデータを再現できる SQL スクリプトを生成します。実行する前に Migration Toolkit によって生成された移行スクリプトをカスタマイズすることにより、移行されたオブジェクトを変更できます。-offlineMigration オプションを使用すると、システムの負荷に最適な時間に実際の移行をスケジュールできます。

-offlineMigration オプションの詳細については、[オフライン移行オプションを参照してください](#)。

第 5 章

移行ツールキットのインストール

RPM パッケージまたは Stack Builder を使用して、Migration Toolkit をインストールできます。Stack Builder は Advanced Server と PostgreSQL のワンクリックインストーラーの両方で配布され、EnterpriseDB から入手できます。

Migration Toolkit をインストールする前に、最初に Java（バージョン 1.7.0 以降）をインストールする必要があります。Java インストーラーおよびインストール手順の無料ダウンロードは、次のサイトで入手できます。

<http://www.java.com/en/download/index.jsp>

5.1 RPM パッケージを使用して Migration Toolkit をインストールする

RPM パッケージを使用して、Migration Toolkit を 64 ビット Linux ホストにインストールできます。Migration Toolkit をインストールする前に、次のことを行う必要があります。

- Migration Toolkit ホストに Java 環境（バージョン 1.7.0 以降）をインストールします。
- EnterpriseDB リポジトリへのアクセスを許可する資格情報も必要です。資格情報のリクエストについては、次をご覧ください。

<https://info.enterprisedb.com/rs/069-ALB-339/images/Repository%20Access%2004-09-2019.pdf>

リポジトリの資格情報を受け取った後、次のことができます。

1. リポジトリ構成ファイルを作成します。
2. ユーザー名とパスワードを指定して、ファイルを変更します。
3. Migration Toolkit をインストールします。

リポジトリ設定ファイルの作成

リポジトリ構成ファイルを作成するには、スーパーユーザー特権を引き受けて、次のコマンドを呼び出します。

```
yum -y install https://yum.enterprisedb.com/edb-repo-rpms/edb-repo-latest.noarch.rpm
```

リポジトリ設定ファイルの名前は `edb.repo` です。ファイルは `/etc/yum.repos.d` にあります。

`edb.repo` ファイルを作成した後、エディターの選択を使用して、1 パラメーターの値が 1 であることを確認し、`username` および `password` プレースホルダーを置き換えます登録された EnterpriseDB ユーザーの名前とパスワードを含む `baseurl` 仕様。

```
[edb]
name=EnterpriseDB RPMs $releasever - $basearch
baseurl=https://<username>:<password>@yum.enterprisedb.com/edb/redhat/rhel-$releasever-
→$basearch
enabled=1
gpgcheck=1
gpgkey=file:///etc/pki/rpm-gpg/ENTERPRISEDB-GPG-KEY
```

設定ファイルへの変更を保存した後、`yum install` コマンドを使用して Migration Toolkit をインストールできます。

```
yum install edb-migrationtoolkit
```

システムで認識されないソースによって署名された RPM パッケージをインストールすると、`yum` はローカルサーバーにキーをインポートする許可を求める場合があります。プロンプトが表示され、パッケージが信頼できるソースからのものであることに満足したら、`y` を入力し、Return を押して続行します。

インストール中に、`yum` は解決できない依存関係に遭遇する場合があります。存在する場合、手動で解決する必要がある依存関係のリストが提供されます。

Migration Toolkit をインストールした後、インストールを構成する必要があります。Migration Toolkit を呼び出す前に、次の手順を実行します。

IDENT 認証で Migration Toolkit を使用

デフォルトでは、RPM インストーラーの `pg_hba.conf` ファイルはリモートクライアントに IDENT 認証を強制します。Migration Toolkit を呼び出す前に、`pg_hba.conf` ファイルを変更して認証方法を IDENT 以外の形式に変更する（およびサーバーを再起動する）か、次の手順を実行して IDENT “サーバーにアクセス可能:

1. `identd` サーバーがインストールされ、実行されていることを確認します。次のコマンドを呼び出して、`yum` パッケージマネージャーを使用して `identd` サーバーをインストールできます。

```
yum install xinetd authd
```

このコマンドは、`/etc/xinetd.d/auth` という名前のファイルを作成する必要があります。

```
service auth
{
disable = yes
```

(次のページに続く)

(前のページからの続き)

```
socket_type = stream
wait = no
user = ident
cps = 4096 10
instances = UNLIMITED
server = /usr/sbin/in.authd
server_args = -t60 --xerror --os
}
```

注: ファイルの `server_args` の末尾に `-E` 引数が含まれている場合は、`-E` を消去してください。

次に、`identd` サーバーを起動するには、コマンドを呼び出します:

```
systemctl enable xinetd
systemctl start xinetd
```

2. `pg_ident.conf` ファイルを開き、ユーザーマッピングを作成します。

```
# map_name system_username postgres_username
edbas enterprisedb enterprisedb
```

どこで:

`map_name` 列で指定された名前は、`pg_hba.conf` ファイル内のマッピングを識別するユーザー定義の名前です。

`system_username` 列に指定された名前は `enterprisedb` です。

`postgres_username` 列に指定された名前は `enterprisedb` です。

3. `pg_hba.conf` ファイルを開き、`IDENT` エントリを変更します。

IPv4 ホストの場合:

```
host all all 127.0.0.0/0 ident map=edbas
```

IPv6 ホストの場合:

```
host all all::1/128 ident map=edbas
```

4. Migration Toolkit を呼び出す前に、Advanced Server サービスを再起動します。

5.2 Stack Builder を使用して移行ツールキットをインストールする

注:このガイドでは、Stack Builder という用語を使用して、* StackBuilder Plus (Advanced Server で配布) または Stack Builder (EnterpriseDB の PostgreSQL ワンクリックインストーラーで配布) を指します。

Stack Builder で Migration Toolkit のインストールを実行する前に、Java JVM (バージョン 1.7.0 以降) を用意する必要があります。Java インストーラーおよびインストール手順の無料ダウンロードは、次のサイトで入手できます。

<http://www.java.com/en/download/index.jsp>

Java 実行可能ファイルは検索パスにある必要があります (Windows では %PATH%、Linux / Unix では \$PATH)。次のコマンドを使用して、検索パスを設定します (* javadir * の Java 実行可能ファイルを保持するディレクトリの名前を置き換えます)。

Windows では、次のコマンドを使用します。

```
SET PATH=/ *javadir*; %PATH%
```

Linux では、次のコマンドを使用します。

```
PATH=/ *javadir*:$PATH
```

検索パスを設定したら、Stack Builder インストールウィザードを使用して、Migration Toolkit を Advanced Server または PostgreSQL にインストールできます。

Advanced Server から StackBuilder Plus を起動

既存の Advanced Server インストールから StackBuilder Plus を起動するには、Start (または Applications) メニューから EDB Postgres メニューに移動します。EDB Add-ons メニューを開き、StackBuilder Plus メニューオプションを選択します。

PostgreSQL から Stack Builder を起動

PostgreSQL インストールから Stack Builder を起動するには、Start (または Applications) メニューから PostgreSQL メニューに移動し、Application StackBuilder Plus メニューオプションを選択します。

以下に示すように、Stack Builder が開きます。

ドロップダウンリストボックスを使用して、使用可能なサーバーのリストからターゲットサーバーのインストールを選択します。ネットワークでプロキシサーバーを使用してインターネットにアクセスする必要がある場合は、Proxy servers ボタンを使用して Proxy servers ダイアログを開き、サーバーを指定します。プロキシサーバーを使用する必要がない場合は、Next をクリックしてアプリケーション選択ウィンドウを開きます。

StackBuilder Plus を使用して Migration Toolkit を Advanced Server インストールに追加する場合は、ツリーコントロールの Add-ons, tools and utilities ノードを展開し、EnterpriseDB Migration Toolkit の横のボックスをオンにします。Next をクリックして続行します。

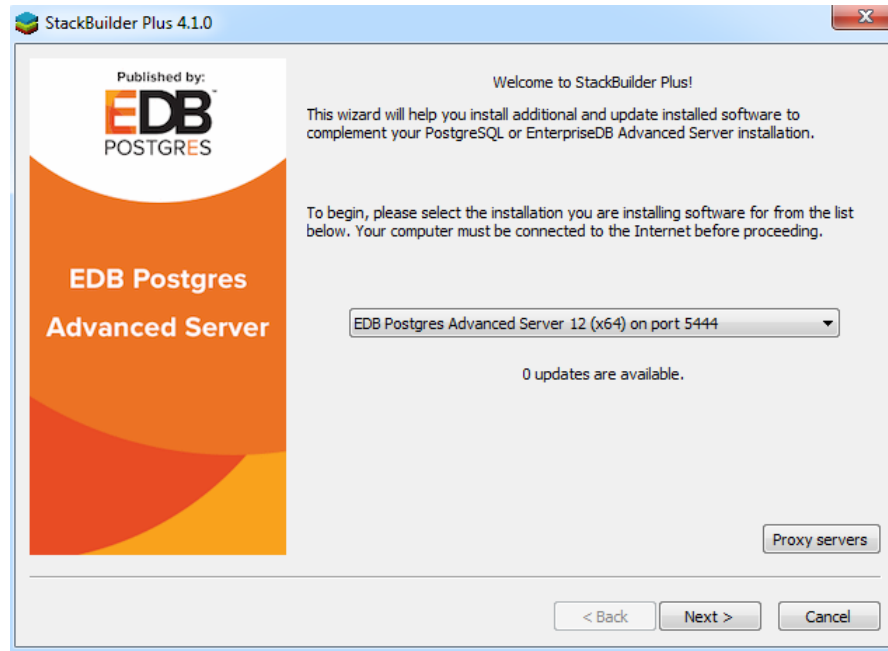


図 1 The Stack Builder welcome window.

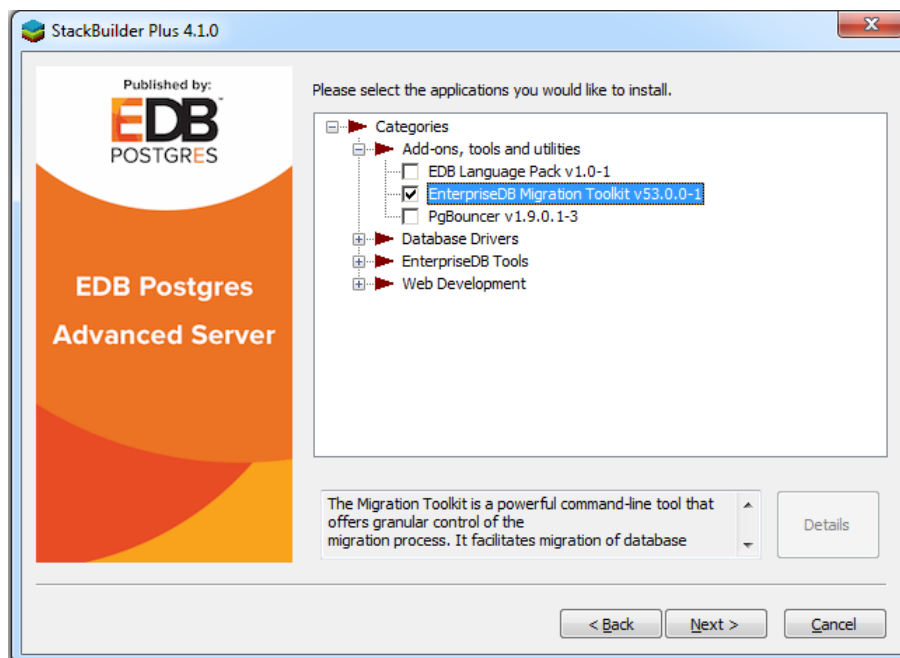


図 2 The Stack Builder application selection window.

Stack Builder を使用して Migration Toolkit を PostgreSQL インストールに追加している場合は、ツリーコントロールの EnterpriseDB Tools ノード（Registration-required and trial products ノードの下にあります）を展開し、Migration Toolkit の横のボックスをオンにします。Next をクリックして続行します。

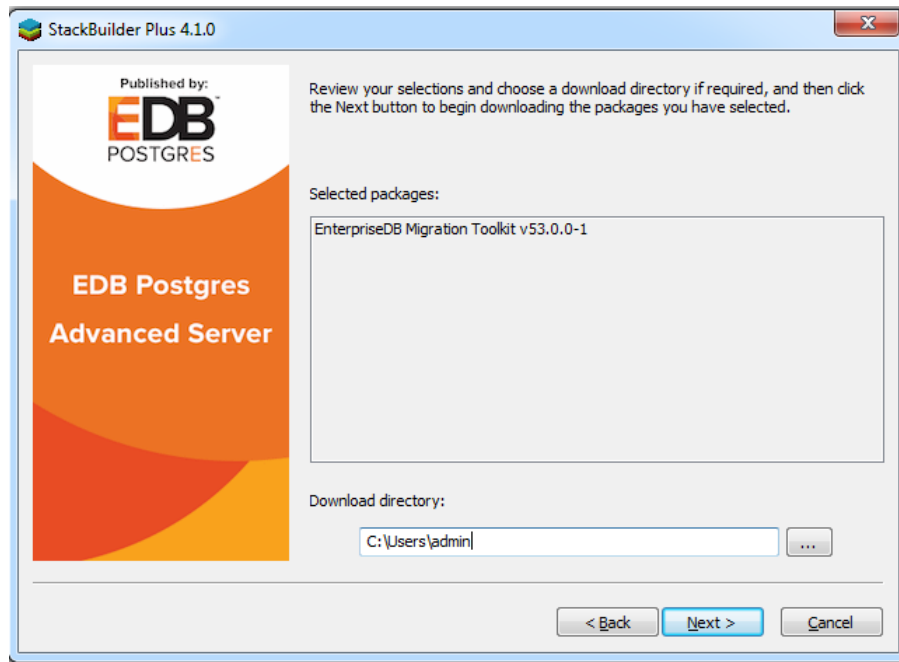


図 3 The Stack Builder selection confirmation window.

Migration Toolkit が Selected Packages リストに含まれており、Download directory フィールドに許容可能なダウンロード場所が含まれていることを確認します。

Next をクリックして、Migration Toolkit のダウンロードを開始します。

ダウンロードが完了すると、Stack Builder はインストールファイルが正常にダウンロードされたことを確認します。Next を選択して、Migration Toolkit インストールウィザードを開きます。

Migration Toolkit インストールウィザードのプロンプトが表示されたら、インストールの言語を指定し、OK をクリックして続行します。

Migration Toolkit インストールウィザードが開きます。Next をクリックして続行します。

適切なラジオボタンを強調表示する前に、ライセンス契約を注意深く確認してください。Next をクリックして続行します。

デフォルトでは、Migration Toolkit は mtk ディレクトリにインストールされます。表示されているデフォルトのインストールディレクトリを受け入れるか、ディレクトリを変更し、Next をクリックして続行します。

インストールウィザードは、Setup プログラムが Migration Toolkit をインストールする準備ができていることを確認します。Next をクリックしてインストールを開始します。

ダイアログで、Migration Toolkit のインストールが完了したことが確認されます。Finish をクリックして、

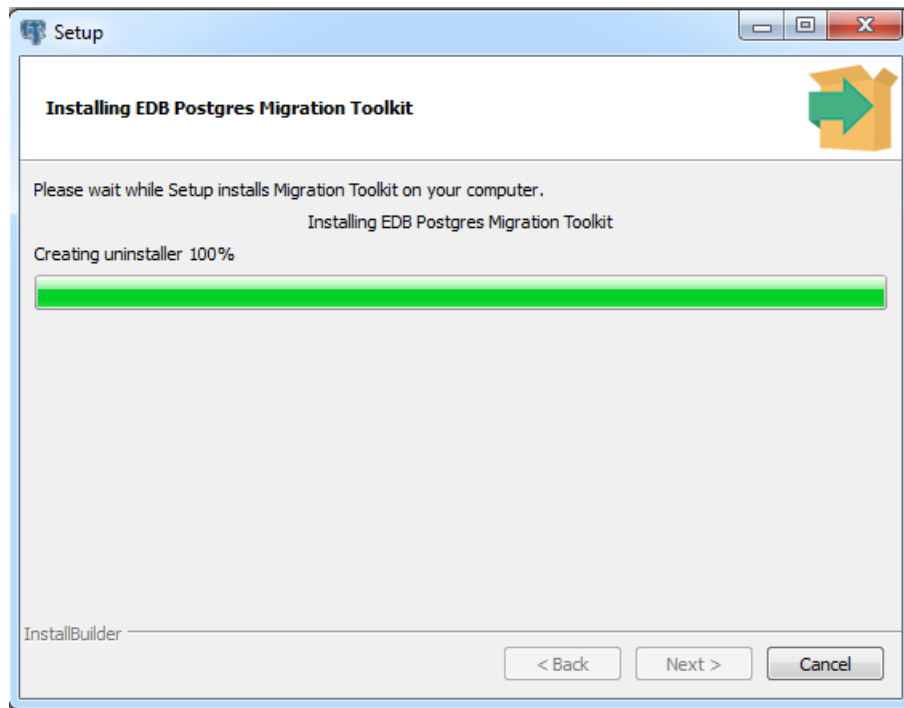


図 4 Downloading the Migration Toolkit installer.

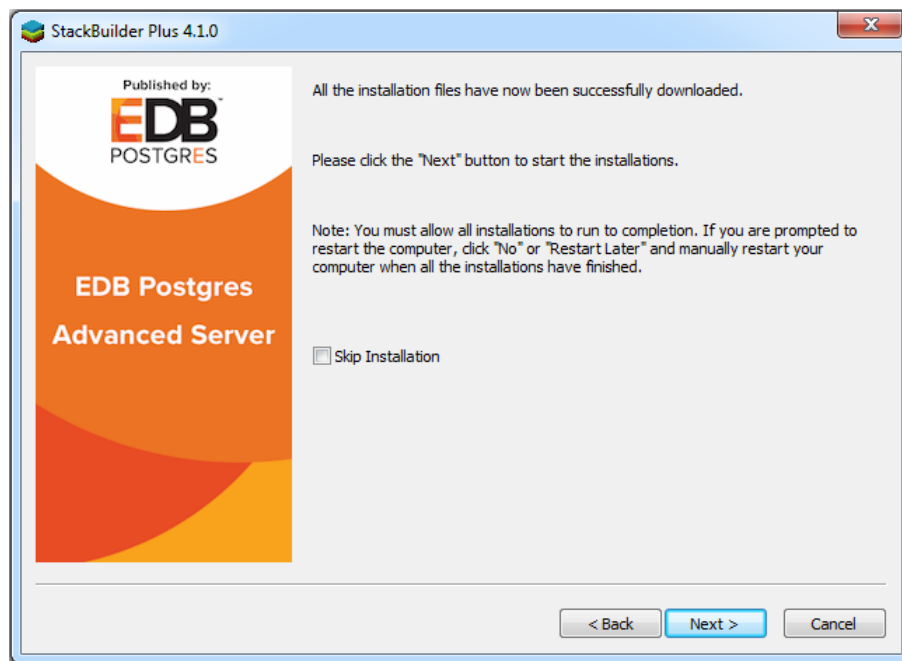


図 5 Stack Builder confirms the file download.

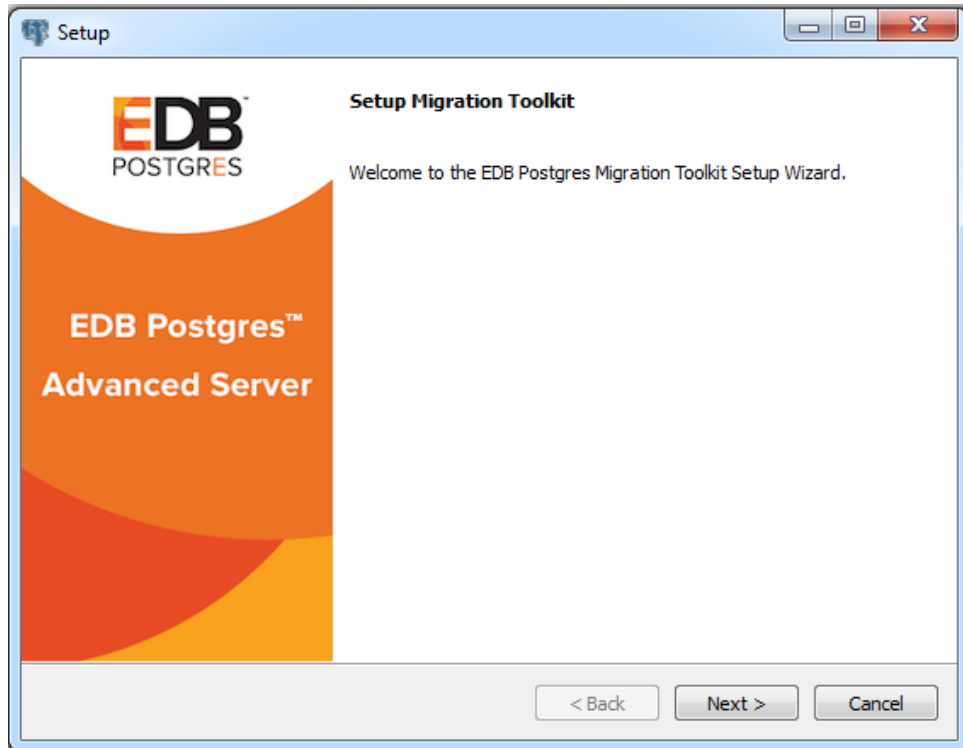


図 6 The Migration Toolkit installation wizard.



図 7 The EnterpriseDB license agreement.

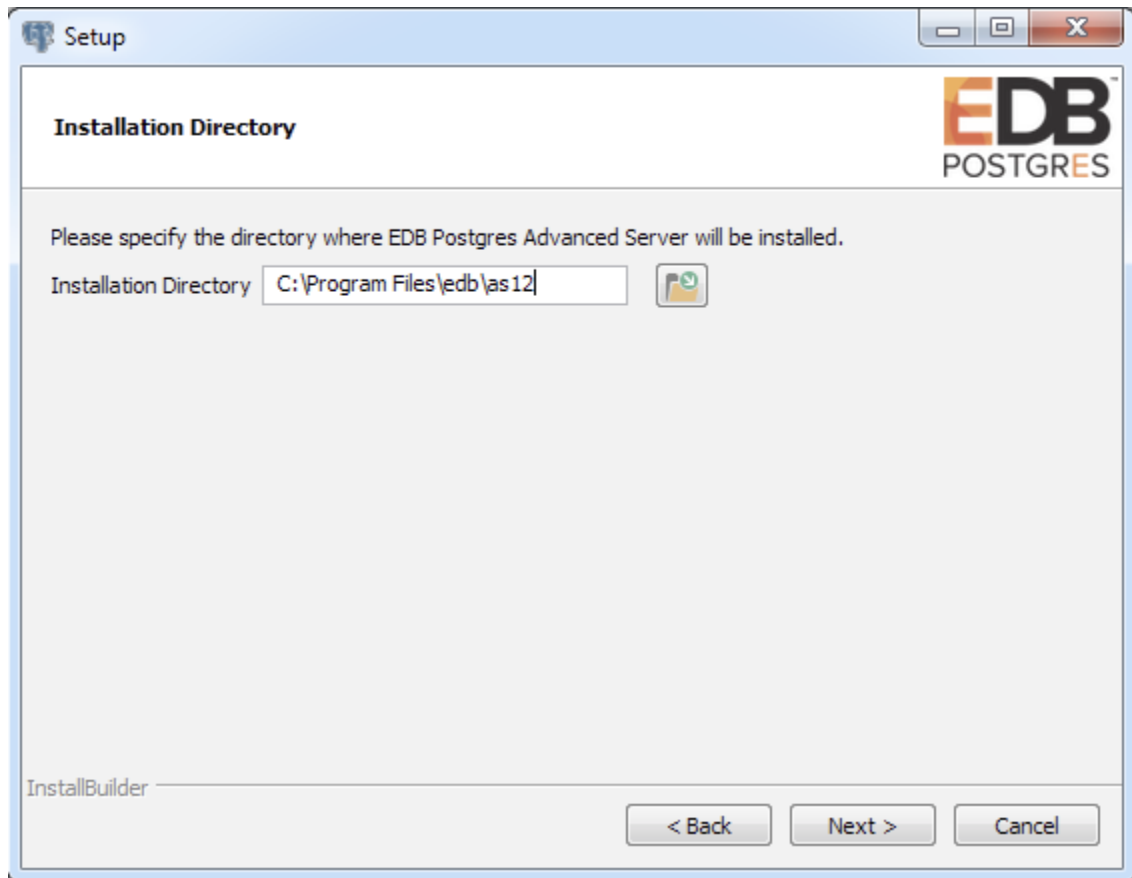


図 8 *Specify a Migration Toolkit installation directory.*

Migration Toolkit インストーラーを終了します。

Stack Builder が最後に選択されたコンポーネントのインストールを完了すると、Installation Completed ウィンドウが表示されます。Finish をクリックして Stack Builder を閉じます。

Stack Builder で Migration Toolkit をインストールした後、移行を実行する前に適切なソース固有のドライバーをインストールする必要があります。 [ソース固有のドライバのインストールを参照してください](#) 詳細については。

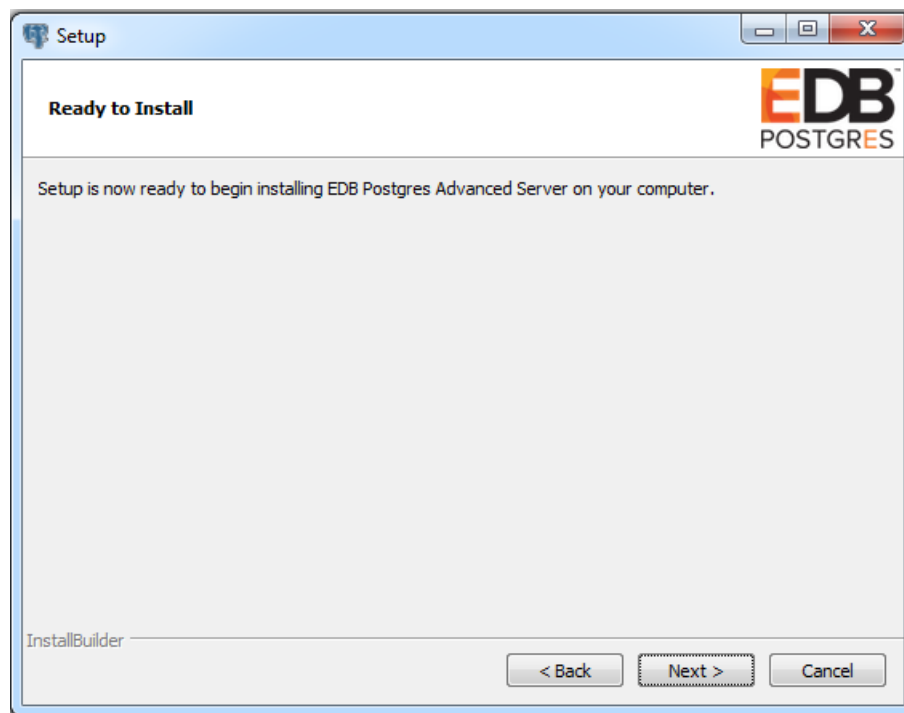


図 9 セットアップで *Migration Toolkit* のインストールを開始する準備ができました。

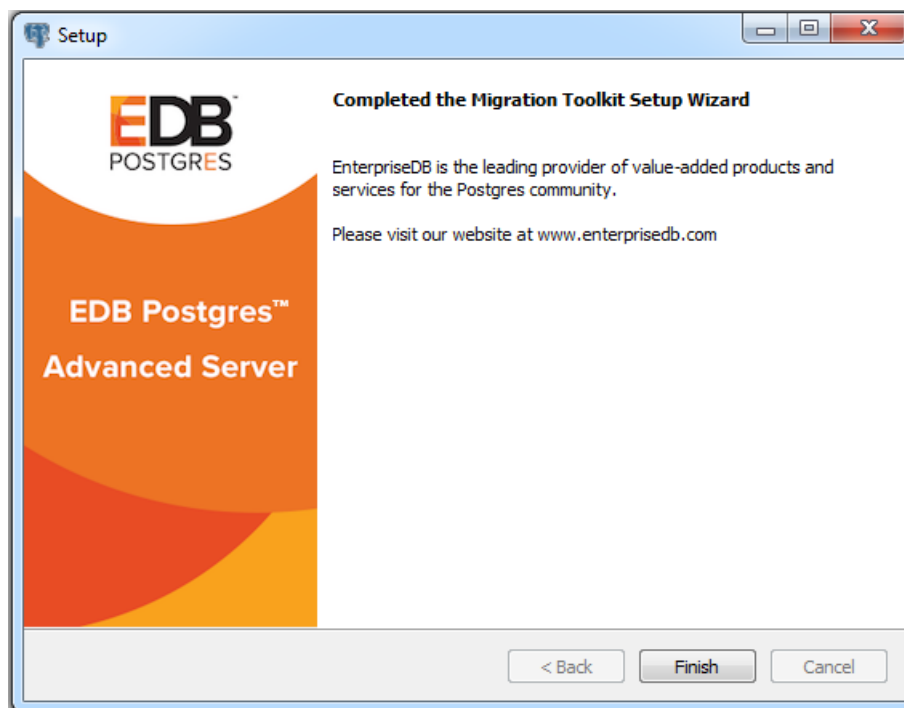


図 10 移行ツールキットウィザードが終了しました。

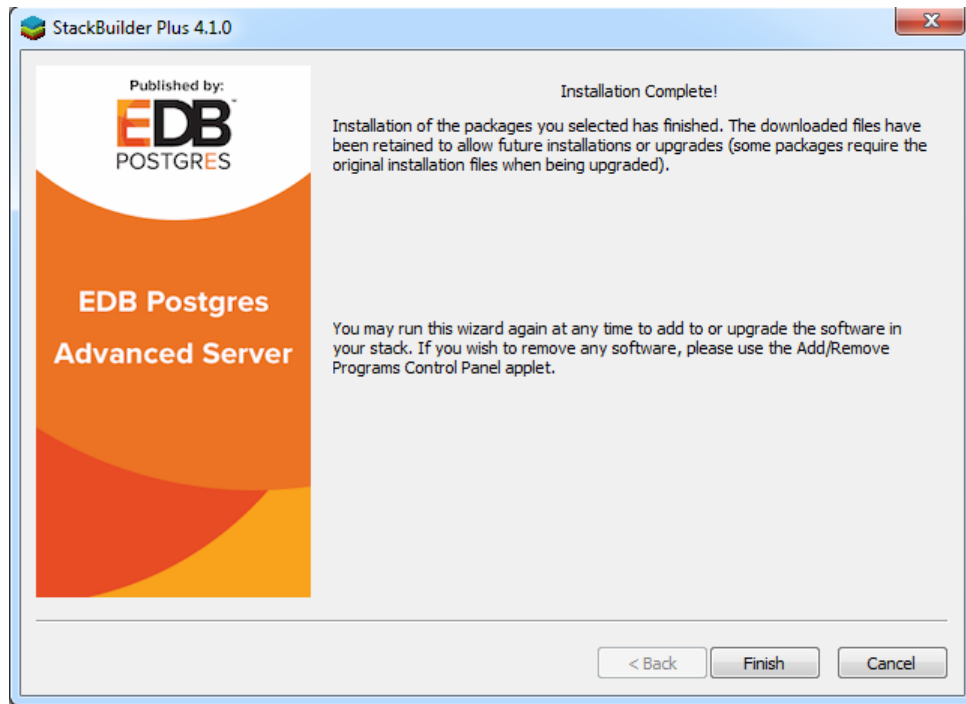


図 11 *Stack Builder confirms the installation is complete.*

5.3 ソース固有のドライバーのインストール

Migration Toolkit を呼び出す前に、無料で入手可能なソース固有のドライバーをダウンロードしてインストールする必要があります。ドライバーをダウンロードするには、またはベンダーのダウンロードサイトへのリンクについては、EnterpriseDB Web サイトの Advanced Downloads ページの Third Party JDBC Drivers セクションにアクセスしてください。

<https://www.enterprisedb.com/advanced-downloads>

ソース固有のドライバーをダウンロードしたら、ドライバーファイルを `JAVA_HOME/jre/lib/ext` ディレクトリに移動します。

第 6 章

toolkit.properties ファイルの作成

Migration Toolkit は、移行プロセス中に toolkit.properties ファイルに保存されている設定および接続情報を使用して、ソースデータベースとターゲットデータベースを識別して接続します。Linux では、toolkit.properties ファイルは次の場所にあります。

```
/usr/edb/migrationtoolkit/etc
```

Windows では、ファイルは次の場所にあります。

```
C:/Program Files/edb/mtk/etc
```

サンプル toolkit.properties ファイルを以下に示します:

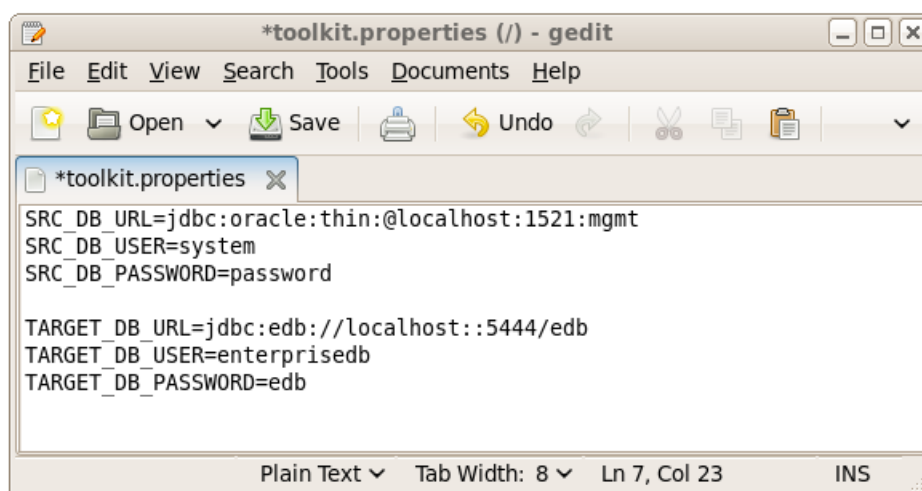


図 1 典型的な toolkit.properties ファイル

Migration Toolkit コマンドを実行する前に、選択したエディターで toolkit.properties ファイルを変更します。ファイルを更新して、次の情報を含めます。

- SRC_DB_URL は、Migration Toolkit がソースデータベースに接続する方法を指定します。URL の形成の詳細については、ソースデータベースに対応するセクションを参照してください。

- SRC_DB_USER は、ソースデータベース内の（十分な特権を持つ）ユーザー名を指定します。
- SRC_DB_PASSWORD は、ソースデータベースユーザーのパスワードを指定します。
- TARGET_DB_URL は、ターゲットデータベースの JDBC URL を指定します。
- TARGET_DB_USER は、特権のあるターゲットデータベースユーザーの名前を指定します。
- TARGET_DB_PASSWORD は、ターゲットデータベースユーザーのパスワードを指定します。

6.1 Advanced ServerURL の定義

Migration Toolkit は、次のプラットフォームから Advanced Server への移行を容易にします。

- オラクル
- MySQL
- Sybase
- SQL サーバー
- PostgreSQL

各データベースタイプから移行されたオブジェクトの決定的なリストについては、[機能の概要を参照してください](#)。

Migration Toolkit は、ソースおよびターゲットデータベースの接続仕様を `toolkit.properties` ファイルから読み取ります。それぞれの接続情報には以下が含まれている必要があります。

- データベースの URL
- 特権ユーザーの名前
- 指定されたユーザーに関連付けられたパスワード。

URL は JDBC 標準に準拠し、次の形式を取ります。

```
{TARGET_DB_URL|SRC_DB_URL}=jdbc:edb://<host>:<port>/<database_id>
```

Advanced Server URL には次の情報が含まれています。

`jdbc`

プロトコルは常に `jdbc` です。

`edb`

Advanced Server を使用している場合、サブプロトコル値に `edb` を指定します。

`<host>`

Postgres インスタンスが実行されているホストの名前または IP アドレス。

`<port>`

Advanced Server データベースリスナーが監視しているポート番号。デフォルトのポート番号は 5444 です。

`<database_id>`

ソースまたはターゲットデータベースの名前。

{TARGET_DB_USER|SRC_DB_USER} は、移行される各タイプのオブジェクトを作成する権限を持つユーザーを指定する必要があります。データをテーブルに移行する場合、指定されたユーザーは各ターゲットテーブルに対して INSERT、TRUNCATE、および REFERENCES 権限も必要とする場合があります。

特権 Advanced Server ユーザーのパスワードには {TARGET_DB_PASSWORD|SRC_DB_PASSWORD} が設定されています。

6.2 PostgreSQL URL の定義

Migration Toolkit は、次のプラットフォームから PostgreSQL への移行を容易にします。

- オラクル
- MySQL
- SQL サーバー
- Advanced Server

各データベースタイプから移行されたオブジェクトの決定的なリストについては、[機能の概要を参照してください](#)。

Migration Toolkit は、ソースおよびターゲットデータベースの接続仕様を `toolkit.properties` ファイルから読み取ります。それぞれの接続情報には以下が含まれている必要があります。

- データベースの URL
- 特権ユーザーの名前
- 指定されたユーザーに関連付けられたパスワード。

PostgreSQL URL は JDBC 標準に準拠し、次の形式を取ります。

```
{SRC_DB_URL|TARGET_DB_URL}=jdbc:postgresql://<host>:<port>/<database_id>
```

URL には次の情報が含まれます。

`jdbc`

プロトコルは常に `jdbc` です。

`postgresql`

PostgreSQL を使用している場合、サブプロトコル値に `postgresql` を指定します。

`<host>`

Postgres インスタンスが実行されているホストの名前または IP アドレス。

`<port>`

Postgres データベースリスナーが監視しているポート番号。デフォルトのポート番号は 5432 です。

`<database_id>`

ソースまたはターゲットデータベースの名前。

{SRC_DB_USER|TARGET_DB_USER} は、移行される各タイプのオブジェクトを CREATE する特権を持つユーザーを指定する必要があります。データをテーブルに移行する場合、指定されたユーザーは、各ターゲットテーブルに INSERT、TRUNCATE、および REFERENCES の権限も必要とする場合があります。

PostgreSQL の特権ユーザーのパスワードには {SRC_DB_PASSWORD|TARGET_DB_PASSWORD} が設定されています。

6.3 Oracle URL の定義

Migration Toolkit は、Oracle データベースから PostgreSQL または Advanced Server データベースへの移行を容易にします。Oracle から移行する場合、`toolkit.properties` ファイルで Oracle ソースデータベースの接続仕様を指定する必要があります。接続情報には以下を含める必要があります。

- Oracle データベースの URL
- 特権ユーザーの名前
- 指定されたユーザーに関連付けられたパスワード。

Oracle データベースから移行する場合、`SRC_DB_URL` には 2 つの形式のいずれかで指定された JDBC URL が含まれている必要があります。最初の形式は次のとおりです。

```
jdbc:oracle:thin:@<host_name>:<port>:<database_id>
```

2 番目の形式は次のとおりです。

```
jdbc:oracle:thin:@//<host_name>:<port>{<database_id|service_name>}
```

Oracle URL には次の情報が含まれます。

`jdbc`

プロトコルは常に `jdbc` です。

`oracle`

サブプロトコルは常に `oracle` です。

`thin`

ドライバのタイプ。 `thin` のドライバタイプを指定します。

`<host_name>`

Oracle サーバーが実行されているホストの名前または IP アドレス。

`<port>`

Oracle データベースリスナーが監視しているポート番号。

`<database_id>`

Oracle データベースのデータベース SID。

`<service_name>`

Oracle サービスの名前。

SRC_DB_USER は特権 Oracle ユーザーの名前を指定する必要があります。Oracle ユーザーには、オブジェクトを Oracle から Advanced Server に移行するための DBA 権限が必要です。Oracle `GRANT DBA TO user` コマンドを使用して DBA 権限を Oracle ユーザーに付与し、目的のデータベースオブジェクトをすべて移行することができます。

SRC_DB_PASSWORD には、指定されたユーザーのパスワードが含まれている必要があります。

6.4 MySQL URL の定義

Migration Toolkit は、MySQL データベースから Advanced Server または PostgreSQL データベースへの移行を容易にします。MySQL から移行する場合、`toolkit.properties` ファイルで MySQL ソースデータベースの接続仕様を指定する必要があります。接続情報には以下を含める必要があります。

- ソースデータベースの URL
- 特権ユーザーの名前
- 指定されたユーザーに関連付けられたパスワード。

MySQL から移行する場合、`SRC_DB_URL` は JDBC URL の形式を取ります。例:

```
jdbc:mysql://<host_name>[:<port>]/<database_id>
```

URL には次の情報が含まれます。

`jdbc`

プロトコルは常に `jdbc` です。

`mysql`

サブプロトコルは常に `mysql` です。

`<host_name>`

移行元サーバーが実行されているホストの名前または IP アドレス。

`[:<port>]`

MySQL データベースリスナーが監視しているポート番号。

`<database_id>`

ソースデータベースの名前。

`SRC_DB_USER` は、特権のある MySQL ユーザーの名前を指定する必要があります。

`SRC_DB_PASSWORD` には、指定されたユーザーのパスワードが含まれている必要があります。

6.5 Sybase URL の定義

Migration Toolkit は、Sybase データベースから Advanced Server データベースへの移行を容易にします。Sybase から移行する場合、`toolkit.properties` ファイルで Sybase ソースデータベースの接続仕様を指定する必要があります。接続情報には以下を含める必要があります。

- ソースデータベースの URL
- 特権ユーザーの名前
- 指定されたユーザーに関連付けられたパスワード。

Sybase から移行する場合、`SRC_DB_URL` は JTDS URL の形式を取ります。例:

```
jdbc:jtds:sybase://<host_name>[:<port>]/<database_id>
```

Sybase URL には次の情報が含まれています。

`jdbc`

プロトコルは常に `jdbc` です。

`jtds`

ドライバー名は常に `jtds` です。

`sybase`

サーバータイプは常に `sybase` です。

`<host_name>`

移行元サーバーが実行されているホストの名前または IP アドレス。

`<port>`

Sybase データベースリスナーが監視しているポート番号。

`<database_id>`

ソースデータベースの名前。

`SRC_DB_USER` は、特権 Sybase ユーザーの名前を指定する必要があります。

`SRC_DB_PASSWORD` には、指定されたユーザーのパスワードが含まれている必要があります。

6.6 SQL Server URL の定義

Migration Toolkit は、SQL Server データベースから PostgreSQL または Advanced Server データベースへの移行を容易にします。Migration Toolkit は、次のオブジェクト定義の移行をサポートしています。

- スキーマ
- テーブル
- テーブルデータ
- 制約
- インデックス

Migration Toolkit は、`toolkit.properties` ファイルからソースデータベースの接続仕様を読み取ります。接続情報には以下を含める必要があります。

- ソースデータベースの URL
- 特権ユーザーの名前
- 指定されたユーザーに関連付けられたパスワード。

SQL Server データベースに接続している場合、`SRC_DB_URL` は JTDS URL の形式を取ります。例:

```
jdbc:jtds:sqlserver:/<server_name>/[:<port>]/<database_id>
```

SQL Server URL には次の情報が含まれます。

`jdbc`

プロトコルは常に `jdbc` です。

`jtds`

ドライバー名は常に `jtds` です。

`sqlserver`

サーバータイプは常に `sqlserver` です。

`<server_name>`

移行元サーバーが実行されているホストの名前または IP アドレス。

`<port>`

ソースデータベースリスナーが監視しているポート番号。

`<database_id>`

ソースデータベースの名前。

SRC_DB_USER は、特権 SQL Server ユーザーの名前を指定する必要があります。

SRC_DB_PASSWORD には、指定されたユーザーのパスワードが含まれている必要があります。

第 7 章

移行ツールキットの呼び出し

Migration Toolkit をインストールし、`toolkit.properties` ファイルでソースデータベースとターゲットデータベースの接続プロパティを指定した後、移行ツールキットは移行を実行する準備ができています。

Migration Toolkit 実行可能ファイルの名前は、Linux システムでは `runMTK.sh`、Windows システムでは `runMTK.bat` です。Linux システムでは、実行可能ファイルは次の場所にあります。

```
/usr/edb/migrationtoolkit/bin
```

Windows では、実行可能ファイルは次の場所にあります。

```
C:/Program Files/edb/mtk/bin
```

注: Migration Toolkit の呼び出し時に次のエラーが表示される場合は、`toolkit.properties` ファイルのファイル権限を確認してください。

```
MTK-11015: The connection credentials file ../etc/toolkit.properties is
not secure and accessible to group/others users. This file contains
plain passwords and should be restricted to Migration Toolkit owner user
only.
```

Migration Toolkit を実行するオペレーティングシステムのユーザーアカウントは、ファイルの読み取り権限が最低限必要な `toolkit.properties` ファイルの所有者でなければなりません。さらに、グループおよび他のユーザーにはいかなる種類のアクセス許可も許可されてはなりません。以下は、ユーザー `enterprisedb` が Migration Toolkit を実行している場合の推奨ファイル許可の例です。

```
-rw----- 1 enterprisedb enterprisedb 191 Aug 1 09:59 toolkit.properties
```

バイナリゼロが埋め込まれた文字データのインポート (NULL 文字)

Migration Toolkit は、値が NULL の列のインポートを適切にサポートします。

ただし、Migration Toolkit は、JDBC 接続プロトコルを使用した NULL 文字値 (埋め込みバイナリゼロ `0x00`) のインポートをサポートしていません。NULL 文字を含むデータをインポートする場合、`-replaceNullChar` オプションを使用して、NULL 文字を単一の非 NULL の置換文字に置き換えます。

注:

- MTK は暗黙的に NULL 文字を空の文字列に置き換えます。
- `-replaceNullChar` オプションは `-copyViaDBLinkOra` オプションでは機能しません。

データが移行されたら、SQL ステートメントを使用して、`-replaceNullChar` で指定された文字をバイナリのゼロに置き換えます。

7.1 Oracle からのスキーマの移行

コマンドラインで指定されていない限り、Migration Toolkit はソースデータベースが Oracle であり、ターゲットデータベースが Advanced Server であると想定しています。Linux で完全なスキーマを移行するには、実行可能ファイルに移動し、次のコマンドを呼び出します。

```
$ ./runMTK.sh <schema_name>
```

Windows で完全なスキーマを移行するには、実行可能ファイルに移動し、次のコマンドを呼び出します。

```
> ./runMTK.bat <schema_name>
```

どこで:

schema_name

schema_name は、移行するソースデータベース (toolkit.properties ファイルで指定) 内のスキーマの名前です。少なくとも 1 つの schema_name を含める必要があります。

注: 移行されたスキーマのデフォルトデータベースユーザーが自動的に移行される場合、そのようなカスタムプロファイルが存在する場合、デフォルトデータベースユーザーのカスタムプロファイルも移行されます。カスタムプロファイルは、ユーザーが作成したプロファイルです。たとえば、カスタムプロファイルは、Oracle プロファイル DEFAULT および MONITORING_PROFILE を除外します。

コマンド名の後にスキーマ名のカンマ区切りリストを続けることで、複数のスキーマを移行できます。

Linux では、次のコマンドを実行します。

```
$ ./runMTK.sh <schema_name1>,<schema_name2>,<schema_name3>
```

Windows では、次のコマンドを実行します。

```
> ./runMTK.bat <schema_name1>,<schema_name2>,<schema_name3>
```

7.2 Oracle 以外のソースデータベースからの移行

ソースデータベースタイプとターゲットデータベースタイプを指定しない場合、Postgres はソースデータベースが Oracle、ターゲットデータベースが Advanced Server であると想定します。

Migration Toolkit を呼び出すには、コマンドウィンドウを開き、実行可能ファイルに移動して、次のコマンドを呼び出します。

```
$ ./runMTK.sh -sourcedbtype <db_type> -targetdbtype <target_type>
[options, ...] schema_name;
```

どこで:

`-sourcedbtype source_type`

`source_type` は、ソースデータベースのサーバータイプを指定します。 `source_type` は大文字と小文字を区別しません。デフォルトでは、 `source_type` は `oracle` です。 `source_type` は次の値のいずれかです:

から移行するには:	以下を指定します。
オラクル	oracle (デフォルト値)
MySQL	mysql
SQL サーバー	sqlserver
Sybase	sybase
PostgreSQL	postgres または postgresql
Advanced Server	enterprisedb

`-targetdbtype target_type`

`target_type` はターゲットデータベースのサーバータイプを指定します。 `target_type` は大文字と小文字を区別しません。デフォルトでは、 `target_type` は `enterprisedb` です。 `target_type` は次の値のいずれかです:

移行するには:	以下を指定します。
Advanced Server	enterprisedb
PostgreSQL	postgres または postgresql

`schema_name`

`schema_name` は、移行したいソースデータベース (`toolkit.properties` ファイルで指定) 内のスキーマの名前です。少なくとも 1 つの `schema_name` を含める必要があります。

次の例は、Linux システム上の MySQL データベースから Advanced Server ホストに HR という名前のスキーマ

(テーブル定義とテーブルコンテンツ) を移行します。コマンドには `-sourcedbtype` および `-targetdbtype` オプションが含まれていることに注意してください:

```
$ ./runMTK.sh -sourcedbtype mysql -targetdbtype enterprisedb HR
```

Windows では、次のコマンドを使用します。

```
> ./runMTK.bat -sourcedbtype mysql -targetdbtype enterprisedb HR
```

Migration Toolkit コマンドの最後にスキーマのコンマ区切りリストを含めることにより、ソースデータベースから複数のスキーマを移行できます。次の例では、複数のスキーマ (HR および ACCTG という名前) を MySQL データベースから PostgreSQL データベースに移行します。

Linux では、次のコマンドを使用して、MySQL データベースから複数のスキーマを移行します。

```
$ ./runMTK.sh -sourcedbtype mysql -targetdbtype postgres HR,ACCTG
```

Windows では、次のコマンド形式を使用します。

```
> ./runMTK.bat -sourcedbtype mysql -targetdbtype postgres HR,ACCTG
```


第 8 章

Migration Toolkit コマンドオプション

Migration Toolkit を実行するときに移行オプションを追加して、移行の詳細を簡単に制御します。たとえば、データベース内のすべてのスキーマを移行するには、コマンドに `-allSchemas` オプションを追加します。

```
$ ./runMTK.sh -allSchemas
```

以下のセクションには、Migration Toolkit で動作する各コマンドオプションの参照資料が含まれています。オプションは、動作によってグループ化されます。次の表に、Migration Toolkit のオプションとそれらがグループ化されているセクションを示します。

機能:	関連オプション:
オフライン移行オプション _	<code>-offlineMigration</code>
インポートオプション _	<code>-sourcedbtype</code> 、 <code>-targetdbtype</code> 、 <code>-schemaOnly</code> 、 <code>-dataOnly</code>
スキーマ作成オプション _	<code>-dropSchema</code> 、 <code>-targetSchema</code>
スキーマオブジェクト選択オプション _	<code>-allTables</code> 、 <code>-tables</code> 、 <code>-constraints</code> 、 <code>-ignoreCheckConstFilter</code> 、 <code>-skipCKConst</code> 、 <code>-skipFKConst</code> 、 <code>-skipColDefaultClause</code> 、 <code>-indexes</code> 、 <code>-triggers</code> 、 <code>-allViews</code> 、 <code>-views</code> 、 <code>-allSequences</code> 、 <code>-sequences</code> 、 <code>-allProcs</code> 、 <code>-procs</code> 、 <code>-allFuncs</code> 、 <code>-funcs</code> 、 <code>-checkFunctionBodies</code> 、 <code>-allPackages</code> 、 <code>-packages</code> 、 <code>-allDomains</code> 、 <code>-allQueues</code> 、 <code>-queues</code> 、 <code>-allRules</code> 、
移行オプション _	<code>-truncLoad</code> 、 <code>-enableConstBeforeDataLoad</code> 、 <code>-retryCount</code> 、 <code>-safeMode</code> 、 <code>-fastCopy</code> 、 <code>-analyze</code> 、 <code>-vacuumAnalyze</code> 、 <code>-replaceNullChar</code> 、 <code>-copyDelimiter</code> 、 <code>-batchSize</code> 、 <code>-cpBatchSize</code> 、 <code>-lobBatchSize</code> 、 <code>-fetchSize</code> 、 <code>-filterProp</code> <code>-customColTypeMappingFile</code> 、 <code>-customColTypeMappingFile</code>
“Oracle 固有のオプション” _	<code>-allUsers</code> 、 <code>-users</code> 、 <code>-allProfiles</code> 、 <code>-profiles</code> 、 <code>-importPartitionAsTable</code> 、 <code>-objectTypes</code> 、 <code>-copyViaDBLinkOra</code> 、 <code>-allDBLinks</code> 、 <code>-allSynonyms</code> 、 <code>-allPublicSynonyms</code> 、 <code>-allPrivateSynonyms</code> 、 <code>-useOraCase</code> 、 <code>-skipUserSchemaCreation</code>
“その他のオプション” _	<code>-help</code> 、 <code>-logDir</code> 、 <code>-logFileCount</code> 、 <code>-logFileSize</code> 、 <code>-logBadSQL</code> <code>-verbose</code> 、 <code>-version</code>

8.1 オフライン移行オプション

コマンドラインで `-offlineMigration` オプションを指定すると、Migration Toolkit は*オフライン*移行を実行します。オフライン移行中、Migration Toolkit は選択された各オブジェクトの定義を読み取り、後で実行されるときに Postgres の各オブジェクトを複製する SQL スクリプトを作成します。

注:次の例は、Linux での移行ツールキットの呼び出しを示しています。Windows で Migration Toolkit を起動するには、`runMTK.sh` コマンドの代わりに `runMTK.bat` コマンドを使用します。

スキーマとデータの両方のオフライン移行を実行するには、`-offlineMigration` キーワードに続けてスキーマ名を指定します:

```
$ ./runMTK.sh -offlineMigration <schema_name>
```

各データベースオブジェクト定義は、ユーザーのホームフォルダー内のスキーマ名とオブジェクトタイプから派生した名前を持つ個別のファイルに保存されます。代替ファイルの宛先を指定するには、`-offlineMigration` オプションの後にディレクトリ名を含めます:

```
$ ./runMTK.sh -offlineMigration <file_dest> <schema_name>
```

スキーマオブジェクトのみのオフライン移行（空のテーブルの作成）を実行するには、Migration Toolkit を呼び出すときに `-offlineMigration` キーワードに加えて `-schemaOnly` キーワードを指定します。

```
$ ./runMTK.sh -offlineMigration -schemaOnly <schema_name>
```

データのみをオフライン移行を実行するには（スキーマオブジェクト定義を省略）、Migration Toolkit を呼び出すときに `-dataOnly` キーワードと `-offlineMigration` キーワードを指定します

```
$ ./runMTK.sh -offlineMigration -dataOnly <schema_name>
```

デフォルトでは、データは **COPY** 形式で書き込まれます。データをプレーンな SQL 形式で書き込むには、`-safeMode` キーワードを含めます:

```
$ ./runMTK.sh -offlineMigration -dataOnly -safeMode <schema_name>
```

デフォルトでは、テーブルデータを含むオフライン移行を実行すると、テーブルごとに個別のファイルが作成されます。複数のテーブルのデータを含む単一のファイルを作成するには、`-singleDataFile` キーワードを指定します:

```
./runMTK.sh -offlineMigration -dataOnly -singleDataFile -safeMode  
<schema_name>
```

注: `-singleDataFile` オプションは、プレーン SQL 形式のデータを移行する場合にのみ使用できます。`-singleDataFile` オプションを含める場合は、`-safeMode` キーワードを含める必要があります。

8.1.1 オフライン移行スクリプトの実行

edb-psql（または psql）コマンドラインを使用して、オフライン移行中に生成されたスクリプトを実行できます。次の例では、スキーマ（* hr という名前）を *Advanced Server* に保存されている新しいデータベース（acctg * という名前）に復元する方法について説明します。

1. createdb コマンドを使用して acctg データベースを作成し、そこに移行したデータベースオブジェクトを復元します。

```
createdb -U enterprisedb acctg
```

2. edb-psql を使用して新しいデータベースに接続します。

```
edb-psql -U enterprisedb acctg
```

3. /i メタコマンドを使用して、オブジェクト定義を作成する移行スクリプトを呼び出します。

```
acctg=# /i ./mtk_hr_ddl.sql
```

4. -offlineMigration コマンドに -singleDataFile キーワードが含まれている場合、mtk_hr_data.sql スクリプトには、新しいターゲットデータベース内のすべてのオブジェクトを再作成するために必要なコマンドが含まれます。次のコマンドでデータベースを作成します。

```
acctg=# /i ./mtk_hr_data.sql
```

8.2 インポートオプション

デフォルトでは、Migration Toolkit はソースデータベースを Oracle、ターゲットデータベースを Advanced Server と見なします。-sourcedbtype および -targetdbtype キーワードを含めて、デフォルト以外のソースまたはターゲットデータベースを指定します。

デフォルトでは、Migration Toolkit はスキーマの移行時にデータとオブジェクト定義の両方をインポートします。または、データまたはオブジェクト定義のいずれかをインポートすることを選択できます。

-sourcedbtype <source_type>

-sourcedbtype オプションは、ソースデータベースタイプを指定します。<source_type> は、mysql、oracle、sqlserver、sybase、postgresql または enterprisedb のいずれかの値です。<source_type> は大文字と小文字を区別しません。デフォルトでは、<source_type> は oracle です。

-targetdbtype <target_type>

-targetdbtype オプションは、ターゲットデータベースタイプを指定します。<target_type> は、enterprisedb、postgres、または postgresql のいずれかの値です。<target_type> は大文字と小文字を区別しません。デフォルトでは、<target_type> は enterprisedb です。

-schemaOnly

このオプションは、スキーマ定義をインポートし、選択したすべてのスキーマオブジェクトをターゲットデータベースに作成します。このオプションは -dataOnly オプションと併用できません。

-dataOnly

このオプションはデータのみをコピーします。-tables オプションとともに使用すると、Migration Toolkit は選択したテーブルのデータのみをインポートします（以下の使用法の詳細を参照）。このオプションは -schemaOnly オプションと一緒に使用できません。

8.3 スキーマ作成オプション

デフォルトでは、**Migration Toolkit** はソーススキーマオブジェクトまたはデータ、あるいはその両方を同じ名前のスキーマにインポートします。ターゲットスキーマが存在しない場合、**Migration Toolkit** は新しいスキーマを作成します。または、`-targetSchema option` を使用してカスタムスキーマ名を指定することもできます。次のオプションを使用して、既存のスキーマを削除し、新しいスキーマを作成することを選択できます。

`-dropSchema [true|false]`

`true` に設定すると、**Migration Toolkit** は既存のスキーマ（およびそのスキーマ内のオブジェクト）を削除し、新しいスキーマを作成します。（デフォルトでは、`-dropSchema` は `false` です）。

`-targetSchema schema_name>`

移行したスキーマの名前を指定するには、`-targetSchema` オプションを使用します。複数のスキーマを移行する場合、各スキーマの名前を、間にスペース文字を入れずにコンマ区切りリストで指定します。コマンドラインに `-targetSchema` オプションが含まれていない場合、新しいスキーマの名前はソーススキーマの名前と同じになります。

ターゲットスキーマ名として `information-schema`、`dbo`、`sys`、または `pg_catalog` を指定することはできません。これらのスキーマ名は、**Advanced Server** のメタデータストレージ用に予約されています。

8.4 スキーマオブジェクト選択オプション

次のオプションを使用して、移行する特定のスキーマオブジェクトを選択します。

`-allTables`

ソーススキーマからすべてのテーブルをインポートします。

`-tables <table_list>`

選択したテーブルをソーススキーマからインポートします。 `<table_list>` はコンマで区切られたテーブル名のリストです（間にスペース文字は挿入されません）（たとえば、`-tables emp、dept、acctg`）。

`-constraints`

テーブル制約をインポートします。このオプションは、スキーマ全体をインポートする場合、または `-allTables` または `-tables <table_list>` オプションを指定する場合にのみ有効です。

`-ignoreCheckConstFilter`

デフォルトでは、Migration Toolkit は、Sybase データベースからのチェック制約とデフォルト句の移行を実装しません。Sybase データベースから制約とデフォルト句を移行するために `-constraints` パラメーターを指定する場合は、`-ignoreCheckConstFilter` パラメーターを含めます。

`-skipCKConst`

チェック制約の移行を省略します。このオプションは、ターゲットデータベースでサポートされていない（ソースデータベースの）組み込み関数に基づくチェック制約を移行する場合に役立ちます。

このオプションは、スキーマ全体をインポートする場合、または `-allTables` または `-tables <table_list>` オプションが指定されている場合にのみ有効です。

`-skipFKConst`

外部キー制約の移行を省略します。このオプションは、スキーマ全体をインポートする場合、または `-allTables` または `-tables <table_list>` オプションが指定されている場合にのみ有効です。

`-skipColDefaultClause`

列 `DEFAULT` 句の移行を省略します。

`-indexes`

テーブルインデックスをインポートします。このオプションは、スキーマ全体をインポートするとき、または `-allTables` または `-tables <table_list>` オプションが指定されているときに有効です。

`-triggers`

テーブルトリガーをインポートします。このオプションは、スキーマ全体をインポートするとき、または `-allTables` または `-tables <table_list>` オプションが指定されているときに有効です。

-allViews

ソーススキーマからビューをインポートします。このオプションは、ソースから動的*および*マテリアライズドビューの両方を移行することに注意してください。（Oracle および Postgres のマテリアライズドビューがサポートされています。）

-views <view_list>

指定されたマテリアライズドビューまたは動的ビューをソーススキーマからインポートします。（Oracle および Postgres のマテリアライズドビューがサポートされています。） <view_list> は、ビュー名（たとえば、-views all_emp、mgmt_list、“3’のコンマ区切りリスト（スペース文字なし）です）¹）。

-allSequences

ソーススキーマからすべてのシーケンスをインポートします。

-sequences <sequence_list>

選択したシーケンスをソーススキーマからインポートします。 <sequence_list> は、シーケンス名のコンマ区切りリスト（間にスペース文字を入れない）です。

-allProcs

ソーススキーマからすべてのストアードプロシージャをインポートします。

-procs <procedures_list>

選択したストアードプロシージャをソーススキーマからインポートします。 <procedures_list> は、プロシージャ名のコンマ区切りリスト（間にスペース文字を入れない）です。

-allFuncs

ソーススキーマからすべての関数をインポートします。

-funcs <function_list>

選択した関数をソーススキーマからインポートします。 <function_list> はコンマで区切られた関数名のリストです（間にスペース文字は挿入されません）。

-checkFunctionBodies [true/false]

false の場合、関数作成中に関数本体の検証を無効にします（関数に前方参照が含まれる場合のエラーを回避するため）。デフォルト値は true です。

-allPackages

ソーススキーマからすべてのパッケージをインポートします。

-packages <package_list>

選択したパッケージをソーススキーマからインポートします。 <package_list> はコンマで区切られたパッケージ名のリストです（間にスペース文字は挿入されません）。

`-allDomains`

ソースデータベースからすべてのドメイン、列挙型、および複合型をインポートします。このオプションは、ソースとターゲットの両方が Postgres ホストに保存されている場合にのみ有効です。

`-allQueues`

ソーススキーマからすべてのキューをインポートします。これらは、DBMS_AQ および DBMS_AQADM 組み込みパッケージによって作成および管理されるキューです。 Oracle がソースデータベースの場合、`-objectTypes` オプションも指定する必要があります。 Advanced Server がソースデータベースの場合、`-allDomains` および `-allTables` オプションも指定する必要があります。（Oracle および Advanced Server キューがサポートされています。）

`-queues <queue_list>`

選択したキューをソーススキーマからインポートします。 <queue_list> は、キュー名のコンマ区切りリスト（間にスペース文字を入れない）です。これらは、DBMS_AQ および DBMS_AQADM 組み込みパッケージによって作成および管理されるキューです。 Oracle がソースデータベースの場合、`-objectTypes` オプションも指定する必要があります。 Advanced Server がソースデータベースの場合、`-allDomains` および `-allTables` オプションも指定する必要があります。（Oracle および Advanced Server キューがサポートされています。）

`-allRules`

ソースデータベースからすべてのルールをインポートします。このオプションは、ソースとターゲットの両方が Postgres ホストに保存されている場合にのみ有効です。

8.5 移行オプション

以下にリストされた移行オプションを使用して、移行プロセスの詳細を制御します。

`-loaderCount [<value>]`

`-loaderCount` オプションを使用して、データのインポート時に Migration Toolkit が使用する必要のある並列スレッドの数を指定します。このオプションは、ソースデータベースに大量のデータが含まれており、(Migration Toolkit を実行している) Postgres ホストにハイエンドの CPU および RAM リソースがある場合に特に役立ちます。ながら<value>ゼロ以外の正の数である可能性があります。 <value> CPU コアの数を超えないでください。デュアルコア CPU には、最適な <value> の 2 が必要です。

大きすぎる <value> を指定すると、Migration Toolkit が終了し、 'Out of heap space' エラーが生成される可能性があることに注意してください。

`-truncLoad`

新しいデータをインポートする前に、テーブルからデータを切り捨てます。このオプションは `-dataOnly` オプションと組み合わせてのみ使用できます。

`-enableConstBeforeDataLoad`

パーティション化されていないソーステーブルがパーティション化されたテーブルにマップされる場合、`-enableConstBeforeDataLoad` オプションを含めます。このオプションは、データ移行の前に、ターゲットテーブルのすべてのトリガー（個々のパーティションにデータをリダイレクトするトリガーを含む）を有効にします。 `-enableConstBeforeDataLoad` は、 `-truncLoad` パラメーターも指定されている場合にのみ有効です。

`-retryCount [<value>]`

複数スキーマの移行を実行している場合、スキーマ間の依存関係のために最初の移行試行中に移行に失敗したオブジェクトは、後続の移行中に正常に移行する場合があります。 `-retryCount` オプションを使用して、Migration Toolkit が最初の移行試行中に失敗したオブジェクトの移行を試行する回数を指定します。 0 より大きい <value> を指定します。デフォルト値は 2 です。

`-safeMode`

`-safeMode` オプションを含めると、Migration Toolkit は各行を移行済みとしてコミットします。移行がすべてのレコードの転送に失敗した場合、障害点の前に挿入された行はターゲットデータベースに残ります。

`-fastCopy`

`-fastCopy` オプションを含めると、Migration Toolkit が最適化された方法で COPY 操作を実行するために WAL ロギングをバイパスする必要があることを指定します。デフォルトでは無効です。 `-fastCopy` オプションを使用することを選択した場合、移行が中断された場合、移行されたデータは（ターゲットデータベースで）回復できない可能性があります。

`-replaceNullChar <value>`

Migration Toolkit は、値が NULL の列のインポートを適切にサポートします。

ただし、Migration Toolkit は、JDBC 接続プロトコルでの NULL 文字値 (埋め込みバイナリゼロ 0x00) のインポートをサポートしていません。NULL 文字を含むデータをインポートする場合、`-replaceNullChar` オプションを使用して、NULL 文字を単一の非 NULL の置換文字に置き換えます。置換文字を引用符またはアポストロフィで囲まないでください。

注:

- MTK は暗黙的に NULL 文字を空の文字列に置き換えます。
- `-replaceNullChar` オプションは `-copyViaDBLinkOra` オプションでは機能しません。

データが移行されたら、SQL ステートメントを使用して、`-replaceNullChar` で指定された文字をバイナリのゼロに置き換えます。

`-analyze`

ターゲットデータベースに対して Postgres ANALYZE 操作を呼び出すために `-analyze` オプションを含めます。オプティマイザーは、ANALYZE 操作によって収集された統計を参照し、その情報を利用して効率的なクエリプランを構築します。

`-vacuumAnalyze`

`-vacuumAnalyze` オプションを含めて、ターゲットデータベースに対して VACUUM および ANALYZE 操作の両方を呼び出します。オプティマイザーは、ANALYZE 操作によって収集された統計を参照し、その情報を利用して効率的なクエリプランを構築します。VACUUM 操作は、ターゲットデータベース内の無効なタプルによって占有されているストレージスペースを再利用します。

`-copyDelimiter`

テーブルデータをロードするときに、コピーコマンドで区切り文字として使用する単一の文字を指定します。デフォルト値は `'\t'` (タブ) です。

`-batchSize`

一括挿入のバッチサイズを指定します。有効な値は 1~1000 です。デフォルトのバッチサイズは 1000 です。Out of Memory 例外が発生した場合、`-batchSize` の値を減らします。

`-cpBatchSize`

COPY コマンドで使用するバッチサイズを MB 単位で指定します。0 より大きい値はすべて有効です。デフォルトのバッチサイズは 8 MB です。

`-lobBatchSize`

LOB データ型のバッチでロードする行数を指定します。BYTEA、BLOB、CLOB などのラージオブジェクトタイプ (LOB) 列を含むテーブルのデータ移行は、デフォルトで一度に 1 行ずつ実行されます。これ

は、個々の LOB 列が数百メガバイトのデータを保持する場合のヒープ領域不足エラーを回避するためです。LOB 列の平均データサイズが下限にある場合、各バッチの行数を 0 より大きい値で指定することにより、LOB バッチサイズをカスタマイズできます。

`-fetchSize`

`-fetchSize` オプションを使用して、結果セットでフェッチされる行の数を指定します。指定された `-fetchSize` が大きすぎる場合、Out of Memory 例外が発生する可能性があります。大きなテーブルを移行するときにこの落とし穴を避けるために `-fetchSize` オプションを含めます。デフォルトのフェッチサイズは JDBC ドライバーの実装に固有であり、データベースによって異なります。

MySQL ユーザーの注意:デフォルトでは、MySQL JDBC ドライバーは、テーブルにあるすべての行を単一のネットワーク往復でクライアントアプリケーション (Migration Toolkit) にフェッチします。この動作は、大きなテーブルで使用可能なメモリを簡単に超える可能性があります。`-fetchSize 1` エラーが発生した場合は、コマンドライン引数として `-fetchSize 1` を指定して、Migration Toolkit にテーブルデータを 1 行ずつ強制的にロードさせます。

`-filterProp <file_name>`

`<file_name>` は、**key = value** ペアの制約を含むファイルの名前を指定します。データベースから読み取られた各レコードは、制約に対して評価されます。制約を満たすものは移行されます。ペアの左側にはテーブル名がリストされます。テーブル名はスキーマで修飾しないでください。右側は、移行される各行に対して真でなければならない条件を指定します。たとえば、プロパティファイルに次の制約を含めます。

```
countries=country_id<>'AR'
```

AR に等しくない `country_id` 値を持つ国のみを移行します。この制約は国テーブルに適用されます。

`-customColTypeMapping <column_list>`

カスタム型マッピングを使用して、移行された列のデータ型を変更します。各ペアの左側は、正規表現で列を指定します。各ペアの右側には、列が想定するデータ型の名前を指定します。セミコロンで区切られた `<column_list>` に複数のペアを含めることができます。たとえば、名前が ID で終わる列をマッピングして INTEGER を入力するには、次のカスタムマッピングエントリを使用します。

```
.*ID=INTEGER
```

カスタムマッピングは、列がテーブル修飾されていない限り、条件に一致するすべてのテーブル列に適用されます。

'/' 文字はエスケープ文字列として機能します。'.' は正規表現の予約文字であるため、Linux では './' を使用して '.' 文字を表します。たとえば、カスタムマッピングを使用して、EMP テーブルの EMP_ID 列から行を選択するには、次のカスタムマッピングエントリを指定します。

```
EMP/./EMP_ID=INTEGER
```

Windows では、'/' を使用して '.' 文字を表します。

```
EMP/.EMP_ID=INTEGER
```

```
-customColTypeMappingFile <property_file>
```

<property_file> に複数のカスタムタイプマッピングを含めることができます。 `key = value` のペアで、ファイルの各エントリを個別の行に指定します。各ペアの左側は、正規表現で列を選択します。各ペアの右側には、列が想定するデータ型の名前を指定します。

8.6 Oracle 固有のオプション

次のオプションは、ソースデータベースが Oracle の場合にのみ適用されます。

-objectTypes

runMTK.sh コマンドの最後に指定されたスキーマリストからユーザー定義のオブジェクトタイプをインポートします。

-allUsers

ソースデータベースからすべてのユーザーとロールをインポートします。-allUsers オプションは、Oracle データベースから Advanced Server データベースに移行する場合にのみサポートされることに注意してください。

-users <user_list>

選択したユーザーまたはロールをソース Oracle データベースからインポートします。<user_list> は、ユーザー/ロール名（例: -users MTK, SAMPLE, acctg）のコンマ区切りリスト（間にスペース文字を入れない）です。-users オプションは、Oracle データベースから Advanced Server データベースに移行する場合にのみサポートされることに注意してください。

-allProfiles

ソースデータベースからすべてのカスタム（つまり、ユーザー作成）プロファイルをインポートします。DEFAULT や MONITORING_PROFILE などの他の Oracle 非カスタムプロファイルはインポートされません。

インポートされたプロファイルの場合、プロファイルに関連付けられている次のパスワードパラメータのみがインポートされます。

```

FAILED_LOGIN_ATTEMPTS

PASSWORD_LIFE_TIME

PASSWORD_REUSE_TIME

PASSWORD_REUSE_MAX

PASSWORD_LOCK_TIME

PASSWORD_GRACE_TIME

PASSWORD_VERIFY_FUNCTION

```

Oracle リソースパラメータなど、他のすべてのプロファイルパラメータはインポートされません。SRC_DB_USER で指定された Oracle データベースユーザーには、Oracle データディクショナリビュー DBA_PROFILES に対する SELECT 権限が必要です。

`-allProfiles` オプションは、Oracle データベースから Advanced Server データベースに移行する場合にのみサポートされることに注意してください。

`-profiles <profile_list>`

選択したカスタム（つまり、ユーザー作成）プロファイルをソース Oracle データベースからインポートします。 `<profile_list>` は、プロファイル名のコンマ区切りリスト（間にスペース文字を入れない）です（例: `-profiles ADMIN_PROFILE,USER_PROFILE`）。DEFAULT や MONITORING_PROFILE などの Oracle 非カスタムプロファイルはインポートされません。

`-allProfiles` オプションと同様に、パスワードパラメータのみがインポートされます。SRC_DB_USER で指定された Oracle データベースユーザーには、Oracle データディクショナリビュー DBA_PROFILES に対する SELECT 権限が必要です。

`-profiles` オプションは、Oracle データベースから Advanced Server データベースに移行する場合にのみサポートされることに注意してください。

`-importPartitionAsTable <table_list>`

`-importPartitionAsTable` パラメーターを含めて、Oracle ホスト上にあるパーティション化されたテーブルの内容を単一のパーティション化されていないテーブルにインポートします。 `<table_list>` は、テーブル名（たとえば、`-importPartitionAsTable emp,dept,acctg`）のコンマ区切りリスト（間にスペース文字を入れない）です。

`-copyViaDBLinkOra`

`dblink_ora` モジュールは、SQL レベルでの Advanced Server と Oracle の接続を提供します。 `dblink_ora` は、Advanced Server データベースのインストールの一部としてバンドルおよびインストールされます。 `dblink_ora` は、COPY API メソッドを使用してデータベース間でデータを転送します。このメソッドは、JDBC COPY メソッドよりもかなり高速です。

次の例では、`dblink_ora` COPY API を使用して、HR スキーマからすべての表を移行します。

```
$./runMTK.sh -copyViaDBLinkOra -allTables HR
```

ターゲット Advanced Server データベースに `dblink_ora` がインストールおよび構成されている必要があります。 `dblink_ora` の詳細については、“Oracle 開発者ガイドのデータベース互換性”を参照してください。 <<https://www.enterprisedb.com/edb-docs/d/edb-postgres-advanced-server/user-guides/database-compatibility-for-oracle-developers-guide/11/toc.html>>‘_。

`-allDBLinks [link_Name_1=password_1,link_Name_2=password_2,...]`

Oracle データベースリンクを移行するには、このオプションを選択します。ソースデータベース内の各リンク接続のパスワード情報は暗号化されているため、指定しない限り、ダミーパスワード（`edb`）に置き換えられます。

接続ユーザーのダミーパスワードとして `edb` を使用してすべてのデータベースリンクを移行するには:

```
$./runMTK.sh -allDBLinks HR
```

または、**name = value** ペアのコンマ区切りリスト（スペース文字を入れない）を使用して、各データベースリンクのパスワードを指定することもできます。ペアの左側にリンク名を指定し、右側にパスワード値を指定します。

コマンドラインで指定された実際のパスワードですべてのデータベースリンクを移行するには:

```
$./runMTK.sh -allDBLinks LINK_NAME1=abc, LINK_NAME2=xyz HR
```

Migration Toolkit は、EnterpriseDB で現在サポートされているデータベースリンクタイプのみを移行します。これには、パブリックおよびプライベートタイプの固定ユーザーリンクが含まれます。

-allSynonyms

すべてのパブリックシノニムとプライベートシノニムを Oracle データベースから Advanced Server データベースに移行するには、**-allSynonyms** オプションを含めます。同じ名前のシノニムがターゲットデータベースに既に存在する場合、既存のシノニムは移行されたバージョンに置き換えられます。

-allPublicSynonyms

すべてのパブリックシノニムを Oracle データベースから Advanced Server データベースに移行するには、**-allPublicSynonyms** オプションを含めます。同じ名前のシノニムがターゲットデータベースに既に存在する場合、既存のシノニムは移行されたバージョンに置き換えられます。

-allPrivateSynonyms

すべてのプライベートシノニムを Oracle データベースから Advanced Server データベースに移行するには、**-allPrivateSynonyms** オプションを含めます。同じ名前のシノニムがターゲットデータベースに既に存在する場合、既存のシノニムは移行されたバージョンに置き換えられます。

-useOraCase

Oracle データベースから Advanced Server データベースに移行するときに、すべてのデータベースオブジェクトの Oracle のデフォルトの大文字の命名規則を保持するために、**-useOraCase** オプションを含めます。

テーブル、ビュー、シーケンス、プロシージャ、関数、トリガー、パッケージなどの大文字の命名規則は保持されます。これらのデータベースオブジェクトの場合、大文字の命名規則は a) データベースオブジェクトの名前、b) 列名、テーブルとビューのキー名、インデックス名、制約名など、c) ビューの **SELECT** 列リスト、d) プロシージャまたは関数ヘッダーの一部であるパラメーター名。

注: プロシージャ、関数、トリガー、またはパッケージのプロシージャコード本体内で、プログラムをエラーなしで適切に実行するには、識別子参照を手動で編集する必要がある場合があります。このような修正は、発生した場合と発生していない場合がある識別子参照の適切な大文字小文字変換に関するものです。

注: **-useOraCase** オプションを指定する場合、**-skipUserSchemaCreation** オプションも指定する必要があります。詳細については、このセクションの **-skipUserSchemaCreation** オプションの説明

明を参照してください。

Migration Toolkit のデフォルトの動作（`-useOraCase` オプションを使用しない場合）は、データベースオブジェクト名が引用符で囲まらずに **Oracle** から抽出されることです（データベースオブジェクトが引用符で **Oracle** で明示的に作成されていない場合）。以下は、Migration Toolkit が `-offlineMigration` オプションを使用して生成したテーブルコマンドの一部です。

```
CREATE TABLE DEPT (
DEPTNO NUMBER(2) NOT NULL,
DNAME VARCHAR2(14),
LOC VARCHAR2(13)
);
ALTER TABLE DEPT ADD CONSTRAINT DEPT_PK PRIMARY KEY (DEPTNO);
ALTER TABLE DEPT ADD CONSTRAINT DEPT_DNAME_UQ UNIQUE (DNAME);
```

このテーブルを Advanced Server に移行して作成すると、引用符で囲まれていないすべてのオブジェクト名が小文字に変換されるため、Advanced Server では次のようにテーブルが表示されます。

```
Table "edb.dept"
Column | Type | Modifiers
-----+-----+-----
deptno | numeric(2,0) | not null
dname | character varying(14) |
loc | character varying(13) |
Indexes:
"dept_pk" PRIMARY KEY, btree (deptno)
"dept_dname_uq" UNIQUE CONSTRAINT, btree (dname)
```

Advanced Server アプリケーションが、引用符で囲まれた大文字の識別子を使用して移行されたデータベースオブジェクトを参照している場合、データベースオブジェクト名が小文字になっているため、アプリケーションは失敗します。

```
usepostcase=# SELECT * FROM "DEPT";
ERROR: relation "DEPT" does not exist
LINE 1: SELECT * FROM "DEPT";
```

アプリケーションが引用符付きの大文字の識別子を使用している場合は、`-useOraCase` オプションを使用して移行を実行します。DDL は、すべてのデータベースオブジェクト名を引用符で囲みます。

```
CREATE TABLE "DEPT" (
"DEPTNO" NUMBER(2) NOT NULL,
"DNAME" VARCHAR2(14),
"LOC" VARCHAR2(13)
);
ALTER TABLE "DEPT" ADD CONSTRAINT "DEPT_PK" PRIMARY KEY ("DEPTNO");
ALTER TABLE "DEPT" ADD CONSTRAINT "DEPT_DNAME_UQ" UNIQUE ("DNAME");
```

このテーブルを Advanced Server に移行して作成すると、すべてのオブジェクト名は大文字で維持されるため、次のように Advanced Server にテーブルが表示されます。

```
Table "EDB.DEPT"
Column | Type | Modifiers
-----+-----+-----
DEPTNO | numeric(2,0) | not null
DNAME | character varying(14) |
LOC | character varying(13) |
Indexes:
"DEPT_PK" PRIMARY KEY, btree ("DEPTNO")
"DEPT_DNAME_UQ" UNIQUE CONSTRAINT, btree ("DNAME")
```

アプリケーションは、引用符で囲まれた大文字の名前を使用してオブジェクトにアクセスできます。

```
useoracase=# SELECT \* FROM "DEPT";
DEPTNO | DNAME | LOC
-----+-----+-----
10 | ACCOUNTING | NEW YORK
20 | RESEARCH | DALLAS
30 | SALES | CHICAGO
40 | OPERATIONS | BOSTON
(4 rows)
```

`-skipUserSchemaCreation`

Oracle ユーザーが移行されると、このロールがまだ存在しない場合、この Oracle ユーザーのターゲットデータベースサーバーにロール（ユーザー名）が作成されます。ロール名は小文字で作成されます。新しいロールが作成されると、同じ名前スキーマも小文字で作成されます。

`-skipUserSchemaCreation` オプションを指定すると、移行された Oracle ユーザー名のこの自動スキーマ作成が防止されます。このオプションは、小文字のスキーマ名と大文字のスキーマ名を除き、同じ名前の 2 つのスキーマの作成を防ぐために `-useOraCase` オプションが指定されている場合に特に便利です。`-useOraCase` オプションを指定すると、Migration Toolkit の起動時にオプションリストに従って指定されたソーススキーマの大文字の Oracle 命名規則でスキーマが作成されます。

したがって、`-skipUserSchemaCreation` オプションなしで `-useOraCase` オプションを指定した場合、ターゲットデータベースは、一方が小文字でもう一方が大文字の同じ名前の 2 つのスキーマを持つことになります。`-useOraCase` オプションが `-skipUserSchemaCreation` オプションとともに指定された場合、ターゲットデータベースは大文字のスキーマのみを持つことになります。

8.7 その他のオプション

以下にリストされている移行オプションを使用して、Migration Toolkit のヘルプとバージョン情報を表示します。このセクションのオプションを使用して、Migration Toolkit のフィードバックとログオプションを制御することもできます。

`-help`

アプリケーションのコマンドラインの使用情報を表示します。

`-logDir <log_path>`

ログファイルが書き込まれる場所を指定するには、このオプションを含めます。<log_path> は、アプリケーションログファイルが保存されるパスを表します。デフォルトでは、Linux ではログファイルは次の場所に書き込まれます。

`$HOME/.enterprisedb/migration-toolkit/logs`

Windows では、ログファイルは次の場所に保存されます。

`%HOMEDRIVE%%HOMEPATH%/.enterprisedb/migration-toolkit/logs`

`-logFileCount <file_count>`

ログファイルのローテーションで使用されるファイルの数を指定するには、このオプションを含めます。値 0 を指定すると、ログファイルのローテーションが無効になり、単一のログファイルが作成されます (logFileSize オプションを使用して指定された値に達すると切り捨てられます)。<file_count> は 0 以上でなければなりません。デフォルトは 20 です。

`-logFileSize <file_size>`

このオプションを含めると、新しいログファイルにローテーションする前に最大ファイルサイズ制限 (MB 単位) を指定できます。<file_size> は 0 より大きくなければなりません。デフォルトは 50 MB です。

`-logBadSQL`

失敗したオブジェクトのスキーマ定義 (DDL スクリプト) をファイルに保存するには、このオプションを含めます。ファイルは、エラーログに使用されるのと同じパスの下に保存され、`mtk_bad_sql_<schema_name_timestamp>.sql` という形式で名前が付けられます。schema_name はスキーマの名前、timestamp は移行のタイムスタンプですツールキットの実行。

`-verbose [on|off]`

標準出力にアプリケーションログメッセージを表示します (デフォルトでは、詳細は on です)。

`-version`

Migration Toolkit バージョンを表示します。

8.8 例

次の例は、Oracle から Advanced Server への移行の実行を示しています。

以下は、`toolkit.properties` ファイルの内容です。

```
SRC_DB_URL=jdbc:oracle:thin:@192.168.2.6:1521:xe
SRC_DB_USER=edb
SRC_DB_PASSWORD=password
TARGET_DB_URL=jdbc:edb://localhost:5444/edb
TARGET_DB_USER=enterprisedb
TARGET_DB_PASSWORD=password
```

次のコマンドは、Migration Toolkit を呼び出します。

```
$ ./runMTK.sh EDB
```

```
Running EnterpriseDB Migration Toolkit (Build 48.0.0) ...
Source database connectivity info...
conn =jdbc:oracle:thin:@192.168.2.6:1521:xe
user =edb
password=*****\*
Target database connectivity info...
conn =jdbc:edb://localhost:5444/edb
user =enterprisedb
password=*****\*
Connecting with source Oracle database server...
Connected to Oracle, version 'Oracle Database 10g Express Edition
Release 10.2.0.1.0 - Production'
Connecting with target EnterpriseDB database server...
Connected to EnterpriseDB, version '9.4.0.0'
Importing redwood schema EDB...
Creating Schema...edb
Creating Sequence: NEXT_EMPNO
Creating Tables...
Creating Table: BAD_TABLE
MTK-15013: Error Creating Table BAD_TABLE
DB-42704: ERROR: type "binary_double" does not exist at position 58
-- CREATE TABLE BAD_TABLE (
-- F1 NUMBER NOT NULL,
-- Line 3: F2 BINARY_DOUBLE
-- ^
Creating Table: DEPT
Creating Table: EMP
Creating Table: JOBHIST
Creating Table: "MixedCase"
Creating Table: "lowercase"
Created 5 tables.
```

(次のページに続く)

(前のページからの続き)

```

Loading Table Data in 8 MB batches...
Loading Table: DEPT ...
[DEPT] Migrated 4 rows.
[DEPT] Table Data Load Summary: Total Time(s): 0.147 Total Rows: 4
Loading Table: EMP ...
[EMP] Migrated 14 rows.
[EMP] Table Data Load Summary: Total Time(s): 0.077 Total Rows: 14
Loading Table: JOBHIST ...
[JOBHIST] Migrated 17 rows.
[JOBHIST] Table Data Load Summary: Total Time(s): 0.042 Total Rows: 17
Total Size(MB): 9.765625E-4
Loading Table: "MixedCase" ...
["MixedCase"] Table Data Load Summary: Total Time(s): 0.098 Total Rows:0
Loading Table: "lowercase" ...
["lowercase"] Table Data Load Summary: Total Time(s): 0.066 Total Rows:0
Data Load Summary: Total Time (sec): 0.806 Total Rows: 35 Total
Size(MB): 0.001
Creating Constraint: DEPT_PK
Creating Constraint: DEPT_DNAME_UQ
Creating Constraint: EMP_PK
Creating Constraint: JOBHIST_PK
Creating Constraint: SYS_C008958
MTK-15001: Error Creating Constraint SYS_C008958
DB-42P01: com.edb.util.PSQLException: ERROR: relation "bad_table" does
not exist
Creating Constraint: EMP_REF_DEPT_FK
Creating Constraint: EMP_SAL_CK
Creating Constraint: JOBHIST_REF_DEPT_FK
Creating Constraint: JOBHIST_REF_EMP_FK
Creating Constraint: JOBHIST_DATE_CHK
Creating Trigger: USER_AUDIT_TRIG
Creating Trigger: EMP_SAL_TRIG
MTK-13009:Warning! Skipping migration of trigger DROP_TRIGGER, currently
non-table triggers are not supported in target database.
Creating View: SALESEMP
Creating Function: EMP_COMP
Creating Package: EMP_ADMIN
MTK-16005:Package Body is Invalid, Skipping...
Schema EDB imported with errors.
MTK-12001: The user/role migration failed due to insufficient
privileges.
Grant the user SELECT privilege on the following Oracle catalogs:
DBA_ROLES
DBA_USERS
DBA_TAB_PRIVS
DBA_PROFILES
DBA_ROLE_PRIVS
ROLE_ROLE_PRIVS

```

(次のページに続く)

(前のページからの続き)

```

DBA_SYS_PRIVS
One or more schema objects could not be imported during the migration
process. Please review the migration output for more details.
Migration logs have been saved to
/home/user/.enterprisedb/migration-toolkit/logs
***** Migration Summary *****
Sequences: 1 out of 1
Tables: 5 out of 6
Constraints: 9 out of 10
Triggers: 2 out of 3 (skipped 1)
Views: 1 out of 1
Functions: 1 out of 1
Packages: 1 out of 1
Total objects: 30
Successful count: 20
Failed count: 2
Skipped count: 1
Invalid count: 7
List of failed objects
=====
Tables
-----
1. EDB.BAD_TABLE
Constraints
-----
1. EDB.BAD_TABLE.SYS_C008958
List of invalid objects
=====
1. EDB.HIRE_CLERK (FUNCTION)
2. EDB.NEW_EMPNO (FUNCTION)
3. EDB.EMP_ADMIN (PACKAGE BODY)
4. EDB.EMP_QUERY (PROCEDURE)
5. EDB.EMP_QUERY_CALLER (PROCEDURE)
6. EDB.LIST_EMP (PROCEDURE)
7. EDB.SELECT_EMP (PROCEDURE)
*****

```

スキップされ、サポートされていないデータベースオブジェクトが省略されていることに注意してください。移行情報は、実行の最後に移行概要に要約されます。

第 9 章

移行エラー

移行プロセス中、移行の概要に移行の進行状況が表示されます。エラーが発生した場合、概要にはエラーに関する情報が表示されます。移行の概要もログファイルに書き込まれます。ログファイルのデフォルトの場所は次のとおりです。

Linux の場合:

```
$HOME/.enterprisedb/migration-toolkit/logs
```

Windows では、ログファイルは次の場所に保存されます。

```
%HOMEDRIVE%%HOMEPATH%/.enterprisedb/migration-toolkit/logs
```

Migration Toolkit の `-logdir *log_path*` オプションを使用して、代替ログファイルディレクトリを指定できます。

9.1 接続エラー

Migration Toolkit は *toolkit.properties* ファイルの情報を 사용합니다 ソースおよびターゲットデータベースに接続します。Migration Toolkit の使用時に発生する接続エラーのほとんどは、*toolkit.properties* ファイルで指定された情報に関連しています。次のセクションを使用して、一般的な接続エラーを特定し、それらの解決方法を学習します。

9.1.1 無効なユーザー名/パスワード

When I try to perform a migration from an Oracle database with Migration Toolkit, I get the following error:

```
MTK-11009: Error Connecting Database "Oracle"
DB-1017: java.sql.SQLException: ORA-01017: invalid username/password;
logon denied
```

(次のページに続く)

(前のページからの続き)

```
The user name or password specified in the toolkit.properties file is
not valid to use to connect to the Oracle source database.
```

このエラーを解決するには、 toolkit.properties file を編集し、 SRC_DB_USER および SRC_DB_PASSWORD プロパティで移行を実行するのに十分な権限を持つ有効なユーザーの名前とパスワードを指定します。

9.1.2 接続が拒否されました:致命的:パスワード

When I try to perform a migration with Migration Toolkit, I get the following error:

```
MTK-11009: Error Connecting Database "Advanced Server"
DB-28P01: com.edb.util.PSQLException: FATAL: password authentication
failed for user "*name*"
```

toolkit.properties ファイルで指定されたユーザー名またはパスワードは、Postgres データベースへの接続に使用できません。

このエラーを解決するには、 toolkit.properties ファイルを編集し、 TARGET_DB_USER および TARGET_DB_PASSWORD プロパティで移行を実行するのに十分な権限を持つ有効なユーザーの名前とパスワードを指定します。

9.1.3 例外:ORA-28000:アカウントはロックされています

When I try to perform a migration from an Oracle database with Migration Toolkit, I get the following error message:

```
MTK-11009: Error Connecting Database "Oracle"
DB-28000: java.sql.SQLException: ORA-28000: the account is locked
The Oracle account associated with the user name specified in the
toolkit.properties file is locked.
```

このエラーを解決するには、Oracle サーバーでユーザーアカウントのロックを解除するか、 toolkit.properties ファイルを編集して、 SRC_DB_USER で移行を実行するのに十分な権限を持つ有効なユーザーの名前とパスワードを指定し、 SRC_DB_PASSWORD パラメーター。

9.1.4 例外:oracle.jdbc.driver.OracleDriver

When I try to perform a migration with Migration Toolkit, the migration fails and I get the error message:

```
MTK-11009: Error Connecting Database "Oracle"
java.lang.ClassNotFoundException: oracle.jdbc.driver.OracleDriver
```

Migration Toolkit を使用する前に、移行元のデータベースに適した JDBC ドライバーをダウンロードしてインストールする必要があります。 [ソース固有のドライバのインストールを参照してください](#) 完全な指示。

9.1.5 I / O 例外:ネットワークアダプターは接続を確立できませんでした

When I try to perform a migration with Migration Toolkit, I get the following error:

```
MTK-11009: Error Connecting Database "Oracle"
DB-17002: java.sql.SQLException: Io exception: The Network Adapter could
not establish the connection
```

toolkit.properties ファイルで指定されたソースデータベースの* **JDBC URL** *に無効な接続プロパティが含まれています。

このエラーを解決するには、 toolkit.properties ファイルを編集します<building_toolkit.properties_file> “SRC_DB_URL” プロパティでソースデータベースの有効な接続情報を指定します。 SRC_DB_URL 値はデータベース固有であることに注意してください。ソースデータベースタイプの詳細については、目次を参照してください。

9.1.6 例外:“ターゲット “データベースに指定された URL は無効です

When I try to perform a migration with Migration Toolkit, I get the following error:

```
MTK-10045: The URL specified for the "target" database is invalid.
Check the connectivity credentials.
```

toolkit.properties ファイルで指定されたターゲットデータベース（Advanced Server）の JDBC URL に無効な接続プロパティが含まれています。

このエラーを解決するには、toolkit.properties ファイルを編集し、TARGET_DB_URL プロパティでターゲットデータベースの有効な接続情報を指定します。Advanced Server の JDBC URL の形成については、 [Advanced Server URL の定義を参照してください](#) 。

9.2 移行エラー

Migration Toolkit がターゲットおよびソースデータベースサーバーに正常に接続した後、次のエラーが発生する場合があります。

9.2.1 エラー:予想される最後の列の後の追加データ

テーブルをオンラインで移行すると、エラーメッセージが表示されます:

```
MTK-17001: Error Loading Data into Table: *table_name*
DB-22P04: com.edb.util.PSQLException: ERROR: extra data after last
expected column
Where: COPY *table_name*, line 5: "50|HR|LOS|ANGELES"
```

このエラーは、`*table_name*` の列のデータに区切り文字が含まれている場合に発生します。このエラーを修正するには、区切り文字をテーブルの内容にない文字に変更します。

注: この例では、パイプ文字 (|) が最後の列への挿入を意図したテキスト LOS|ANGELES にあり、移行ツールキットは `-copyDelimiter '|'` オプションを使用して実行されます。エラーになります。

9.2.2 テーブルへのデータ読み込みエラー: *TABLE_NAME*

データのための移行を実行すると、次のエラーが表示されます:

```
MTK-17001: Error Loading Data into Table: *TABLE_NAME*
DB-42P01: com.edb.util.PSQLException: ERROR: relation
"*schema*.\ *table_name*" does not exist
*I also get the error:*
Trying to reload table: *TABLE_NAME* through bulk inserts with a batch
size of 100
MTK-17001: Error Loading Data into Table: *TABLE_NAME*
DB-42P01: com.edb.util.PSQLException: ERROR: relation
"*schema*.\ *table_name*" does not exist
```

データを移行する前に、ターゲットデータベースのデータを受信するテーブルを作成する必要があります。 (*TABLE_NAME* という名前の) テーブルがターゲットデータベースに存在することを確認します。必要に応じてテーブルを作成し、データ移行を再試行してください。

9.2.3 制約 CONS_NAME_FK の作成エラー

インデックスと制約を含むテーブルの移行を実行すると、次のエラーメッセージが表示されます。

```

MTK-15001: Error Creating Constraint EMP_DEPT_FK
DB-42P01: com.edb.util.PSQLException: ERROR: relation "hr.departments"
does not exist
Creating Constraint: EMP_JOB_FK
MTK-15001: Error Creating Constraint EMP_JOB_FK
DB-42P01: com.edb.util.PSQLException: ERROR: relation "hr.jobs" does not exist
Creating Constraint: EMP_MANAGER_FK
Schema HR imported with errors.
One or more schema objects could not be imported during the migration
process. Please review the migration output for more details.
Migration logs have been saved to
/home/user/.enterprisedb/migration-toolkit/logs
***** Migration Summary *****
Tables: 1 out of 1
Constraints: 4 out of 6
Total objects: 7
Successful count: 5
Failed count: 2
Invalid count: 0
List of failed objects
=====
Constraints
-----
1. HR.EMPLOYEES.EMP_DEPT_FK
2. HR.EMPLOYEES.EMP_JOB_FK

```

移行するテーブルには、ターゲットデータベースに存在しないテーブルの外部キー制約が含まれています。Migration Toolkit は、外部キー制約を省略してテーブルを作成します。

Migration Toolkit コマンドに `-skipFKConst` オプションを含めることで、エラーメッセージの生成を回避できます。

9.2.4 テーブルへのデータの読み込みエラー

テーブル定義は既に移行済みです。データをテーブルに移行しようとすると、エラーが発生します:

```

MTK-17001: Error Loading Data into Table: DEPARTMENTS
DB-22P04: com.edb.util.PSQLException: ERROR: extra data after last
expected column
Where: COPY departments, line 1: "10 Administration 200 1700"
Trying to reload table: DEPARTMENTS through bulk inserts with a batch
size of 100
MTK-17000: Batch entry 0 INSERT INTO hr.DEPARTMENTS VALUES (10,
'Administration', 200, 1700); was aborted. Call getNextException to see
the cause.
DB-42601: java.sql.BatchUpdateException: Batch entry 0 INSERT INTO

```

(次のページに続く)

(前のページからの続き)

```
hr.DEPARTMENTS VALUES (10, 'Administration', 200, 1700); was aborted.  
Call getNextException to see the cause., Skipping Batch  
MTK-17000:ERROR: INSERT has more expressions than target columns  
Position: 48  
[DEPARTMENTS] Table Data Load Summary: Total Time(s): 0.037 Total Rows:0  
Data Load Summary: Total Time (sec): 0.168 Total Rows: 0 Total Size(MB):0.0  
Schema HR imported with errors.
```

(ターゲットデータベース内の) テーブル定義が、移行されたデータと一致しません。ターゲットまたはソーステーブルを変更した場合は、テーブル定義とデータに互換性があることを確認してください。

エラー:タイプに対して値が長すぎます

テーブル定義は既に移行済みです。データをテーブルに移行しようとすると、次のエラーが表示されます。

```
MTK-17001: Error Loading Data into Table: DEPARTMENTS  
DB-22001: com.edb.util.PSQLException: ERROR: value too long for type  
character(1)  
Where: COPY departments, line 1, column location_id: "1700"  
Trying to reload table: DEPARTMENTS through bulk inserts with a batch  
size of 100  
MTK-17000: Batch entry 0 INSERT INTO hr.DEPARTMENTS VALUES (10,  
'Administration', 200, 1700); was aborted. Call getNextException to see  
the cause.  
DB-22001: java.sql.BatchUpdateException: Batch entry 0 INSERT INTO  
hr.DEPARTMENTS VALUES (10, 'Administration', 200, 1700); was aborted.  
Call getNextException to see the cause., Skipping Batch  
MTK-17000:ERROR: value too long for type character(1)
```

ターゲットデータベースの列は、移行されたデータを受け取るのに十分な大きさではありません。この問題は、移行後にテーブル定義が変更された場合に発生する可能性があります。列名 (この例では、location_id) は、Where: で始まる行で識別されます。

```
Where: COPY departments, line 1, column location_id:/"1700/"
```

問題を修正するには、列サイズを調整し、移行を再試行します。

エラー:thread:OutOfMemoryError の例外

When migrating from a MySQL database, I encounter the following error:

```
Loading Table: big_range ...  
Exception in thread "dataload_job_1" java.lang.OutOfMemoryError: Java  
heap space  
at com.mysql.jdbc.MysqlIO.nextRow(MysqlIO.java:1370)
```

(次のページに続く)

(前のページからの続き)

```
at com.mysql.jdbc.MysqlIO.readSingleRowSet(MysqlIO.java:2369)
at com.mysql.jdbc.MysqlIO.getResultSet(MysqlIO.java:451)
at com.mysql.jdbc.MysqlIO.readResultsForQueryOrUpdate(MysqlIO.java:2076)
at com.mysql.jdbc.MysqlIO.readAllResults(MysqlIO.java:1451)
at com.mysql.jdbc.MysqlIO.sqlQueryDirect(MysqlIO.java:1787)
at com.mysql.jdbc.Connection.execSQL(Connection.java:3277)
at com.mysql.jdbc.Connection.execSQL(Connection.java:3206)
at com.mysql.jdbc.Statement.executeQuery(Statement.java:1232)
at com.edb.dbhandler.mysql.Data.getTableData(Data.java:90)
at com.edb.DataLoader.loadDataInFastMode(DataLoader.java:594)
at com.edb.DataLoader.run(DataLoader.java:186)
at java.lang.Thread.run(Thread.java:722)
```

デフォルトでは、MySQL JDBC ドライバーは、1 つのネットワークラウンドトリップで、テーブルにあるすべての行をクライアントアプリケーション（Migration Toolkit）にフェッチします。この動作は、大きなテーブルで使用可能なメモリを簡単に超える可能性があります。

問題を修正するには、移行を再試行するときにコマンドライン引数として `-fetchSize 1` を指定します。

第 10 章

エラーコード

Migration Toolkit で問題が発生すると、エラーコードとメッセージの形式で通知が発行されます。

この章では、エラーコード、メッセージ、およびそれらの解決策について説明します。

各エラーコードは、接頭辞 **MTK-**で始まり、その後に 5 桁が続きます。最初の 2 桁はエラーの一般的な分類であるエラークラスを示します。最後の 3 桁は特定のエラー状態を表します。

次の表に、エラークラスを示します。

エラークラス	説明
00	正常終了
01	情報
02	警告
03	一般的なエラー
0x	予約済み
10	無効なユーザー入力
11	構成エラー
12	不十分な特権
13	サポートされていない機能
14	不足しているオブジェクト
15	スキーマの移行
16	手続き言語の移行
17	データの読み込み

特定のデータベースサーバーからエラーが報告された場合、このエラーメッセージの前に **DB-**が付きます。たとえば、Postgres データベースサーバーの既存のテーブルが原因でテーブルの作成が失敗した場合、エラーコード **42P07** がデータベースサーバーによって返されます。Migration Toolkit ログの特定のエラーは、**DB-42P07** として表示されます。

10.1 エラーコードの概要

次のセクションでは、Migration Toolkit のエラーコードを要約します。次の表の “エラーコード” 列には、Migration Toolkit のエラーコードがリストされています。[メッセージと解像度] 列には、エラーコードと共に表示されるメッセージが含まれます。メッセージは、エラーの原因と解決方法を説明しています。

Message and Resolution カラムの \$NAME は、実行時に適切な値に置き換えられる情報のプレースホルダーです。

10.1.1 クラス 02-警告

このクラスは警告メッセージを表します。

エラーコード	メッセージと解決策
MTK-02000	このクラスに関連し、特定のエラーコードバインディングを持たないすべての警告は、このエラーコードを使用します。
MTK-02001	“runMTK -help”を実行して、使用方法の詳細を確認します。
MTK-02002	警告！オフライン移行パス* \$ OFFLINE_PATH *は存在しません。スクリプトはユーザーのホームフォルダーの下に作成されます。

10.1.2 クラス 10-無効なユーザー入力

このクラスは、無効なユーザー入力値を表します。

エラーコード	メッセージと解決策
MTK-10000	このクラスに関連し、特定のエラーコードバインディングを持たないすべてのエラーは、このエラーコードを使用します。
MTK-10001	information_schema、dbo、sys、または pg_catalog をターゲットスキーマとして選択することはできません。これらは、メタデータ情報を* \$ DATABASE *に保存するために使用されます。
MTK-10002	-schemaOnly` オプションと-dataOnly オプションを同時に指定することはできません。
MTK-10003	escapeTabDelimiter オプションを使用する場合、copyDelimiter として '' / t '' が必要です。
MTK-10004	-truncateLoad オプションは、-dataOnly オプションでのみ指定できます。
MTK-10005	'-dataOnly' オプションは、-allTables / -tables オプションにのみ適用可能です。このオプションが設定されている場合、スキーマ DDL はコピーできません。

次のページに続く

表 1 – 前のページからの続き

エラーコード	メッセージと解決策
MTK-10006	-constraints、-indexes、および-triggers オプションは、-allTables / -tables オプションのコンテキストでのみ適用できます。
MTK-10007	-vacuumAnalyze `` オプションと-analyze `` オプションを同時に指定することはできません。
MTK-10008	-skipFKConst オプションは、-constraints オプションでのみ指定できます。
MTK-10009	-skipCKConst オプションは、-constraints オプションでのみ指定できます。
MTK-10010	-schemaOnly オプションと一緒に-fastCopy オプションを指定することはできません。
MTK-10011	-skipColDefaultClause を-dataOnly オプションと一緒に指定することはできません。
MTK-10012	-customColTypeMapping `` オプションと-customColTypeMappingFile `` オプションを同時に指定することはできません。
MTK-10013	デフォルトの日付時刻は、`yyyy-MM-dd_HH:mm:ss` の形式で指定する必要があります。時間部分はオプションです。時間を指定するには、アンダースコア記号 `_` が必要です。
MTK-10014	MySQL および Sybase を移行のターゲットデータベースとして選択することはできません。
MTK-10015	コピー区切り文字は 1 文字にする必要があります。
MTK-10017	バッチサイズの数値が無効です。1~50000 の数値を使用してください。
MTK-10018	コピーバッチサイズは 0 より大きくする必要があります。
MTK-10019	コピーバッチサイズの数値が無効です。0 より大きい数値を使用してください。
MTK-10020	フェッチサイズは 0 より大きくする必要があります。
MTK-10021	フェッチサイズの数値が無効です。0 より大きい数値を使用してください。
MTK-10022	1 つ以上のコマンドライン引数が無効です。
MTK-10023	-targetDBVersion 値は、`* major *`。`/ * minor *` パターンを使用して指定する必要があります。
MTK-10024	-targetDBVersion は、-offlineMigration モードでのみ使用できます。
MTK-10025	オプション (-constraints / -indexes / -triggers / -tables / -views / -sequences / -procs / -funcs / -packages / -synonyms) は複数のスキーマオプションでは使用できません。
MTK-10026	-allSchemas オプションは、コマンドラインオプションリストの最後のオプションとして指定する必要があります。
MTK-10027	無効なコマンドライン引数を指定しました。`runMTK -help` を実行して、使用方法の詳細を確認します。
MTK-10028	-replaceNullChar を-schemaOnly オプションと一緒に指定することはできません。
MTK-10029	-nullReplacementChar は、-replaceNullChar オプションでのみ指定できます。
MTK-10030	-ignoreCheckConstFilter は、-constraints オプションでのみ指定できます。
MTK-10031	-enableConstBeforeDataLoad は、-truncLoad オプションでのみ指定できます。
MTK-10032	retryCount 値は 0 より大きい必要があります。
MTK-10033	retryCount の数値が無効です。0 より大きい数値を使用してください。

次のページに続く

表 1 – 前のページからの続き

エラーコード	メッセージと解決策
MTK-10034	loaderCount 値は 0 より大きい必要があります。
MTK-10035	loaderCount の数値が無効です。0 より大きい数値を使用してください。
MTK-10036	COPY 形式のオフラインデータ移行に singleDataFile オプションを使用できません。
MTK-10037	COPY 形式でのオフラインデータ移行は、PostgreSQL または EnterpriseDB (EDB Postgres Advanced Server) がターゲットデータベースである場合にのみサポートされます。
MTK-10038	The <i>\$SCHEMA</i> cannot be used as schema name in <i>\$DATABASE</i> . Choose a different schema name via -targetSchema option.
MTK-10039	ログファイルのサイズは 0 より大きくする必要があります。
MTK-10040	ログファイルの数は 0 以上である必要があります。
MTK-10041	オフライン移行は、スキーマのみのオプションでのみ使用できます。
MTK-10042	copyViaDBLinkOra オプションは、Oracle から EnterpriseDB にデータをコピーする場合にのみ使用できます。
MTK-10043	-recreateConst オプションは、-dataOnly オプションでのみ指定できます。
MTK-10044	The <i>\$DATABASE</i> database type is not supported by Migration Toolkit. Specify a valid database type string (i.e., EnterpriseDB, Postgres, Oracle, SQLServer, Sybase, or MySQL).
MTK-10045	The URL specified for the <i>\$DATABASE</i> database is invalid. Check the connectivity credentials.
MTK-10046	-escapeKeywords 値は、true または false です。
MTK-10047	-useOraCase オプションは、Oracle からの移行にのみ使用できます。
MTK-10048	Postgres のエクスポートされたスナップショット ID は無効です。
MTK-10049	Oracle データベースに指定された URL は、dblink_oracle ではサポートされていません。接続資格情報を確認し、有効な URL を提供してください。
MTK-10050	スキーマ* \$ SCHEMA *がソースデータベースに見つかりません。
MTK-10051	skipUserSchemaCreation オプションは、Oracle から EnterpriseDB へのユーザー移行にのみ使用できます。

10.1.3 クラス 11-構成の問題

このクラスは、toolkit.properties ファイルまたは Migration Toolkit で使用されるその他の構成ファイルで提供される無効な構成設定を表します。

エラーコード	メッセージと解決策
MTK-11000	このクラスに関連し、特定のエラーコードバインディングを持たないすべてのエラーは、このエラーコードを使用します。
MTK-11001	テーブルフィルタ句を含むプロパティファイルを読み込みません。
MTK-11002	カスタムタイプマッピングファイルをロードできませんでした。理由: * \$ REASON *。
MTK-11003	カスタムタイプマッピングファイルに無効なマッピングエントリが含まれています。
MTK-11004	カスタムタイプマッピングファイルには、マッピングエントリが含まれていません。
MTK-11005	ソースデータベースの接続情報は利用できません。
MTK-11006	ターゲットデータベースの接続情報は利用できません。
MTK-11007	ログディレクトリを作成できません。 * \$ PATH *。
MTK-11008	テーブル列フィルタリストを含むプロパティファイルを読み込みません。
MTK-11009	データベースの接続エラー * \$ DATABASE *。
MTK-11010	Error loading \$DBLINKORA_FILE file.
MTK-11011	DBLink Ora モジュールのロード中にエラーが発生しました。 * \$ DBLINKORA_MODULE *。ターゲットの EnterpriseDB サーバーに dblink_ora がインストール/構成されていることを確認します。dblink_ora モジュールのインストールと構成の詳細については、『Oracle 開発者ガイドのデータベース互換性ガイド』を参照してください。
MTK-11012	指定された DBLink_Ora モジュールのロード中にエラーが発生しました。 * \$ DBLINKORA_MODULE *。ターゲットの EnterpriseDB サーバーに dblink_ora がインストール/構成されていることを確認します。dblink_ora モジュールのインストールと構成の詳細については、『Oracle 開発者ガイドのデータベース互換性ガイド』を参照してください。
MTK-11013	DBLinkOra モジュールをロードできませんでした。
MTK-11014	DBLinkOra への接続エラー。
MTK-11015	接続資格情報ファイル * \$ TOOLKIT_PROP_FILE * は安全ではなく、グループ/他のユーザーはアクセスできません。このファイルにはプレーンパスワードが含まれており、Migration Toolkit の所有者ユーザーのみに制限する必要があります。

10.1.4 クラス 12-不十分な特権

このクラスは、データベースユーザー/ロール情報をロードするための不十分な特権エラーを表します。

エラーコード	メッセージと解決策
MTK-12000	このクラスに関連し、特定のエラーコードバインディングを持たないすべてのエラーは、このエラーコードを使用します。
MTK-12001	権限が不十分なため、ユーザー/ロールの移行に失敗しました。 Oracle カタログ DBA_ROLES、DBA_USERS、DBA_TAB_PRIVS、DBA_PROFILES、DBA_ROLE_PRIVS、ROLE_ROLE_PRIVS、DBA_SYS_PRIVS に対する SELECT 特権をユーザーに付与します。
MTK-12002	権限が不十分なため、権限の移行に失敗しました。ユーザーに Oracle カタログ dba_tab_privs に対する SELECT 権限を付与します。

10.1.5 クラス 13-サポートされていない機能

このクラスは、サポートされていないオブジェクトおよび句の移行に関連するエラーを表します。ターゲットデータベースが指定されたオブジェクトの実装を提供していないか、Migration Toolkit がその移行を処理していません。

エラーコード	メッセージと解決策
MTK-13000	このクラスに関連し、特定のエラーコードバインディングを持たないすべてのエラーは、このエラーコードを使用します。
MTK-13001	Oracle がインデックスをスキップすると、同じ列リストに対する複数のインデックスは許可されません。
MTK-13002	For <i>\$DATABASE</i> views migration is not supported.
MTK-13003	For <i>\$DATABASE</i> roles migration is not supported.
MTK-13004	MySQL、SQL Server、Sybase データベースのトリガー、シーケンス、プロシージャ、関数、パッケージ、シノニムは移行できません。
MTK-13005	SQL Server データベースのシーケンスとパッケージを移行することはできません。
MTK-13006	トリガーが移行されない場合、トリガーは別のユーザースキーマによって所有され、スキーマ間トリガーは EnterpriseDB でサポートされません。
MTK-13007	指定されたトリガーは移行されません。トリガーには、EnterpriseDB でサポートされていない WHEN 句があります。
MTK-13008	データベースリンク* <i>\$ DATABASE_LINK *</i> をスキップしています。EnterpriseDB は現在、このタイプのデータベースリンクをサポートしていません。
MTK-13009	警告！トリガー* <i>\$ TRIGGER *</i> の移行をスキップします。現在、テーブル以外のトリガーはターゲットデータベースではサポートされていません。
MTK-13010	プロシージャ、パッケージ、同義語、およびデータベースリンクを PostgreSQL データベースに移行することはできません。
MTK-13011	ドメインオブジェクトは、ターゲットデータベースではサポートされていません。
MTK-13012	Rules are not supported in <i>\$DATABASE</i> database.
MTK-13013	データベースの種類はサポートされていません。
MTK-13014	ハッシュパーティションは EnterpriseDB ではサポートされていません。
MTK-13015	<i>\$TYPE</i> is Not Supported by COPY.
MTK-13016	PostgreSQL への移行は、Oracle または PostgreSQL がソースデータベースである場合にのみサポートされます。
MTK-13017	グループは* <i>\$ DATABASE *</i> データベースではサポートされていません。
MTK-13018	Profile migration is not supported in <i>\$SRC_DB</i> to <i>\$TARGET_DB</i> permutation.
MTK-13019	不明なプロファイル* <i>\$ PROFILE *</i> を移行できません。
MTK-13020	プロファイルをデータベースバージョン* <i>\$ VERSION *</i> に移行できません。
MTK-13021	パスワードプロファイル確認関数* <i>\$ MY_VERIFICATION_FUNCTION *</i> は明示的に移行する必要があります。
MTK-13022	Advanced Queues migration is not supported from <i>\$SRC_DB</i> database to <i>\$TARGET_DB</i> database.
MTK-13023	アドバンストキューは、データベースバージョン* <i>\$ VERSION *</i> に移行できません。
MTK-13024	INTERVAL パーティションは* <i>\$ DATABASE *</i> ではサポートされていません。テーブルは INTERVAL 定義なしで移行されます。
MTK-13025	警告！ユーザープロファイルの移行は、ターゲットデータベースバージョン* <i>\$ VERSION</i> ではサポートされていません。ユーザー “ <i>\$ USER</i> ” のプロファイル “ <i>\$ PROFILE *</i> ” はスキップされます。
MTK-13026	Object Type migration is not supported from <i>\$SRC_DB</i> to <i>\$TARGET_DB</i> .

10.1.6 クラス 14-欠落オブジェクト

このクラスは、ソースデータベース内のメタデータ情報の検索の失敗を表します。

エラーコード	メッセージと解決策
MTK-14000	このクラスに関連し、特定のエラーコードバインディングを持たないすべてのエラーは、このエラーコードを使用します。
MTK-14001	ソース* \$ DATABASE *データベースに 1 つ以上のテーブルがありません。
MTK-14002	ソース* \$ DATABASE *データベースに 1 つ以上のテーブルが見つかりませんでした。 -tables モードでは、大文字と小文字を区別しない限り、テーブル名は大文字にする必要があります。
MTK-14003	ソース* \$ DATABASE *データベースに 1 人以上のユーザーが見つかりませんでした。 -users モードでは、大文字と小文字を区別しない限り、ユーザー名は大文字にする必要があります。

10.1.7 クラス 15-スキーマの移行

このクラスは、テーブル、制約、インデックス、同義語、ビュー、ユーザー、ロールなどの非手続き型データベースオブジェクトに関連する移行の問題を表します。

エラーコード	メッセージと解決策
MTK-15000	このクラスに関連し、特定のエラーコードバインディングを持たないすべてのエラーは、このエラーコードを使用します。
MTK-15001	制約* \$ CONSTRAINT *の作成中にエラーが発生しました。
MTK-15002	ロール* \$ ROLE *の作成中にエラーが発生しました。
MTK-15003	Error granting given privilege \$PRIVILEGE on \$OBJECT to \$USER.
MTK-15004	Error granting \$ROLE to \$USER.
MTK-15005	特権* \$ REASON *の付与エラー。
MTK-15006	ユーザー* \$ USER *の作成中にエラーが発生しました。
MTK-15007	インデックス* \$ INDEX *の作成中にエラーが発生しました。
MTK-15008	ビュー* \$ VIEW *の作成中にエラーが発生しました。
MTK-15009	マテリアライズドビュー* \$ MVIEW *の作成中にエラーが発生しました。
MTK-15010	パブリックシノニム* \$ SYNONYM *の作成中にエラーが発生しました。
MTK-15011	プライベートシノニム* \$ SYNONYM *の作成中にエラーが発生しました。
MTK-15012	シーケンス* \$ SEQUENCE *の作成中にエラーが発生しました。
MTK-15013	テーブル* \$ TABLE *の作成中にエラーが発生しました。
MTK-15014	データベースリンク* \$ DATABASE_LINK *の作成中にエラーが発生しました。
MTK-15015	指定されたオブジェクトタイプ* \$ OBJECT_TYPE *の作成中にエラーが発生しました。
MTK-15016	テーブル* \$ TABLE を \$ DATABASE *データベースに作成できませんでした。
MTK-15017	リンクされたスキーマ* \$ LINKED_SCHEMA *はターゲットデータベースに存在しません。スキーマを作成してから再試行してください。
MTK-15018	ドメイン* \$ DOMAIN *の作成中にエラーが発生しました。
MTK-15019	カスタムデータ型* \$ DATA_TYPE *の作成中にエラーが発生しました。
MTK-15020	グループ* \$ GROUP *のメンバーシップの作成中にエラーが発生しました。
MTK-15021	テーブル名* \$ TABLE *にはスキーマ修飾子がありません。複数のスキーマ移行コンテキストでは、各テーブルをスキーマで修飾する必要があります。
MTK-15022	スキーマ修飾子* \$ SCHEMA *はスキーマリストと一致しません。
MTK-15023	テーブルのメタデータ情報は利用できません。
MTK-15024	テーブルリストはまだ初期化されていません。
MTK-15025	データベースメタデータ情報の取得中にエラーが発生しました。
MTK-15026	グループ* \$ GROUP *の作成中にエラーが発生しました。
MTK-15027	プロファイル* \$ PROFILE *の作成中にエラーが発生しました。

10.1.8 クラス 16-手続き言語の移行

このクラスは、プロシージャ、関数、パッケージ、匿名ブロック、トリガー、ルールなどの手続き言語に基づくデータベースオブジェクトに関連する移行の問題を表します。

エラーコード	メッセージと解決策
MTK-16000	このクラスに関連し、特定のエラーコードバインディングを持たないすべてのエラーは、このエラーコードを使用します。
MTK-16001	トリガー* \$ TRIGGER *の作成中にエラーが発生しました。
MTK-16002	ルール* \$ RULE *の作成エラー。
MTK-16003	パッケージ仕様* \$ PACKAGE *の作成中にエラーが発生しました。
MTK-16004	パッケージ本体* \$ PACKAGE *の作成中にエラーが発生しました。
MTK-16005	パッケージボディが無効です。スキップしています...**注:** このエラーメッセージは、パッケージ仕様が正常に移行されたときにも表示されますが、ソースデータベースに対応するパッケージボディがありません。この場合、エラーメッセージが表示されても、ターゲットデータベースでパッケージ仕様が適切に機能する必要があります。
MTK-16006	プロシージャ* \$ PROCEDURE *の作成中にエラーが発生しました。
MTK-16007	関数* \$ FUNCTION *の作成中にエラーが発生しました。
MTK-16008	匿名ブロック* \$ BLOCK *の作成エラー。
MTK-16009	キュー* \$ QUEUE *の作成中にエラーが発生しました。

10.1.9 クラス 17-データの読み込み

このクラスは、ソースからターゲットデータベースにデータをコピーするときに発生する可能性のあるエラーを表します。

エラーコード	メッセージと解決策
MTK-17000	このクラスに関連し、特定のエラーコードバインディングを持たないすべてのエラーは、このエラーコードを使用します。
MTK-17001	テーブルへのデータ読み込みエラー:* \$ TABLE *
MTK-17002	エンコード変換では、代わりに一括挿入を使用して読み込まれるいくつかの文字が検出されました。
MTK-17003	コピーテーブル* \$ REASON *のエラー。
MTK-17004	無効なデータタイプ。
MTK-17005	このテーブルには、一括挿入ロード用にマークされた CLOB データが含まれます。

第 11 章

サポートされていない機能

Advanced Server は、一部の Oracle 機能を完全にサポートし、その他の機能を部分的にサポートします。Migration Toolkit は、サポートされていない機能を使用するオブジェクトを移行できません。

場合によっては、Migration Toolkit は部分的な互換性を提供する機能を使用するオブジェクトを移行できます。それ以外の場合、Advanced Server は適切な回避策をサポートします。

全文検索は、Oracle との完全な互換性がない機能の例です。Advanced Server データベースには、かなり以前から全文検索のサポートが含まれていましたが、実装は Oracle とはまったく異なります。Migration Toolkit は、この機能を利用するオブジェクトを移行できません。

Advanced Server がまだサポートしていない機能もあります。このカテゴリの機能には、自動ストレージ管理、テーブル圧縮、外部テーブルが含まれます。多くの場合、成功する回避策を調整できます。

- 自動ストレージ管理は、(システム固有の) ボリューム管理ソフトウェアに置き換えることができます。
- テーブル圧縮は、圧縮されたファイルシステムにあるテーブルスペースにデータを保存することで実装できます。
- Advanced Server には外部テーブルはありませんが、フラットテキストファイルをデータベースのステージングテーブルに*読み込む*ことができます。EDB * Loader ユーティリティを使用して、データを Advanced Server データベースにすばやくロードすることをお勧めします。
- Oracle データベースから Advanced サーバーに複数のプロファイルを移行する場合、移行の完了時にユーザー (または複数のユーザー) にプロファイルを手動で割り当てる必要があります。

11.1 サポートされていない Postgres 機能

Migration Toolkit は、次の Postgres 機能の移行をサポートしていません。

`OPERATOR CLASS`

`OPERATOR FAMILY`

`OPERATOR CLASS` および `OPERATOR FAMILY` については、次の場所にある PostgreSQL コアドキュメントを参照してください。

<https://www.postgresql.org/docs/11/static/indexes-opclass.html>

11.2 よくある質問

Migration Toolkit はパッケージの移行をサポートしていますか？

Migration Toolkit は、Oracle データベースから Advanced Server へのパッケージの移行をサポートしています。機能概要セクションを参照してください Advanced Server が提供する移行サポートに関する情報。

ソースデータベースからのデータのみを転送する方法はありますか？

はい。データ転送は、オンラインまたはオフラインの移行の一部としてサポートされています。

Migration Toolkit は、**CLOB** データ型のデータを含むテーブルの移行をサポートしていますか？

Migration Toolkit は、CLOB タイプのデータを含むテーブルの移行をサポートします。

Advanced Server は **enum** データ型をサポートしていますか？

Advanced Server は現在 enum データ型をサポートしていませんが、将来のリリースでそれらをサポートします。それまでは、チェック制約を使用して、Advanced Server データベースに追加されるデータを制限できます。チェック制約は、列が取ることができる有効な値のリストを定義します。

次のコードサンプルには、列の値を 3 つの dept タイプのいずれかに制限するチェック制約の簡単な例が含まれています。sales、admin または technical

```
CREATE TABLE emp (
emp_id INT NOT NULL PRIMARY KEY,
dept VARCHAR(255) NOT NULL,
CHECK (dept IN ('sales', 'admin', 'technical'))
);
```

有効な dept タイプを入力してチェック制約をテストすると、INSERT ステートメントはエラーなしで機能します。

```
test=# INSERT INTO emp VALUES (7324, 'sales');

INSERT 0 1
```

制約（サポート）に含まれていない値を挿入しようとすると、Advanced Server はエラーをスローします。

```
test=# INSERT INTO emp VALUES (7325, 'support');

ERROR:new row for relation /"emp/" violates check constraint /
"emp_dept_check/"
```

Advanced Server はマテリアライズドビューをサポートしていますか？

Postgres は、Oracle データベースと互換性のあるマテリアライズドビューをサポートしていません。Postgres でマテリアライズドビュー/サマリーテーブルをセットアップするには、サマリーテーブルを維持するトリガーを手動で作成する必要があります。現在、自動クエリ書き換えはサポートされていません。サマリーテーブルの存在をアプリケーションに認識させる必要があります。

TIME データ型を含む **MySQL** データベースから移行しようとする、次のエラーが表示されます。データをテーブルにロード中にエラーが発生しました:時間の形式が正しくありません。 **Postgres** は **MySQL** の “**TIME**” データ型をサポートしていますか？

時間コンポーネントの値が 24 以下である限り、**Postgres** は **TIME** データ型を保存するのに問題はありません。

Postgres とは異なり、**MySQL** の **TIME** データ型を使用すると、**TIME** または **INTERVAL** の値を表す値を格納できます。**INTERVAL** 値を表す **MySQL** の **TIME** 列に格納された値は、有効な **Postgres** の **TIMESTAMP** 値の許容範囲外になる可能性があります。移行プロセス中に、**Postgres** が **TIME** データ列に格納された値を検出し、その値が範囲外であると認識された場合、エラーが返されます。

索引

Configuring IDENT Authentication, 16
Connection Errors, 67

Defining a MySQL URL, 35
Defining a PostgreSQL URL, 31
Defining a SQL Server URL, 37
Defining a Sybase URL, 36
Defining an Advanced Server URL, 29
Defining an Oracle URL, 33

Executing Offline Migration Scripts, 47

FAQ, 87

Import Options, 48
Installing Migration Toolkit with Stack Builder
Plus, 18
Installing Source-Specific Drivers, 25

Migration Errors, 70
Migration Options, 53
Migration Toolkit™, 1
Migration Toolkit コマンドオプション, 45

Offline Migration Options, 46
Oracle Specific Options, 57

Schema Creation Options, 49
Schema Object Selection Options, 50

toolkit.properties ファイルの作成, 27

Unsupported PostgreSQL Features, 86

エラーコード, 75

サポートされていない機能, 85
サポートされているバージョン, 5

移行エラー, 67
移行ツールキットのインストール, 15
移行ツールキットの呼び出し, 39
移行方法, 7
機能概要, 11
新機能, 3