



EDB

EDB Migration Toolkit

リリース **55.6.0**

EnterpriseDB Corporation

2023 年 05 月 26 日

目次

第 1 章	Release notes	3
1.1	Version 55.6.0	3
1.2	Version 55.5.0	4
1.3	Version 55.4.0	4
1.4	Version 55.3.0	5
1.5	Version 55.2.3	6
1.6	Version 55.2.2	6
1.7	Version 55.2.1	7
1.8	Version 55.2.0	7
1.9	Version 55.1.0	8
1.10	Version 55.0.0	8
第 2 章	Supported platforms and databases	11
2.1	データベースバージョン	11
2.2	サポートされているオペレーティングシステムのバージョン	12
第 3 章	Overview	13
3.1	オブジェクト移行のサポート	14
3.2	オンライン移行とオフライン移行	14
3.3	制限事項	15
3.3.1	サポートされていない Postgres 機能	15
第 4 章	Installing Migration Toolkit	17
4.1	Linux x86-64 (amd64)	17
4.1.1	Red Hat Enterprise Linux (RHEL) およびその派生物	17
4.1.2	SUSE Linux Enterprise (SLES)	17
4.1.3	Debian とその派生物	17
4.2	Linux Installing Migration Toolkit on Linux IBM Power (ppc64le)	17
4.2.1	Red Hat Enterprise Linux (RHEL) およびその派生物	17
4.2.2	SUSE Linux Enterprise (SLES)	18
4.3	Macintosh	18
4.4	Windows	18
4.4.1	Installing Migration Toolkit on Linux x86 (amd64)	18
4.4.2	Installing Migration Toolkit on Linux IBM Power (ppc64le)	27
4.4.3	Installing on Mac OS X	29
4.4.4	Installing on Windows	31
4.4.5	Installing a JDBC driver	33
第 5 章	Upgrading	35
第 6 章	Specifying connection properties	37

6.1	Advanced Server の URL の定義	38
6.2	PostgreSQL URL の定義	39
6.3	BigAnimal URL の定義	40
6.4	Oracle URL の定義	41
6.5	SQL Server の URL の定義	42
6.5.1	Microsoft JDBC URL	42
6.5.2	JTDS の URL	43
6.6	MySQL URL の定義	43
6.7	Sybase URL の定義	44
第 7 章	Invoking Migration Toolkit	47
7.1	バイナリゼロ (NULL 文字) が埋め込まれた文字データのインポート	47
7.2	Oracle からスキーマを移行する	48
7.3	Oracle 以外のソースデータベースからの移行	49
7.3.1	Migration Toolkit command options	50
第 8 章	Migration errors	73
8.1	接続エラー	73
8.1.1	無効なユーザー名/パスワード	73
8.1.2	接続が拒否されました: FATAL: パスワード	74
8.1.3	例外: ORA-28000: アカウントはロックされています	74
8.1.4	例外: oracle.jdbc.driver.OracleDriver	74
8.1.5	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	75
8.1.6	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	75
8.2	移行エラー	75
8.2.1	エラー: 予想される最後の列の後に余分なデータ	75
8.2.2	テーブルにデータをロード中にエラーが発生しました: <i>TABLE_NAME</i>	76
8.2.3	制約 CONS_NAME_FK の作成中にエラーが発生しました	76
8.2.4	テーブルにデータをロード中にエラーが発生しました	77
8.2.5	エラー: タイプに対して値が長すぎます	78
8.2.6	ERROR: スレッドの例外: OutOfMemoryError	78
第 9 章	Error codes	81
9.1	エラーコードの概要	81
9.1.1	クラス 02 - 警告	82
9.1.2	クラス 10 - 無効なユーザー入力	82
9.1.3	クラス 11-構成の問題	84
9.1.4	クラス 12 - 不十分な特権	85
9.1.5	クラス 13 - サポートされていない機能	85
9.1.6	クラス 14 - 見つからないオブジェクト	87
9.1.7	クラス 15-スキーマの移行	87
9.1.8	クラス 16 - 手続き型言語の移行	88
9.1.9	クラス 17-データの読み込み	88
第 10 章	FAQ	91

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を EDB Postgres Advanced Server または PostgreSQL データベースに移行するのに役立ちます。

- Oracle
- MySQL
- Microsoft SQL Server

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

EDB Postgres Advanced Server インストーラーまたは Stack Builder を使用して、Migration Toolkit をインストールできます。Stack Builder は、EDB Postgres Advanced Server と [EnterpriseDB website](#) から入手可能な PostgreSQL ワンクリックインストーラーの両方とともに配布されます。

注釈: このドキュメントの Postgres という用語は、EDB Postgres Advanced Server または PostgreSQL のインストールを指します。スタックビルダーという用語は、StackBuilder Plus (EDB Postgres Advanced Server とともに配布) または Stack Builder (EnterpriseDB から 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。

`whats_new` `supported_operating_systems_and_database_versions` `migration_methodology` `features_overview` `installation_mtk` `building_toolkit.properties_file` `invoking_mtk` `mtk_command_options` `mtk_errors` `error_codes` `unsupported_features` `結論`

第1章 Release notes

Migration Toolkit のドキュメントには、マイナーリリースとパッチを含む Migration Toolkit 55 の最新バージョンが記載されています。このセクションのリリースノートには、各リリースの新機能に関する情報が記載されています。マイナーリリースまたはパッチリリースで導入された新機能の場合、コンテンツ内には、どのリリースで機能が導入されたかに関するインジケータもあります。

Version	Release Date
<i>Version 55.6.0</i>	2023 May 25
<i>Version 55.5.0</i>	2023 Feb 14
<i>Version 55.4.0</i>	2022 Nov 29
<i>Version 55.3.0</i>	2022 Oct 06
<i>Version 55.2.3</i>	2022 Jun 16
<i>Version 55.2.2</i>	2022 Mar 10
<i>Version 55.2.1</i>	2022 Jan 13
<i>Version 55.2.0</i>	2021 Dec 2
<i>Version 55.1.0</i>	2021 Sep 20
<i>Version 55.0.0</i>	2021 Mar 19

1.1 Version 55.6.0

Migration Toolkit 55.6 の新機能、機能強化、バグ修正、およびその他の変更には次のものが含まれます。

Type	Description
Enhancement	Migration Toolkit は、移行中にオブジェクトのサブセットを除外できる 2 つの新しいコマンドオプションを提供するようになりました。`-excludeTables` オプションは、すべてのテーブルが移行用に選択されたときにテーブルのサブセットをスキップし、`-excludeViews` オプションは、すべてのビューが移行用に選択されたときにビューのサブセットをスキップします。
Bug fix	Migration Toolkit がシステム生成インデックスを削除しようとする、データ移行が失敗する問題を修正しました。システムによって生成されたインデックスは、削除からスキップされるようになりました。【サポートチケット：# 90282】
Bug fix	`XMLTYPE` 列を持つ Oracle テーブルを移行するときに、「無効な LOB ロケータが指定されました」エラーでデータ移行が失敗する問題を修正しました。【サポートチケット：# 92284】

次のページに続く

表 2 – 前のページからの続き

Type	Description
Bug fix	ソーステーブルにLOB タイプの列が含まれている場合、SQL Server からのデータのみの移行を実行すると、 <i>TRUNCATE</i> が常にターゲットデータベースに適用される問題を修正しました。この動作は修正され、ユーザーが <code>-truncLoad</code> オプションを指定した場合にのみ <code>TRUNCATE</code> が適用されるようになりました。
Bug fix	Debian の Migration Toolkit パッケージを更新し、JDBC ドライバーが Migration Toolkit にバンドルされなくなり、Migration Migration Toolkit でサポートされる他の OS プラットフォームのパッケージと一貫性を保ちました。

1.2 Version 55.5.0

Migration Toolkit 55.5 の新機能、機能強化、バグ修正、およびその他の変更には以下が含まれます。

Type	Description
Enhancement	Migration Toolkit は、さまざまな移行順列のソースおよび/またはターゲットとして EDB Postgres Advanced Server バージョン 15 をサポートするようになりました。
Enhancement	Migration Toolkit が拡張され、ソースデータベースとターゲットデータベースの列の順序が異なる場合に、デフォルトの <code>COPY</code> を使用したデータの移行をサポートするようになりました。【サポートチケット #88336】
Bug fix	ソース PostgreSQL / EDB Postgres Advanced Server データベースのユニークインデックス制約に基づく外部キー制約が、ターゲットデータベースへの移行からスキップされる問題を修正しました。【サポートチケット #89491】
Bug fix	パラレルデータロードモードで 1 つ以上のテーブルの移行が停止する問題を修正しました。

1.3 Version 55.4.0

Migration Toolkit 55.4 の新機能、機能強化、バグ修正、およびその他の変更には次が含まれます。

Type	Description
Enhancement	Migration Toolkit は、PostgreSQL バージョン 15 で動作することが認定されました。PostgreSQL 15 は、さまざまな移行の順列のソースおよび/またはターゲットとしてサポートされるようになりました。
Bug fix	複数の MySQL ソーステーブルが同じ制約名を共有している場合、テーブル制約エラーの原因となる問題を修正しました。【サポートチケット #86413】
Bug fix	バックスラッシュで終わるテキスト列値を持つ MySQL テーブル行のデータ移行が失敗する問題を修正しました。【サポートチケット #86412】

次のページに続く

表 4 – 前のページからの続き

Type	Description
Other	Migration Toolkit で使用するための PostgreSQL 10 および EDB Postgres Advanced Server (EPAS) 10 のサポートを削除しました。PostgreSQL および EPAS のこれらのバージョンは廃止され、通常のサポート契約ではサポートされなくなりました。
Other	Migration Toolkit で使用する Oracle 10g のサポートが削除されました。このバージョンは、以前に Migration Toolkit での使用についてテストおよび認定されています。Oracle 10g は Migration Toolkit と連携し続ける可能性があります、EDB は継続的な検証を実行しなくなりました。

1.4 Version 55.3.0

Migration Toolkit 55.3 の新機能、機能強化、バグ修正、およびその他の変更には、次のものが含まれます。

Type	Description
Enhancement	Migration Toolkit は、Microsoft が提供する SQL Server JDBC ドライバーバージョン 10.2 で動作することが認定されました。Migration Toolkit で Microsoft SQL Server への接続にはオープンソースの jTDS ドライバーを引き続き使用できますが、Microsoft が提供する SQL Server JDBC ドライバーを使用することをお勧めします SQL Server のより新しいバージョンのサポートを提供し SQL Server。
Enhancement	Migration Toolkit は、Debian 11 プラットフォームで動作することが認定されました。
Security Fix	Migration Toolkit は、 CVE-2022-31197 のセキュリティ修正が含まれる最新の EDB JDBC ドライバーバージョン 42.5.0.1 および PostgreSQL コミュニティ JDBC ドライバーバージョン 42.5.0 で認定されています。
Security Fix	Migration Toolkit は、 CVE-2022-21363 のセキュリティ修正が含まれる MySQL JDBC ドライバーバージョン 8.0.30 で動作することが保証されています。Migration Toolkit ドキュメントの MySQL JDBC ドライバーへの参照は、対応して更新されています。
Bug Fix	仮想列が含まれている場合、EPAS への Oracle テーブルの移行が失敗する問題を修正しました。[サポートチケット # 82718]
Other	Migration Toolkit で使用する Microsoft SQL Server 2008 および SQL Server Server 2012 のサポートは、これらのバージョンが Microsoft によってサポートされなくなったため、削除されました。これらのバージョンは、以前に Migration Toolkit での使用についてテストおよび認定されています。彼らは Migration Toolkit で作業し続けるかもしれませんが、EDB はもはやそれらの継続的な検証を実行していません。

1.5 Version 55.2.3

Migration Toolkit 55.2.3 の新機能、機能強化、バグ修正、およびその他の変更には、次のものが含まれます。

Type	Description
Bug Fix	列定義にデフォルト句として NEXTVAL 呼び出しを使用したシーケンスが含まれる場合、EDB Postgres Advanced Server への Oracle テーブルの移行が失敗する問題を修正しました。
Bug Fix	POSITION キーワードにちなんで名前付けられた列を持つテーブルの移行が失敗する問題を修正しました。
Bug Fix	Migration Toolkit Windows インストーラーを修正し、アップグレードプロセスの一環として古いバージョンの commons-lang ライブラリを削除するようにしました。
Bug Fix	スキーマ修飾テーブル名を報告するように、情報/エラーメッセージの一部を修正しました。

1.6 Version 55.2.2

Migration Toolkit 55.2.2 の新機能、機能強化、バグ修正、およびその他の変更には、次のものが含まれます。

Type	Description
Enhancement	SQL Server 2016、2017、および 2019 と Sybase (SAP ASE) 16.0 のサポートを提供します。
Security Fix	Windows および macOS で使用されるインストーラーが更新され、 CVE-2022-21724 をアドレス指定する EDB JDBC Connector の最新バージョンが含まれています。Linux の場合、JDBC ドライバーは Migration Toolkit にパッケージされていないため、この EDB knowledge base article に記載されている Linux の手順を使用して、ドライバを個別に更新する必要があります。
Bug Fix	Migration Toolkit は、テーブルデータのロード中に ENABLE TRIGGER が失敗した場合、例外を適切に処理するようになりました。特定のテーブルで ENABLE TRIGGER が失敗した場合、残りのテーブルではスキップされません。
Bug Fix	Migration Toolkit は、MySQL データベースからの移行中に TEXT フィールドに入力されるバックスラッシュをエスケープするようになりました。

1.7 Version 55.2.1

Migration Toolkit 55.2.1 の新機能、機能強化、バグ修正、およびその他の変更には、次のものが含まれます。

Type	Description
Security Fix	Migration Toolkit のロギングは、log4j 2.17.1 に基づいています。以前は、log4j v1 に基づいていました。この変更により、log4j v1 に対して報告された脆弱性が削除され、log4j v2 の古いバージョンに対して発見および報告された RCE 脆弱性がバイパスされます。
Bug Fix	カスタム列型のマッピングは、Oracle RAW 型から PostgreSQL / Postgres Plus Advanced Server UUID 型に固定されています。
Bug Fix	<code>-loaderCount</code> パラメーターは、Sybase、MySQL、および SQL Server データベースでは許可されていません。 該当なし
Bug Fix	MySQL から PostgreSQL への移行中のインデックス名の重複の問題を修正しました。
Bug Fix	結果セットを返さない MySQL で実行されるクエリの <code>executeQuery</code> を <code>executeUpdate</code> に置き換えます。

1.8 Version 55.2.0

Migration Toolkit 55.2.0 の新機能、機能強化、バグ修正、およびその他の変更には、次のものが含まれます。

Type	Description
Enhancement	Migration Toolkit は、PostgreSQL 14 のサポートが認定されています。
Enhancement	新しい Oracle 互換性機能の一部を含む EPAS 14 のサポートが追加されました。
Enhancement	LIST BY PARTITION 句の後に AUTOMATIC キーワードを指定したパーティションテーブルを、ターゲットの EPAS 13 および 14 データベースに移行できるようになりました。
Enhancement	Migration Toolkit は、SUBPARTITION TEMPLATE 句を使用してパーティションテーブルをターゲット EPAS 14 データベースに正常に移行するようになりました。
Bug Fix	ORA-01795 問題を修正して、1000 人を超えるユーザーを正常に移行できるようにしました。
Bug Fix	ORA-01795 問題を修正して、1000 を超えるパーティションを正常に移行できるようにしました。

1.9 Version 55.1.0

Migration Toolkit 55.1.0 の新機能、機能強化、バグ修正、およびその他の変更には、次のものが含まれます。

Type	Description
Enhancement	Migration Toolkit が拡張され、MySQL 5.7 および 8.0 からの移行をサポートするようになりました。
Bug Fix	MySQL から PostgreSQL への移行、ブールデータ型が LO モードで移行されませんでした。この問題は修正されました。
Bug Fix	数字で始まるテーブル名を MySQL から PostgreSQL に移行できるようにする問題を修正しました。
Bug Fix	MySQL から PostgreSQL への移行時に TIMESTAMP 値が同じタイムゾーン内で変化する問題を修正しました。
Bug Fix	MySQL から PostgreSQL への移行中に、列挙型は fastcopy モードのターゲットデータベースで作成されませんでした。この問題は修正されました。
Bug Fix	n > 1 の場合の MySQL から PostgreSQL への BIT(n) データ型の移行に関連するエラーを修正しました。
Bug Fix	Oracle から PostgreSQL への移行中に、数値形式の例外が報告される問題を修正しました (72863)。
Bug Fix	関連付けられた OID 値が INTEGER の範囲を超えると、PostgreSQL テーブルの移行が失敗します。この問題は修正されました (72694)。
Bug Fix	PostgreSQL から Oracle への JSON データ型の移行に関連する問題を修正しました。
Bug Fix	Oracle 19c から EDB Postgres Advanced Server へ、特定のシステムユーザー/スキーマが誤って移行対象として選択されました。この問題は修正されました。
Bug Fix	問題を修正して、パラレルロードモードで大文字と小文字を区別するスキーマ/テーブル名を Oracle から EDB Postgres Advanced Server に移行できるようにしました。
Bug Fix	カスタム列タイプマッピングオプションの使用に関連していた特定のコーナーケースが修正されました。

1.10 Version 55.0.0

Migration Toolkit 55.0.0 の新機能、機能強化、バグ修正、およびその他の変更には、次のものが含まれます。

Type	Description
Enhancement	テーブルレベルでの並列データロードのサポート。これにより、マルチプルのスレッドを使用して、ソースデータベーステーブルからデータを並列にロードし、ターゲットデータベーステーブルに並列に適用できます。これにより、比較的大きなサイズのテーブルを処理するときのデータのロード時間が大幅に改善されます。ラボのベンチマーク結果では、マルチプルの並列スレッドを使用すると、大きなテーブルのデータロード時間が 25~60 %削減されることが明らかになりました。この機能は、ソース PostgreSQL、EDB Postgres Advanced Server、および Oracle データベースでサポートされています。 並列データロードの移行オプション を参照してください。
Enhancement	テーブルレベルの平行データロードの機能強化の一環として、 <code>tableLoaderLimit</code> および <code>parallelLoadRowLimit</code> オプションがコマンドラインオプションリストに追加されました。
Enhancement	Oracle バージョン 18c および 19c からのスキーマとデータの移行のサポート。
Enhancement	スキーマとデータ移行セッションの合計移行時間は、 <code>Total Elapsed Migration Time</code> 属性を介して報告されます。
Bug Fix	サマリーレポートは、テーブルのデータ移行が失敗した場合、ステータスを正確に報告しません。
Bug Fix	概要レポートは、冗長な無効なオブジェクトを削除しません。
Bug Fix	指数形式で報告されるテーブルのデータサイズ。
Bug Fix	'-tables' list を介して重複するテーブル名が指定された場合のエラーを回避します。
Bug Fix	以前は 0.0 MB と表示されていた小さなテーブルサイズを反映します。

第2章 Supported platforms and databases

2.1 データベースバージョン

Migration Toolkit では、次のデータベース製品バージョンを使用できます。

- PostgreSQL バージョン 11、12、13、14、および 15
- EDB Postgres Advanced Server バージョン 11、12、13、14、および 15
- Oracle 11g
- Oracle 12c
- オラクル 18c
- オラクル 19c
- SQL Server 2014
- SQL Server 2016
- SQL Server 2017
- SQL Server 2019
- MySQL 5.5
- MySQL 5.7
- MySQL 8.0
- (Sybase) SAP Adaptive Server Enterprise 15.7
- (Sybase) SAP Adaptive Server Enterprise 16.0

他のデータベース製品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。

注釈: Migration Toolkit は Oracle Real Application Clusters (RAC) および Exadata でテストされておらず、公式に使用をサポートしていません。ただし、Migration Toolkit は、単一の永続ノードに接続されている場合にも機能する場合があります。詳細については、EDB の担当者にお問い合わせください。

2.2 サポートされているオペレーティングシステムのバージョン

Migration Toolkit は、Linux、Windows、および MacOS プラットフォームでのインストールをサポートしています。詳細は [Product Compatibility](#) を参照してください。

注釈: ojdbc7.jar は、ResultSet からフェッチする際の Integer.MAX_VALUE (2147483647) を超えるレコード数の制限が原因で、大規模なデータの移行が不完全になる場合があります。この問題を回避するには、ojdbc8.jar にアップグレードすることをお勧めします。

第3章 Overview

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 を起動し、移行オプションを指定します。

Migration Toolkit は、データベースオブジェクトとデータを EDB Postgres Advanced Server または PostgreSQL データベースに移行するのに役立ちます。

- Oracle
- MySQL
- SQL Server

Migration Toolkit では、データベースオブジェクトとデータを Sybase から EDB Postgres 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、EDB Postgres Advanced Server と PostgreSQL の間で移行することもできます。Migration Toolkit には多く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。

- `-safeMode` オプションを使用して、移行時に各行をコミットします。
- `-fastCopy` オプションを使用して WAL ロギングをバイパスし、移行を最適化します。
- `-batchSize` オプションを使用して、一括挿入のバッチサイズを制御します。
- `-cpBatchSize` オプションを使用して、COPY コマンドで使用するバッチサイズを指定します。
- `-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。
- `-filterProp` オプションを使用して、指定した条件を満たす行のみを移行します。
- `-customColTypeMapping` オプションを使用して、選択した列のデータ型を変更します。
- `-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。
- EDB Postgres 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。
- EDB Postgres 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。

3.1 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 で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。

Object	Oracle	Sybase	SQL Server	MySQL
Schemas	X	X	X	X
Tables	X	X	X	X
List-partitioned table	X			
Range-partitioned table	X			
Hash partitioned table	X			
Constraints	X	X	X	X
Indexes	X	X	X	X
Triggers	X			
Table data	X	X	X	X
Views	X		X	
Materialized views	X			
Packages	X			
Procedures	X			
Functions	X			
Sequences	X			
Users/Roles	X			
Profiles	X			
Object types	X			
Object type methods	X			
Database links	X			
Queues	X			

インポートされるオブジェクトをきめ細かく制御でき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。

3.2 オンライン移行とオフライン移行

Migration Toolkit は、Postgres データベースにすぐに直接移行できます（**オンライン移行**）。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を選択することもできます（**オフライン移行**）。

デフォルトでは、Migration Toolkit はオブジェクトを Postgres データベースに直接作成します。または、`-offlineMigration` オプションを含めて SQL スクリプトを生成し、後で使用して移行したオブジェクトまたはデータを新しいデータベースに再現します。Migration Toolkit によって生成された移行スクリプトを実行する前に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。

3.3 制限事項

EDB Postgres Advanced Server は、一部の Oracle 機能を完全にサポートし、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。他の場合、EDB Postgres Advanced Server は適切な回避策をサポートします。

全文検索は、Oracle と完全な互換性がない機能の例です。EDB Postgres Advanced Server データベースには、かなり長い間全文検索のサポートが含まれていますが、実装は Oracle のものとは異なります。Migration Toolkit は、この機能を使用するオブジェクトを移行できません。

EDB Postgres Advanced Server はまだ他の機能をサポートしていません。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、成功する回避策を実装できます。

- Automated Storage Management 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。

- 圧縮ファイルシステム上にある表領域にデータを格納することにより、テーブル圧縮を実装できます。
- EDB Postgres Advanced Server に外部テーブルは存在しませんが、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ロードできます。EDB*Loader ユーティリティを使用して、データを EDB Postgres 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。
- 複数のプロファイルを Oracle データベースから EDB Postgres Advanced Server に移行する場合、移行の完了時にプロファイルをユーザーに手動で割り当てる必要があります。

3.3.1 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** の詳細については、

[PostgreSQL core documentation](#) を参照してください。

第4章 Installing Migration Toolkit

リンクを選択して、該当するインストール手順にアクセスします。

4.1 Linux x86-64 (amd64)

4.1.1 Red Hat Enterprise Linux (RHEL) およびその派生物

- *Installing Migration Toolkit on RHEL 8 ppc64le*、*Installing Migration Toolkit on RHEL 7 or OL 7 x86_64*
- Oracle Linux (OL) 8、Oracle Linux (OL) 7
- *Installing Migration Toolkit on AlmaLinux 8 or Rocky Linux 8 x86_64*
- *Installing Migration Toolkit on AlmaLinux 8 or Rocky Linux 8 x86_64*
- *Installing Migration Toolkit on CentOS 7 x86_64*

4.1.2 SUSE Linux Enterprise (SLES)

- *Installing Migration Toolkit on SLES 15 ppc64le*、*Installing Migration Toolkit on SLES 12 ppc64le*

4.1.3 Debian とその派生物

- *Installing Migration Toolkit on Ubuntu 22.04 x86_64*、*Installing Migration Toolkit on Ubuntu 20.04 x86_64*
、*Installing Migration Toolkit on Ubuntu 18.04 x86_64*
- *Installing Migration Toolkit on Debian 11 x86_64*、*Installing Migration Toolkit on Debian 10 x86_64*

4.2 Linux Installing Migration Toolkit on Linux IBM Power (ppc64le)

4.2.1 Red Hat Enterprise Linux (RHEL) およびその派生物

- *Installing Migration Toolkit on RHEL 8 ppc64le*

4.2.2 SUSE Linux Enterprise (SLES)

- *Installing Migration Toolkit on SLES 15 ppc64le*、*Installing Migration Toolkit on SLES 12 ppc64le*

4.3 Macintosh

- *Installing on Mac OS X*

4.4 Windows

- *Windows Server 2019*

4.4.1 Installing Migration Toolkit on Linux x86 (amd64)

リポジトリへのアクセスを含む、オペレーティングシステム固有のインストール手順については、以下を参照してください。

Red Hat Enterprise Linux (RHEL) およびその派生物

- *Installing Migration Toolkit on RHEL 8 ppc64le*
- *Installing Migration Toolkit on RHEL 7 or OL 7 x86_64*
- Oracle Linux (OL) 8
- Oracle Linux (OL) 7
- *Installing Migration Toolkit on AlmaLinux 8 or Rocky Linux 8 x86_64*
- *Installing Migration Toolkit on AlmaLinux 8 or Rocky Linux 8 x86_64*
- *Installing Migration Toolkit on CentOS 7 x86_64*

SUSE Linux Enterprise (SLES)

- *Installing Migration Toolkit on SLES 15 ppc64le*
- *Installing Migration Toolkit on SLES 12 ppc64le*

Debian とその派生物

- *Installing Migration Toolkit on Ubuntu 22.04 x86_64*
- *Installing Migration Toolkit on Ubuntu 20.04 x86_64*
- *Installing Migration Toolkit on Ubuntu 18.04 x86_64*
- *Installing Migration Toolkit on Debian 11 x86_64*
- *Installing Migration Toolkit on Debian 10 x86_64*

Installing Migration Toolkit on RHEL 8 or OL 8 x86_64

前提条件

インストールプロセスを開始する前に：

- リポジトリをセットアップする

リポジトリのセットアップは1回限りのタスクです。リポジトリをすでに設定している場合、この手順を実行する必要はありません。

リポジトリをセットアップするには、

EDB repositories に移動し、そこに記載されている指示に従います。

- EPEL リポジトリをインストールします

```
sudo dnf -y install https://dl.fedoraproject.org/pub/epel/epel-release-latest-8.
↪noarch.rpm
```

- 追加のリポジトリを有効にして依存関係を解決します

```
ARCH=$( /bin/arch ) subscription-manager repos --enable "codeready-builder-for-rhel-8-
↪${ARCH}-rpms"
```

パッケージをインストールする

```
sudo dnf -y install edb-migrationtoolkit
```

初期構成

Migration Toolkit を呼び出す前に、ソースおよびターゲットデータベースに接続するための 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。詳細は *Installing a JDBC driver* を参照してください。

Installing Migration Toolkit on AlmaLinux 8 or Rocky Linux 8 x86_64

前提条件

インストールプロセスを開始する前に：

- リポジトリをセットアップする

リポジトリのセットアップは 1 回限りのタスクです。リポジトリをすでに設定している場合、この手順を実行する必要はありません。

リポジトリをセットアップするには、

EDB repositories に移動し、そこに記載されている指示に従います。

- その他の前提条件に対処する

```
# Install the EPEL repository:
sudo dnf -y install https://dl.fedoraproject.org/pub/epel/epel-release-latest-8.
↳noarch.rpm

# Enable additional repositories to resolve dependencies:
ARCH=$( /bin/arch ) subscription-manager repos --enable "codeready-builder-for-rhel-8-
↳${ARCH}-rpms"
```

パッケージをインストールする

```
sudo dnf -y install edb-migrationtoolkit
```

初期構成

Migration Toolkit を呼び出す前に、ソースおよびターゲットデータベースに接続するための 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。詳細は *Installing a JDBC driver* を参照してください。

Installing Migration Toolkit on RHEL 7 or OL 7 x86_64

前提条件

インストールプロセスを開始する前に：

- リポジトリをセットアップする

リポジトリのセットアップは1回限りのタスクです。リポジトリをすでに設定している場合、この手順を実行する必要はありません。

リポジトリをセットアップするには、

EDB repositories に移動し、そこに記載されている指示に従います。

- その他の前提条件に対処する

```
# Install the EPEL repository:  
sudo yum -y install https://dl.fedoraproject.org/pub/epel/epel-release-latest-7.  
↳noarch.rpm
```

パッケージをインストールする

```
sudo yum -y install edb-migrationtoolkit
```

初期構成

Migration Toolkit を呼び出す前に、ソースおよびターゲットデータベースに接続するための 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。詳細は *Installing a JDBC driver* を参照してください。

Installing Migration Toolkit on CentOS 7 x86_64

前提条件

インストールプロセスを開始する前に：

- リポジトリをセットアップする

リポジトリのセットアップは1回限りのタスクです。リポジトリをすでに設定している場合、この手順を実行する必要はありません。

リポジトリをセットアップするには、

EDB repositories に移動し、そこに記載されている指示に従います。

- EPEL リポジトリをインストールします

```
sudo yum -y install https://dl.fedoraproject.org/pub/epel/epel-release-latest-7.  
↪noarch.rpm
```

パッケージをインストールする

```
sudo yum -y install edb-migrationtoolkit
```

初期構成

Migration Toolkit を呼び出す前に、ソースおよびターゲットデータベースに接続するための 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。詳細は *Installing a JDBC driver* を参照してください。

Installing Migration Toolkit on SLES 15 x86_64

前提条件

インストールプロセスを開始する前に：

- リポジトリをセットアップする

リポジトリのセットアップは 1 回限りのタスクです。リポジトリをすでに設定している場合、この手順を実行する必要はありません。

リポジトリをセットアップするには、

[EDB repositories](#) に移動し、そこに記載されている指示に従います。

- 必要な SUSE モジュールをアクティブにします

```
sudo SUSEConnect -p PackageHub/15.4/x86_64
```

- メタデータを更新する

```
sudo zypper refresh
```

パッケージをインストールする

```
sudo zypper -n install edb-migrationtoolkit
```

初期構成

Migration Toolkit を呼び出す前に、ソースおよびターゲットデータベースに接続するための 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。詳細は *Installing a JDBC driver* を参照してください。

Installing Migration Toolkit on SLES 12 x86_64

前提条件

インストールプロセスを開始する前に：

- リポジトリをセットアップする

リポジトリのセットアップは 1 回限りのタスクです。リポジトリをすでに設定している場合、この手順を実行する必要はありません。

リポジトリをセットアップするには、

[EDB repositories](#) に移動し、そこに記載されている指示に従います。

- 必要な SUSE モジュールをアクティブにします

```
sudo SUSEConnect -p PackageHub/12.5/x86_64
sudo SUSEConnect -p sle-sdk/12.5/x86_64
```

- メタデータを更新する

```
sudo zypper refresh
```

パッケージをインストールする

```
sudo zypper -n install edb-migrationtoolkit
```

初期構成

Migration Toolkit を呼び出す前に、ソースおよびターゲットデータベースに接続するための 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。詳細は *Installing a JDBC driver* を参照してください。

Installing Migration Toolkit on Ubuntu 22.04 x86_64

前提条件

インストールプロセスを開始する前に：

- リポジトリをセットアップする

リポジトリのセットアップは1回限りのタスクです。リポジトリをすでに設定している場合、この手順を実行する必要はありません。

リポジトリをセットアップするには、

EDB repositories に移動し、そこに記載されている指示に従います。

パッケージをインストールする

```
sudo apt-get -y install edb-migrationtoolkit
```

初期構成

Migration Toolkit を呼び出す前に、ソースおよびターゲットデータベースに接続するための 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。詳細は *Installing a JDBC driver* を参照してください。

Installing Migration Toolkit on Ubuntu 20.04 x86_64

前提条件

インストールプロセスを開始する前に：

- リポジトリをセットアップする

リポジトリのセットアップは1回限りのタスクです。リポジトリをすでに設定している場合、この手順を実行する必要はありません。

リポジトリをセットアップするには、

EDB repositories に移動し、そこに記載されている指示に従います。

パッケージをインストールする

```
sudo apt-get -y install edb-migrationtoolkit
```

初期構成

Migration Toolkit を呼び出す前に、ソースおよびターゲットデータベースに接続するための 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。詳細は [Installing a JDBC driver](#) を参照してください。

Installing Migration Toolkit on Ubuntu 18.04 x86_64

前提条件

インストールプロセスを開始する前に：

- リポジトリをセットアップする

リポジトリのセットアップは 1 回限りのタスクです。リポジトリをすでに設定している場合、この手順を実行する必要はありません。

リポジトリをセットアップするには、

[EDB repositories](#) に移動し、そこに記載されている指示に従います。

パッケージをインストールする

```
sudo apt-get -y install edb-migrationtoolkit
```

初期構成

Migration Toolkit を呼び出す前に、ソースおよびターゲットデータベースに接続するための 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。詳細は [Installing a JDBC driver](#) を参照してください。

Installing Migration Toolkit on Debian 11 x86_64

前提条件

インストールプロセスを開始する前に：

- リポジトリをセットアップする

リポジトリのセットアップは 1 回限りのタスクです。リポジトリをすでに設定している場合、この手順を実行する必要はありません。

リポジトリをセットアップするには、

[EDB repositories](#) に移動し、そこに記載されている指示に従います。

パッケージをインストールする

```
sudo apt-get -y install edb-migrationtoolkit
```

初期構成

Migration Toolkit を呼び出す前に、ソースおよびターゲットデータベースに接続するための 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。詳細は [Installing a JDBC driver](#) を参照してください。

Installing Migration Toolkit on Debian 10 x86_64

前提条件

インストールプロセスを開始する前に：

- リポジトリをセットアップする

リポジトリのセットアップは 1 回限りのタスクです。リポジトリをすでに設定している場合、この手順を実行する必要はありません。

リポジトリをセットアップするには、

[EDB repositories](#) に移動し、そこに記載されている指示に従います。

パッケージをインストールする

```
sudo apt-get -y install edb-migrationtoolkit
```

初期構成

Migration Toolkit を呼び出す前に、ソースおよびターゲットデータベースに接続するための 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。詳細は [Installing a JDBC driver](#) を参照してください。

4.4.2 Installing Migration Toolkit on Linux IBM Power (ppc64le)

リポジトリへのアクセスを含む、オペレーティングシステム固有のインストール手順については、以下を参照してください。

Red Hat Enterprise Linux (RHEL)

- *Installing Migration Toolkit on RHEL 8 ppc64le*

SUSE Linux Enterprise (SLES)

- *Installing Migration Toolkit on SLES 15 ppc64le*
- *Installing Migration Toolkit on SLES 12 ppc64le*

Installing Migration Toolkit on RHEL 8 ppc64le

前提条件

インストールプロセスを開始する前に：

- リポジトリをセットアップする

リポジトリのセットアップは1回限りのタスクです。リポジトリをすでに設定している場合、この手順を実行する必要はありません。

リポジトリをセットアップするには、

[EDB repositories](#) に移動し、そこに記載されている指示に従います。

- EPEL リポジトリをインストールします

```
sudo dnf -y install https://dl.fedoraproject.org/pub/epel/epel-release-latest-8.
↪noarch.rpm
```

- キャッシュを更新する

```
sudo dnf makecache
```

- 追加のリポジトリを有効にして依存関係を解決します

```
ARCH=$( /bin/arch ) subscription-manager repos --enable "codeready-builder-for-rhel-8-
↪${ARCH}-rpms"
```

パッケージをインストールする

```
sudo dnf -y install edb-migrationtoolkit
```

初期構成

Migration Toolkit を呼び出す前に、ソースおよびターゲットデータベースに接続するための 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。詳細は [Installing a JDBC driver](#) を参照してください。

Installing Migration Toolkit on SLES 15 ppc64le

前提条件

インストールプロセスを開始する前に：

- リポジトリをセットアップする

リポジトリのセットアップは 1 回限りのタスクです。リポジトリをすでに設定している場合、この手順を実行する必要はありません。

リポジトリをセットアップするには、

[EDB repositories](#) に移動し、そこに記載されている指示に従います。

- 必要な SUSE モジュールをアクティブにします

```
sudo SUSEConnect -p PackageHub/15.4/ppc64le
```

- メタデータを更新する

```
sudo zypper refresh
```

パッケージをインストールする

```
sudo zypper -n install edb-migrationtoolkit
```

初期構成

Migration Toolkit を呼び出す前に、ソースおよびターゲットデータベースに接続するための 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。詳細は [Installing a JDBC driver](#) を参照してください。

Installing Migration Toolkit on SLES 12 ppc64le

前提条件

インストールプロセスを開始する前に：

- リポジトリをセットアップする

リポジトリのセットアップは1回限りのタスクです。リポジトリをすでに設定している場合、この手順を実行する必要はありません。

リポジトリをセットアップするには、

[EDB repositories](#) に移動し、そこに記載されている指示に従います。

- 必要な SUSE モジュールをアクティブにします

```
sudo SUSEConnect -p PackageHub/12.5/ppc64le
sudo SUSEConnect -p sle-sdk/12.5/ppc64le
```

- メタデータを更新する

```
sudo zypper refresh
```

パッケージをインストールする

```
sudo zypper -n install edb-migrationtoolkit
```

初期構成

Migration Toolkit を呼び出す前に、ソースおよびターゲットデータベースに接続するための 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。詳細は [Installing a JDBC driver](#) を参照してください。

4.4.3 Installing on Mac OS X

EDB は、MacOS 用のグラフィカルな対話型インストーラーを提供します。次の2つの方法でアクセスできます。

- [Downloads page](#) からグラフィカルインストーラーをダウンロードし、インストーラーを直接起動します。[直接インストール](#) を参照してください。
- Stack Builder with PostgreSQL を使用して、EDB インストーラーパッケージをダウンロードし、グラフィカルインストーラーを呼び出します。

[スタックビルダーの使用](#) を参照してください。

前提条件

Stack Builder で Migration Toolkit のインストールを実行するには、Java JVM（バージョン 1.8.0 以降）を配置する必要があります。

[Adoptium](#) は、完全にオープンソースのビルド済み OpenJDK バイナリを提供します。

JAVA_HOME 環境変数も設定する必要があります。次のコマンドを使用します。

```
export JAVA_HOME=$(/usr/libexec/java_home)
```

スタックビルダーの使用

PostgreSQL を使用している場合は、Stack Builder でグラフィカルインストーラーを呼び出すことができます。

[スタックビルダーの使用](#) を参照してください。

1. スタックビルダーで、モジュール選択ページが表示されるまでプロンプトに従います。

[Welcome] ページで、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** を選択して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** をクリックして続行します。

1. **登録必須および試用製品** ノードを展開します。
2. **EnterpriseDBTools** ノードを展開し、**MigrationToolkit** を選択します。
3. **Next** を選択し、[グラフィカルインストーラーの使用](#) に進みます。

直接インストール

1. グラフィカルインストーラーをダウンロードした後、ターミナルを使用して次のコマンドを実行します。必要に応じて、プロンプトが表示されたらパスワードを入力します。スーパーユーザーまたは管理者権限が必要です。

```
sudo ~/Downloads/edb-migrationtoolkit-55.5.0-1-osx.app/Contents/MacOS/installbuilder.  
→ sh
```

1. [グラフィカルインストーラーの使用](#) に進みます。

グラフィカルインストーラーの使用

1. インストール言語を選択し、**OK** を選択します。

1. Setup Migration Toolkit ページで、**Next** を選択します。
2. 使用許諾契約を読みます。契約に同意する場合は、[**私は契約に同意します**] を選択し、[**次へ**] を選択します。
3. Migration Toolkit をインストールするディレクトリを参照するか、インストーラーがデフォルトの場所に Migration Toolkit をインストールできるようにします。**Next** を選択します。
4. [Ready to Install] ページで、[**Next**] を選択します。

情報ボックスには、選択したコンポーネントのインストールの進行状況が表示されます。

1. インストールが完了したら、**Finish** を選択します。

Migration Toolkit をインストールした後、移行を実行する前に、適切な JDBC ドライバーをインストールする必要があります。詳細については、[Installing a JDBC driver](#) を参照してください。

注釈: EDB JDBC ドライバーは Mac OS では利用できません。

4.4.4 Installing on Windows

EDB は、Windows のグラフィカルな対話型インストーラーを提供します。次の 2 つの方法でアクセスできます。

- [Downloads page](#) からグラフィカルインストーラーをダウンロードし、インストーラーを直接起動します。[直接インストール](#) を参照してください。
- Stack Builder（PostgreSQL を使用）または StackBuilder Plus（EDB Postgres Advanced Server を使用）を使用して、EDB インストーラーパッケージをダウンロードし、グラフィカルインストーラーを起動します。

[Stack Builder または StackBuilder Plus の使用](#) を参照してください。

前提条件

Stack Builder で Migration Toolkit のインストールを実行するには、Java JVM（バージョン 1.8.0 以降）を配置する必要があります。

[Adoptium](#) は、完全にオープンソースのビルド済み OpenJDK バイナリを提供します。

Java 実行可能ファイルは、検索パス（Windows%PATH%）にある必要があります。Windows では、次のコマンドを使用して検索パスを設定し、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。

```
SET PATH=javadir;%PATH%
```

直接インストール

グラフィカルインストーラーをダウンロードした後、インストールウィザードを起動して、十分な特権（スーパーユーザーまたは管理者）を引き受け、インストーラーアイコンをダブルクリックします。プロンプトが表示されたら、パスワードを入力します。

Windows の一部のバージョンでは、管理者権限でインストーラーを起動するには、インストーラーアイコンを右クリックし、コンテキストメニューから **RunasAdministrator** を選択する必要があります。

このトピックの [グラフィカルインストーラーの使用](#) セクションに進みます。

Stack Builder または StackBuilder Plus の使用

PostgreSQL を使用している場合は、Stack Builder でグラフィカルインストーラーを呼び出すことができます。

[スタックビルダーの使用](#) を参照してください。

1. スタックビルダーで、モジュール選択ページが表示されるまでプロンプトに従います。

[Welcome] ページで、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** を選択して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** をクリックして続行します。

1. **登録必須および試用製品** ノードを展開します。
2. **EnterpriseDBTools** ノードを展開し、**MigrationToolkit** を選択します。
3. [グラフィカルインストーラーの使用](#) に進みます。

EDB Postgres Advanced Server を使用している場合、StackBuilder Plus でグラフィカルインストーラーを呼び出すことができます。

Using StackBuilder Plus を参照してください。

1. StackBuilder Plus で、モジュール選択ページが表示されるまでプロンプトに従います。

[Welcome] ページで、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** を選択して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** をクリックして続行します。

1. **アドオン、ツール、およびユーティリティ** ノードを展開し、**EnterpriseDBMigrationToolkit** を選択します。
2. [グラフィカルインストーラーの使用](#) に進みます。

グラフィカルインストーラーの使用

1. インストール言語を選択し、**OK** を選択します。

1. Setup Migration Toolkit ページで、**Next** を選択します。
2. 使用許諾契約を読みます。契約に同意する場合は、**私は契約に同意します** オプションを選択し、**次へ** を選択します。
3. Migration Toolkit をインストールするディレクトリを参照するか、インストーラーがデフォルトの場所に Migration Toolkit をインストールできるようにします。**Next** を選択します。
4. [Ready to Install] ページで、[**Next**] を選択します。

情報ボックスには、選択したコンポーネントのインストールの進行状況が表示されます。これには数分かかる場合があります。

1. インストールが完了したら、**Finish** を選択します。

Migration Toolkit をインストールした後、移行を実行する前に、適切な JDBC ドライバーをインストールする必要があります。詳細については、[Installing a JDBC driver](#) を参照してください。

4.4.5 Installing a JDBC driver

Migration Toolkit には、Java バージョン 1.8.0 以降が必要です。

ドライバの選択

使用する JDBC ドライバーは、移行元または移行先のデータベースによって異なります。

- EDB Postgres Advanced Server との間で移行する場合は、EDB JDBC ドライバーを使用します。最新のドライバーをダウンロードするには、EDB Downloads ページの [EDB Connectors](#) を参照してください。インストール手順については、

[Installing and configuring EDB JDBC Connector](#) を参照してください。

注釈: EDB JDBC ドライバーは Mac OS では利用できません。

- PostgreSQL との間で移行する場合は、PostgreSQL JDBC ドライバーを使用します。サポートされている最新のドライバーをダウンロードするには、PostgreSQL Downloads ページの [JDBC drivers section](#) を参照してください。
- Oracle、MySQL、Microsoft SQL Server、または Sybase から移行する場合は、無料で入手可能なソース固有の JDBC ドライバーを使用します。 - [Oracle JDBC](#)
- [MySQL JDBC](#)
- [Microsoft SQL Server JDBC](#)
- [Sybase ASE \(jTDS\) JDBC](#)

注釈: オープンソースの jTDS ドライバーは、古いバージョンの Microsoft SQL Server（バージョン 2012 以前）もサポートしており、Migration Toolkit で使用できます。Migration Toolkit でサポートされる SQL Server の新しいバージョンには、Microsoft が提供する SQL Server 用の JDBC ドライバーをお勧めします。

lib にドライバーを追加する

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<mtk_install_dir>/lib ディレクトリに移動します。

!!!note 「EDB JDBC ドライバーの場合」 Migration Toolkit をインストールする前に EDB JDBC ドライバーがインストールされている場合、そのコピーを<mtk_install_dir>/lib ディレクトリに配置する必要はありません。Migration Toolkit のインストール中に、edb-jdbc18.jar ファイルへのシンボリックリンクが作成されます。

Migration Toolkit をインストールした後、次のコマンドを実行して、シンボリックリンクが存在するかどうかを確認できます。

```
ls -l /usr/edb/migrationtoolkit/lib/edb-jdbc18.jar
```

第5章 Upgrading

既存の RPM インストールがある場合は、`yum`（またはオペレーティングシステムに適したパッケージマネージャー）を使用して、リポジトリ構成ファイルをアップグレードし、より新しい製品バージョンに更新できます。`edb.repo` ファイルを更新するには、スーパーユーザー権限を引き受けて、次のように入力します。

```
yum upgrade edb-repo
```

`yum` は `edb.repo` ファイルを更新して、`edb.repo` ファイルで指定された資格情報で接続するように構成された現在の EnterpriseDB リポジトリへのアクセスを有効にします。次に、`yum` を使用して移行ツールキットをアップグレードできます。

```
yum upgrade edb-migrationtoolkit
```


第6章 Specifying connection properties

Migration Toolkit は、移行プロセス中に `toolkit.properties` ファイルに保存された構成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 `/usr/edb/migrationtoolkit/etc` にあります。

Windows では、ファイルは `C:\Program Files\edb\mtk\etc` にあります。

Mac OS では、ファイルは `/Library/edb/mtk/etc` にあります。

図は、サンプルの `toolkit.properties` ファイルを示しています。

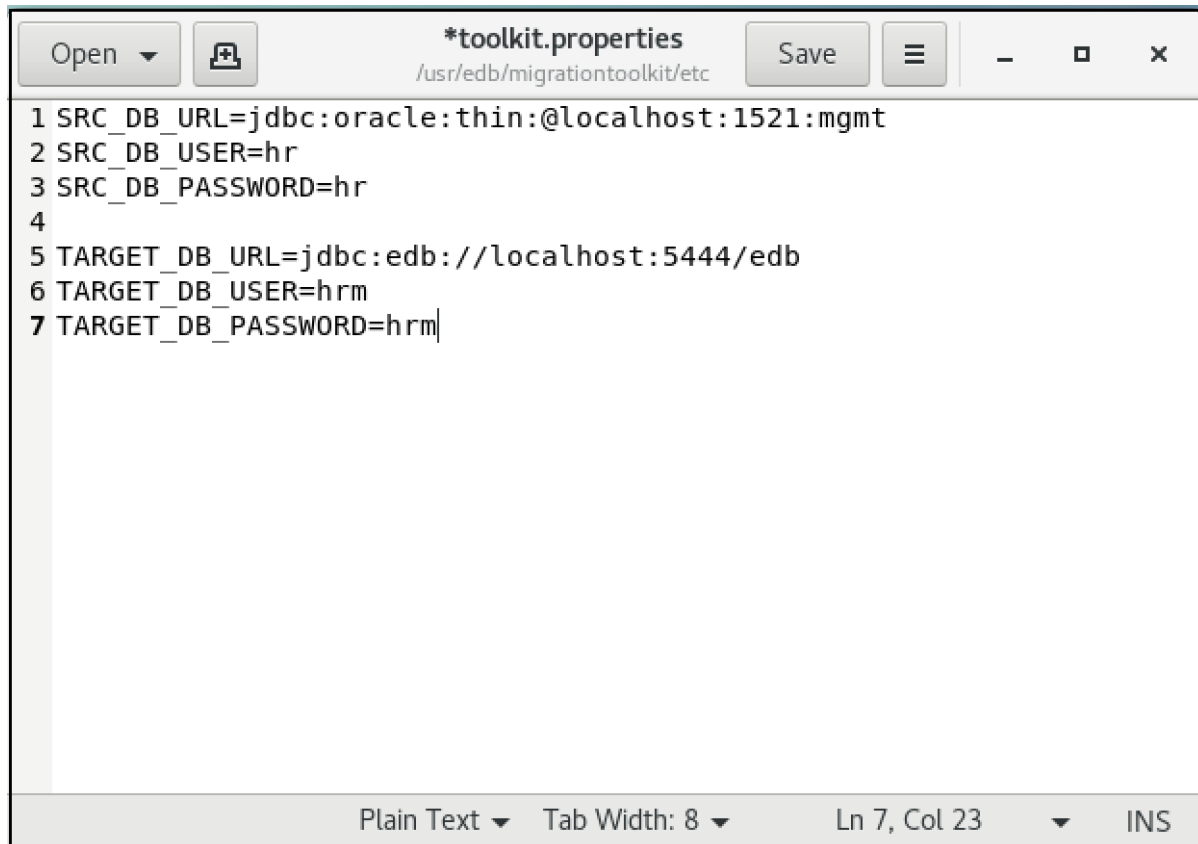


図 1: A typical toolkit.properties file.

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。各ソースデータベースの URL の形成の詳細は次のとおりです。
- `SRC_DB_USER` は、ソースデータベースで（十分な権限を持つ）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。

注釈: コマンドラインで指定されない限り、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 EDB Postgres Advanced Server であると想定しています。他のソースまたはターゲットデータベースの場合、[Oracle 以外のソースデータベースからの移行](#)の説明に従って、`-sourcedbtype` または `-targetdbtype` オプションを指定します。

BigAnimal でターゲットデータベースを指定するには、[BigAnimal URL の定義](#)を参照してください。

6.1 Advanced Server の URL の定義

Migration Toolkit は、次のプラットフォームから Advanced Server への移行に役立ちます。

- Oracle
- PostgreSQL
- MySQL
- Sybase
- SQL 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明確なリストについては、

Functionality overview を参照してください。

Migration Toolkit は、ソースデータベースと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 の 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}` — 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します。データをテーブルに移行する場合、指定されたユーザーには、各ターゲットテーブルの挿入、切り捨て、および参照特権も必要になる場合があります。
- `{TARGET_DB_PASSWORD|SRC_DB_PASSWORD}` — Advanced Server 特権ユーザーのパスワードに設定します。

6.2 PostgreSQL URL の定義

Migration Toolkit は、次のプラットフォームから PostgreSQL への移行に役立ちます。

- Oracle
- Advanced Server
- MySQL
- SQL 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、[Functionality overview](#) を参照してください。

Migration Toolkit は、ソースデータベースと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ています。

- `jdbc` — プロトコルは常に `jdbc` です。
- `postgresql` — PostgreSQL を利用している場合、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} —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します。データをテーブルに移行する場合、指定されたユーザーには、各ターゲットテーブルの挿入、切り捨て、および参照特権も必要になる場合があります。
- {SRC_DB_PASSWORD|TARGET_DB_PASSWORD} —特権 PostgreSQL ユーザのパスワードに設定します。

6.3 BigAnimal URL の定義

Migration Toolkit は、次のプラットフォームから BigAnimal への移行に役立ちます。

- Oracle
- Advanced Server
- PostgreSQL
- MySQL
- SQL Server

Migration Toolkit は、ソースデータベースと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

BigAnimal に移行する場合、`TARGET_DB_URL` は JDBC URL の形式を取ります。例：

```
jdbc:<postgres_type>://<host_name>[:<port>]/<database_id>?sslmode=<sslmode>
```

注釈： ターゲットデータベースの URL に必要な値の多くは、BigAnimal ポータルから入手できます。BigAnimal で、クラスターを選択し、[**Connect**] タブに移動して値を見つけます。!!!

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。
- `postgres_type` —サブプロトコルは `Postgres` タイプです。Advanced Server を使用している場合は `edb` を、PostgreSQL を使用している場合は `postgresql` を指定します。
- `<host_name>` —クラスターのホスト名。BigAnimal ポータルの **Connect** タブの **Host** フィールドからコピーできます。
- `<port>` —データベースリスナーが監視しているポート番号。BigAnimal ポータルの **Connect** タブの **Port** フィールドからコピーできます。
- `<database_id>` —ターゲットデータベースの名前。これを、移行ターゲットデータベースとして使用するクラスター内のデータベースの名前に設定します。クラスターのデフォルトデータベースの名

前は、BigAnimal ポータルの **Connect** タブの **Dbname** フィールドに表示されます。多くの場合、移行ターゲットとして使用する別のデータベースが作成されます。

- **TARGET_DB_USER** —特権データベースユーザーの名前を指定します。BigAnimal ポータルの **Connect** タブの **User** フィールドからコピーできます。
- **TARGET_DB_PASSWORD** —指定されたユーザーのパスワードが含まれています。
- **sslmode** —「**require**」または「**verify-full**」のいずれか。
Recommended settings for SSL mode を参照してください。BigAnimal ポータルの **Connect** タブの **ServiceURI** 値の最後にリストされています。

6.4 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に役立ちます。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。接続情報には以下が含まれる必要があります。

- 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 を含める必要があります。最初のフォームは次のとおりです。

```
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、次の情報が含まれています。

- **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** —ドライバの種類。 **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** —特権 Oracle ユーザーの名前を指定します。Oracle ユーザーは、オブジェクトを Oracle から Advanced Server に移行する 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、目的のすべてのデータベースオブジェクトが移行されるようにすることができます。

- SRC_DB_PASSWORD —指定されたユーザーのパスワードが含まれています。

6.5 SQL Server の 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に役立ちます。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。接続情報には以下が含まれる必要があります。

- 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

Microsoft JDBC URL（推奨）または *JTDS の URL* の形式を取ります。

6.5.1 Microsoft JDBC URL

既定では、Microsoft JDBC Driver for SQL Server は、クライアントと SQL Server 間のすべての通信に TLS 暗号化を使用します。TLS 暗号化を無効にする場合は、`encrypt` を `false` に設定します。JDBC Type 4 ドライバーを使用した Microsoft SQL Server への接続の詳細については、[Building the connection URL](#) を参照してください。

`jdbc:sqlserver://<server_name>:<port>[;databaseName=<database_id>][;encrypt=false]`

SQL サーバーの 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。
- `sqlserver` —サーバータイプは常に `sqlserver` です。
- `<server_name>` —ソース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 —特権 SQL Server ユーザーの名前を指定します。
- SRC_DB_PASSWORD —指定されたユーザーのパスワードが含まれています。

6.5.2 JTDS の URL

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

Tip: JTDS ドライバーは、古いバージョンの Microsoft SQL Server をサポートするオープンソースの JDBC 3.0 Type 4 ドライバーです。 <http://jtds.sourceforge.net/> を参照してください。 Migration Toolkit を使用して Microsoft SQL Server の新しいバージョンを接続する場合、Microsoft JDBC Driver for SQL Server をお勧めします。

SQL サーバーの 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。
- sqlserver — サーバータイプは常に sqlserver です。
- <server_name> — ソース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 — 特権 SQL Server ユーザーの名前を指定します。
- SRC_DB_PASSWORD — 指定されたユーザーのパスワードが含まれています。

6.6 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支援します。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。接続情報には以下が含まれる必要があります。

- 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ています。

- 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> — ソース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` — 特権 MySQL ユーザーの名前を指定します。
- `SRC_DB_PASSWORD` — 指定されたユーザーのパスワードが含まれています。

注釈:

- データ型 `tinyInt(1)` を使用して 0 および 1 以外のバイト値を MySQL ソースデータベースに保存する場合、MySQL Connector/J JDBC Driver URL にオプションのパラメーター `?tinyInt1isBit=false` を必ず追加してください。

-
- MySQL Connector/J JDBC Driver 8.0.26 のバグにより、特定の場合に外部キー制約の移行が失敗します。EDB は、Migration Toolkit を使用してデータを移行するためにこのドライバーを使用しないことをお勧めします。代わりに、MySQL Connector/J JDBC Driver 8.0.30 以降を使用してください。

このバグの詳細については、[MySQL bug report](#) を参照してください。

TINYINT(1) は、PostgreSQL / EDB Postgres Advanced Server の BIT(1) にマップされます（これは、場合によっては期待されない場合があります）。しかし、MySQL JDBC ドライバーは BIT(1) として報告するため、Migration Toolkit はそれを PostgreSQL / EDB Postgres Advanced Server の BIT(1) にマッピングします。

- 128～127 の値を保持できる TINYINT(1) とは異なり、BIT(1) は 0 と 1 のみを保持できます。したがって、このフィールドに 0 と 1 以外の値を保存する場合は、MySQL JDBC ドライバーのデフォルトの動作を変更する必要がありますドライバーが TINYINT として報告するようにします。

この動作を変更するには、接続文字列で次の URL オプションを設定します。

```
jdbc:mysql://localhost:3306/world?tinyInt1isBit=false
```

この場合、JDBC ドライバーは TINYINT(1) を TINYINT として報告し、ターゲット PostgreSQL / EDB Postgres Advanced Server の SMALLINT にマップされます。

6.7 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に役立ちます。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。接続情報には以下が含まれる必要があります。

- 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>
```

Tip: Sybase ASE および古いバージョンの Microsoft SQL Server 用のオープンソース JDBC 3.0 タイプ 4 ドライバーについては、<http://jtds.sourceforge.net/> を参照してください。

Sybase の 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> — ソース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 — 特権 Sybase ユーザの名前を指定します。
- SRC_DB_PASSWORD — 指定されたユーザーのパスワードが含まれています。

第7章 Invoking Migration Toolkit

Migration Toolkit をインストールし、 `toolkit.properties` file の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、Migration Toolkit 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付けは、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
```

移行の詳細を制御する方法については、[Migration Toolkit command options](#) を参照してください。

注釈: Migration Toolkit の呼び出し時に次のエラーが表示される場合、`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 を実行しているオペレーティングシステムのユーザーアカウントは、ファイルに対する最小限の読み取り権限を持つ `toolkit.properties` ファイルの所有者である必要があります。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権限も与えてはなりません。以下は、ユーザー `enterprisedb` が Migration Toolkit を実行している場合の、推奨されるファイル許可の例です。

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

7.1 バイナリゼロ (NULL 文字) が埋め込まれた文字データのインポート

Migration Toolkit は、値が NULL の列のインポートをサポートしています。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は、`-replaceNullChar` オプションを使用して、NULL 文字を単一の NULL 以外の置換文字に置き換えます。

注釈:

- 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` オプションと連携しません。
-

データを移行したら、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ゼロに置き換えます。

7.2 Oracle からスキーマを移行する

コマンドラインで指定されない限り、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 EDB Postgres Advanced Server であると想定しています。Linux でスキーマ全体を移行するには、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 でスキーマ全体を移行するには、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>
```

そこで：

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

注釈:

- 移行されたスキーマのデフォルトのデータベースユーザーが自動的に移行される場合、カスタムプロファイルが存在する場合はデフォルトの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。 - PostgreSQL のデフォルト行のサイズは 8 KB に制限されています。これは、各テーブルの行が単一の 8 KB ブロックに収まる必要があることを意味します。そうしないと、エラーが発生し、たとえば、1600 列の INT を持つテーブルを作成し、すべての列にデータを挿入できます。ただし、INT は 4 バイトとして格納されるため、BIGINT 列で同じことを行うことはできませんが、各 BIGINT にはより多くの領域（8 バイト）が必要です。詳細については、PostgreSQL ドキュメントの [PostgreSQL Limits](#) を参照してください。
-

コマンド名の後にスキーマ名のコンマ区切りリストを指定することにより、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>
```

7.3 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 であり、ターゲットデータベースが EDB Postgres 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 <source_type> -targetdbtype <target_type> [options, ...]
<schema_name>;
```

-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、次のいずれかの値になります。

To migrate from:	Specify:
Oracle	oracle (the default value)
MySQL	mysql
SQL Server	sqlserver
Sybase	sybase
PostgreSQL	postgres or postgresql
EDB Postgres 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、次のいずれかの値です。

To migrate to:	Specify:
EDB Postgres Advanced Server	enterprisedb
PostgreSQL	postgres or postgresql

<schema_name>

<schema_name> は、移行する `toolkit.properties` ファイルで指定されたソースデータベース内のスキーマの名前です。少なくとも 1 つの **<schema_name>** 値を含める必要があります。

次の例では、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 Linux システムの MySQL データベースから EDB Postgres Advanced Server ホストに移行します。コマンドには、**- sourcedbtype** および **-targetdbtype** オプションが含まれています。

Linux では、次のコマンドを使用します。

```
$ ./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ます。

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

Windows では、次のコマンドを使用します。

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

7.3.1 Migration Toolkit command options

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、`-allSchemas` オプションをコマンドに追加します。

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

注釈:

- `-allSchemas` パラメーターは、Oracle、EDB Postgres Advanced Server、および PostgreSQL ソースデータベースでのみサポートされています。 Sybase、MS SQL Server、および MySQL ソースデータベースではサポートされていません。 - Migration Toolkit は、データの移行中にユーザーが作成したトリガーを無効にします。ただし、PostgreSQL / EDB Postgres Advanced Server は `ALTER TABLE` の `DISABLE TRIGGER USER` 句を使用して継承された子パーティショントリガーを無効にすることを許可していないため、トリガーはパーティションテーブルのデータ移行中にアクティブになり、予期しない結果を引き起こす可能性があります。

Migration Toolkit で動作するコマンドオプションは、表に示すように、動作ごとにグループ化されています。

Feature	Relevant options
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>

次のページに続く

表 3 – 前のページからの続き

Feature	Relevant options
スキーマオブジェクト選択オプション	-allTables, -tables, -excludeTables, -constraints, -ignoreCheckConstFilter, -skipCKConst, -skipFKConst, -skipColDefaultClause, -indexes, -triggers, -allViews, -views, -excludeViews, -allSequences, -sequences, -allProcs, -procs, -allFuncs, -funcs, -checkFunctionBodies, -allPackages, -packages, -allDomains, -allQueues, -queues, -allRules,
オフライン移行オプション	-truncLoad, -enableConstBeforeDataLoad, -retryCount, -safeMode, -fastCopy, -analyze, vacuumAnalyze, -replaceNullChar, -copyDelimiter, -batchSize, -cpBatchSize, -lobBatchSize, -fetchSize, -filterProp, -customColTypeMapping, -customColTypeMappingFile
Oracle 固有のオプション	-allUsers, -users, -allProfiles, -profiles, -importPartitionAsTable, -objectTypes, -copyViaDBLinkOra, -allDBLinks, -allSynonyms, -allPublicSynonyms, -allPrivateSynonyms, -useOraCase, -skipUserSchemaCreation
その他のオプション	-help, -logDir, -logFileCount, -logFileSize, -logBadSQL -verbose, -version
並列データロードの移行オプション	-loaderCount, -parallelLoadRowLimit, -tableLoaderLimit

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。オフライン移行中に、Migration Toolkit は選択された各オブジェクトの定義を読み取り、SQL スクリプトを作成します。後で実行すると、Postgres の各オブジェクトを複製します。

注釈: 次の例では、Linux で Migration Toolkit を呼び出します。Windows で Migration Toolkit を呼び出すには、`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、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でも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>
```

スキーマオブジェクト定義を省略して、データのみをオフライン移行を実行するには、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 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>
```

既定では、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>
```

注釈: -`singleDataFile` オプションは、プレーン SQL 形式で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。

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 EDB Postgres 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ます。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にデータを入力します。

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

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が Oracle で、ターゲットデータベースが EDB Postgres Advanced Server であると想定します。- **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 はスキーマを移行するとき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インポートを選択できます。

-sourcedbtype <source_type>

-sourcedbtype オプションは、ソースデータベースのタイプを指定します。**source_type** の場合、次のいずれかの値を使用します。**mysql**、**oracle**、**sqlserver**、**sybase**、**postgresql** または **enterprisedb**。**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のタイプを指定します。**target_type** の場合、次のいずれかの値を使用します：**enterprisedb**、**postgres**、または **postgresql**。**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。このオプションを **-dataOnly** オプションと一緒に使用することはできません。

-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。このオプションを **-schemaOnly** オプションと一緒に使用することはできません。

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]

このオプションを **true** に設定すると、Migration Toolkit は既存のスキーマと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>

`-targetSchema` オプションを使用して、移行したスキーマの名前を指定します。複数のスキーマを移行する場合は、各スキーマの名前を、間にスペース文字を入れないコンマ区切りのリストで指定します。
`-targetSchema` オプションを指定しない場合、新しいスキーマの名前はソーススキーマの名前と同じです。

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

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、間にスペース文字を含まないテーブル名のコンマ区切りリストです（例、`-tables emp,dept,acctg`）。

`-excludeTables <table_list>`

選択したテーブルを移行から除外します。 `table_list` は、間にスペース文字を含まないテーブル名のコンマ区切りリストです（例、`-excludeTables emp,jobhist`）。このオプションは、スキーマのすべてのテーブルが `-allTables` オプションを介した移行用に選択されている場合に適用されます。

`-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。 `-constraints` パラメーターを指定して制約とデフォルト句を Sybase データベースから移行するときに `-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た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た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場合、または **-allTables** または **-tables <table_list>** オプションを指定した場合に有効です。

-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場合、または **allTables** または **-tables <table_list>** オプションを指定した場合に有効です。

-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、ソースから動的ビューとマテリアライズドビューを移行します。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。

-views <view_list>

指定し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、間にスペース文字を含まないビュー名のコンマ区切りリストです（例、**-views all_emp,mgmt_list,acct_list**）。

-excludeViews <view_list>

選択したビューを移行から除外します。**view_list** は、間にスペース文字を含まないビュー名のコンマ区切りリストです（例、**-excludeViews all_emp,acct_list**）。このオプションは、スキーマからのすべてのビューが **-allViews** オプションを介した移行用に選択されている場合に適用されます。

-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、間にスペース文字を含まないシーケンス名のコンマ区切りリストです。

-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、間にスペース文字を含まないプロシージャ名のコンマ区切りリストです。

-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、間にスペース文字を含まない関数名のコンマ区切りリストです。

-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。この検証を無効にすると、関数に前方参照が含まれる場合のエラーが回避されます。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、間にスペース文字を含まないパッケージ名のコンマ区切りリストです。

-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 ビルトイン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。EDB Postgres Advanced Server がソースデータベースである場合、**-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 EDB Postgres 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、間にスペース文字を含まないキュー名のコンマ区切りリストです。これらは、DBMS_AQ および DBMS_AQADM ビルトイン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である場合、**-objectTypes** も指定する必要があります。EDB Postgres Advanced Server がソースデータベースである場合、**-allDomains** および **-allTables** も指定する必要があります。Oracle および EDB Postgres 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。

移行オプション

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。

-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** オプションとともに使用します。

-enableConstBeforeDataLoad

非パーティション化ソース表がパーティション化表にマップされる場合は、**-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を有効にします。これには、データを個々のパーティションにリダイレクトするトリガーも含まれます。**-enableConstBeforeDataLoad** は、**-truncLoad** パラメーターも指定する場合にのみ有効です。

-retryCount [<value>]

マルチスキーマの移行を実行している場合、クロススキーマの依存関係が原因で最初の移行の試行中に移行に失敗したオブジェクトは、後の移行中に正常に移行される場合があります。-retryCount オプションを使用して、最初の移行の試行中に失敗したオブジェクトの移行を Migration Toolkit が試行する回数を指定します。0 より大きい値を指定します。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。移行ですべてのレコードの転送に失敗した場合、障害ポイント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 WAL ロギングをバイパスし、最適化された方法で COPY 操作を実行するように指定されます。デフォルトでは無効になっています。-fastCopy オプションの使用を選択した場合、移行が中断された場合、ターゲットデータベース内の移行されたデータを回復できない場合があります。

-replaceNullChar <value>

Migration Toolkit は、値が NULL の列のインポートをサポートしています。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は、-replaceNullChar オプションを使用して、NULL 文字を単一の非 NULL 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。

データを移行したら、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ゼロに置き換えます。

-analyze

-analyze オプションを含めて、ターゲットデータベースに対して Postgres ANALYZE オペレーションを呼び出します。オプティマイザーは、ANALYZE オペレーションによって収集された統計を参照し、その情報を使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 オペレーションによって収集された統計を参照し、その情報を使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デッドタプルが占有するストレージスペースを再利用します。

-copyDelimiter

テーブルデータロード時の COPY コマンドで区切り文字として使用する 1 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。メモリ不足例外が発生した場合は、-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 データ型の場合、一括ロードする行数を指定します。BYTEA、BLOB、または CLOB などのラージオブジェクト型 (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に、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。**-fetchSize** オプションを含めて、大きなテーブルを移行するときのこの落とし穴を回避します。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、テーブル内のすべての行を 1 回のネットワークラウンドトリップで 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。ヒープ領域不足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 に強制的にテーブルデータを 1 行ずつロードさせます。

-filterProp <file_name>

<file_name> は、key=value ペアに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修飾であってはなりません。- Oracle のテーブル名は大文字と小文字を区別しませんが、Oracle はデフォルトで大文字になります。一方、Postgres はデフォルトで小文字になります。Oracle データを移行するときは、使用している大文字と小文字が Oracle の大文字と小文字と一致することを確認してください。そうしないと、Migration Toolkit は一致できず、制約を無視します。右側は、移行される各行について true でなければならない条件を指定します。

たとえば、

```
COUNTRIES=COUNTRY_ID<>'AR'
```

AR と等しくない COUNTRY_ID 値を持つ国のみを移行します。この制約は、COUNTRIES テーブルに適用されます。

複数のテーブルの条件を指定することもできます。ただし、各テーブルの条件は、プロパティファイルの新しい行に記述する必要があります。

例：

プロパティファイルの次のエントリは、EMPLOYEES および DEPARTMENTS テーブルから関連するデータのみを移行します。

```
EMPLOYEES=(LAST_NAME IN ('Grant','Weiss') AND PHONE_NUMBER LIKE '650%')
```

```
DEPARTMENTS=(DEPARTMENT_ID BETWEEN 10 AND 30)
```

-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。各ペアの右側は、その列が想定するデータ型の名前を指定します。<column_list> のセミコロン区切りリスト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 INTEGER 型にマップするには、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、基準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、'\\.' を使用して '.' 文字を表します。

```
EMP\\.*EMP_ID=INTEGER
```

または、<property_file> に複数のカスタム型マッピングを含めることができます。

```
-customColTypeMappingFile <property_file>
```

ファイル内の各エントリを key=value ペアで、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。各ペアの右側は、その列が想定するデータ型の名前を指定します。<property_file> で使用される場合、'\\' 文字は、オペレーティングシステムに関係なく、エスケープ文字列として機能します。

並列データロードの移行オプション

比較的多数のテーブルまたは大きなサイズのテーブルを扱う場合、通常のデータ移行には時間がかかります。この問題の解決策として、Migration Toolkit には、スキーマレベルとテーブルレベルでデータを並列に移行できる並列データロードアルゴリズムがあります。

スキーマレベルでの並列データ移行では、マルチスレッドを使用して、同時に複数のテーブルを移行します。テーブルレベルでのパラレルデータロードは、マルチスレッドを使用して、単一のテーブルをソースデータベースからターゲットデータベースに、より効率的かつ短時間で移行します。

注釈:

- パラレルロードは、大きなテーブルでのみ役立ちます。小さなテーブルの場合、並列スレッドにはより多くの時間がかかる場合があります。したがって、比較的大きなテーブルには、テーブルレベルでパラレルデータロードを使用することをお勧めします。
-

- パラレルデータロードは、主/一意キーのないテーブルには適用されません。
- fastCopy パラメーターは、パラレルデータロードでは使用できません。
- パラレルロード機能は、ソース Oracle、EDB Postgres Advanced Server、および PostgreSQL データベースにのみ適用されます。

次のオプションは、MTK がデータを並行して移行する方法を制御します。

`-loaderCount` [`<value>`]

`-loaderCount` オプションを使用して、プールで使用可能な並列スレッドの数を指定し、データのロードを並列で実行します。このオプションは、Migration Toolkit を実行しているホストシステムにハイエンドの CPU および RAM リソースがある場合に特に役立ちます。`value` にはゼロ以外の正の数を指定できますが、値は CPU コアの数を超えないことをお勧めします。たとえば、デュアルコア CPU の最適値は 2 です。デフォルトは 1 です。

注釈: マルチスレッドを使用する場合、実際のスレッド数とテーブルデータサイズによっては、Migration Toolkit のメモリヒープサイズの調整が必要になる場合があります。

`-parallelLoadRowLimit` [`<value>`]

`-parallelLoadRowLimit` オプションを使用して、テーブルレベルで並列にデータロードを実行するためにテーブルに必要な最小行数を指定します。`<value>` パラメーターは 0 より大きい必要があります。デフォルトは 100,000 です。

たとえば、`-parallelLoadRowLimit` の値が 10,000 の場合、行数が 10,000 以上のテーブルのみがテーブルレベルで並列にロードされます。他のテーブルは、シーケンシャルモードまたはスキーマレベルで移行します。

`-tableLoaderLimit` [`<value>`]

`-tableLoaderLimit` オプションを使用して、スレッドプールから割り当てるスレッドの最大数を指定し、単一のテーブルデータをテーブルレベルで並列チャンクにロードします。`<value>` パラメーターは、`-loaderCount` の値以下でなければなりません。デフォルト値は、`-loaderCount` オプションの値と同じです。このオプションは、`-loaderCount` のデフォルト以外のカスタム値に基づいて暗黙的に変更されます。

たとえば、`-loaderCount` が 4 に設定され、`-tableLoaderLimit` が指定されていない場合、`-tableLoaderLimit` も 4 に設定されます。したがって、すべてのスレッドを単一のテーブルロードに使用することを意図していない場合は、`-tableLoaderLimit` をより低い値に減らします。

並列データロードアルゴリズムの高レベルの概要

アルゴリズムは 3 つのステップで動作します。

1. テーブルとテーブルチャンクのリストは、移行するテーブルの順序付けられたリストから作成され、テーブル名でソートされます。リストは、`-parallelLoadRowLimit` および `-tableLoaderLimit` オプション値に基づいて作成されます。

1. `-loaderCount` オプションの値に基づいてスレッドプールが作成されます。

1. 並列データロードアルゴリズムは、テーブルとテーブルチャンクのリストから各テーブル/チャンクのタスクを作成し、スレッドプールで実行します。スレッドの数だけタスクが並列実行されます。タスクの実行が完了するとすぐに、そのスレッドはリストから次のタスクの実行を開始します。並列データの読み込みは、リストの最後のタスクが完了するとすぐに完了します。

例：

このシナリオでは、並列データのロードを使用して 4 つのテーブル-tab1, tab2, tab3, tab4 を移行します。

```
# ./runMTK.sh -sourcedbtype enterprisedb -targetdbtype enterprisedb -loaderCount 4 -
→parallelLoadRowLimit 10000 -tableLoaderLimit 2 -tables tab1,tab2,tab3,tab4 public
```

テーブル tab1 には 50,000 行、テーブル tab2 には 9,500 行、テーブル tab3 には 50,001 行、テーブル tab4 には 9,999 行があります。ステップ 1 のテーブルのリストは tab1, tab2, tab3、および tab4 です。

テーブル tab1 および tab3 のみが、テーブルレベル (-parallelLoadRowLimit 10000) でのパラレルロードの資格があります。

-tableLoaderLimit オプションの値は 2 であるため、テーブル tab1 は 2 つのチャンク、tab1_chunk1 および tab1_chunk2 にロードされます。各チャンクの行数は、totalRowCount/tableLoaderLimit に基づいて決定されます。この場合、両方のチャンクには 25,000 行 (50,000/2) があります。

テーブル tab3 も 2 つのチャンク、tab3_chunk1 および tab3_chunk2 でロードされます。テーブル tab3 の行の総数が奇数 (50,001) であるため、最初のチャンクは追加の行を収容するため、tab3_chunk1 には 25,001 行、tab3_chunk2 には 25,000 行があります。

作成されたテーブルとテーブルチャンクのリストは次のとおりです。

tab1_chunk1	(25,000 rows)
tab1_chunk2	(25,000 rows)
tab2	(9500 rows)
tab3_chunk1	(25,001 rows)
tab3_chunk2	(25,000 rows)
tab4	(9999 rows)

4 つのスレッド T1, T2, T3, T4 を持つスレッドプールが作成されます。手順 1 で作成したリストからテーブル/チャンクを移行するタスクは、スレッドプールで実行されます。

T1 → tab1_chunk1	(25,000 rows)
T2 → tab1_chunk2	(25,000 rows)
T3 → tab2	(9500 rows)
T4 → tab3_chunk1	(25,001 rows)

スレッド T3 が最初に終了し、リスト内の次のチャンクである tab3_chunk2 の実行を開始します。

T1 → tab1_chunk1	(25,000 rows)
T2 → tab1_chunk2	(25,000 rows)
T3 → tab3_chunk2	(25,000 rows)
T4 → tab3_chunk1	(25,001 rows)

スレッド T1 は 2 番目に終了し、tab4 の実行を開始します。

T1 → tab4	(9999 rows)
T2 → tab1_chunk2	(25,000 rows)

(次のページに続く)

(前のページからの続き)

```
T3 → tab3_chunk2      (25,000 rows)
T4 → tab3_chunk1      (25,001 rows)
```

このように、4つのスレッドを使用して、4つのテーブルを混合スキーマとテーブルレベルで並行して移行します。コマンドオプションを変更して `-tableLoaderLimit` を 1 に設定する場合：

```
# ./runMTK.sh -sourcedbtype enterprisedb -targetdbtype enterprisedb -loaderCount 4 -
→parallelLoadRowLimit 10000 -tableLoaderLimit 1 -tables tab1,tab2,tab3,tab4 public
```

純粋なスキーマレベルの並列処理を行うことができます。各テーブルは1つのスレッドで同時に処理されます。

```
T1 → tab1      (50,000 rows)
T2 → tab2      (9500 rows)
T3 → tab3      (50,001 rows)
T4 → tab4      (9999 rows)
```

システムリソースの推奨事項と制約

MTK を実行しているホストシステムで使用可能な CPU および RAM リソースに応じて、スレッドの数を選択します。CPU、IO、およびディスク使用率に関してソースデータベースに特定のオーバーヘッドがあるため、特定のしきい値を超える並列スレッド数を超えると、MTK のパフォーマンスが低下します。

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オブジェクト型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 EDB Postgres Advanced Server データベースに移行する場合にのみサポートされます。

`-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、間にスペース文字を含まないユーザー/ロール名のコンマ区切りリストです（例、`-users MTK, SAMPLE, acctg`）。`-users` オプションは、Oracle データベースから EDB Postgres Advanced Server データベースに移行する場合にのみサポートされます。

`-allProfiles`

ソースデータベースからすべてのカスタム（つまり、ユーザーが作成した）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た次のパスワードパラメーター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ーなど、他のすべてのプロファイルパラメーター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は、Oracle データディクショナリビュー DBA_PROFILES に対する SELECT 権限を持っている必要があります。

注釈: - *allProfiles* オプションは、Oracle データベースから EDB Postgres Advanced Server データベースに移行する場合にのみサポートされます。

-profiles <profile_list>

選択したカスタム（つまり、ユーザーが作成した）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、間にスペース文字を含まないプロファイル名のコンマ区切りリストです（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ー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は、Oracle データディクショナリビュー DBA_PROFILES に対する SELECT 権限を持っている必要があります。

注釈: -profiles オプションは、Oracle データベースから EDB Postgres Advanced Server データベースに移行する場合にのみサポートされます。

-importPartitionAsTable <table_list>

-importPartitionAsTable パラメーターを含めて、Oracle ホストにあるパーティション表の内容を単一の非パーティション表にインポートします。table_list は、間にスペース文字を含まないテーブル名のコンマ区切りリストです（例、-importPartitionAsTable emp,dept,acctg）。

-copyViaDBLinkOra

dblink_ora モジュールは、SQL レベルでの EDB Postgres Advanced Server -to-Oracle 接続を提供します。dblink_ora は、EDB Postgres Advanced Server データベースインストールの一部としてバンドルおよびイ

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 メソッドよりも大幅に高速です。

この例では、dblink_ora COPY API を使用して、HR スキーマからすべてのテーブルを移行します。

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

ターゲット EDB Postgres Advanced Server データベースには、dblink_ora をインストールして構成する必要があります。

dblink_ora を参照してください。

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

このオプションを選択して、Oracle データベースリンクを移行します。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、指定がない限り、ダミーのパスワード edb が置き換えられます。

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、間にスペース文字を含まない名前=値のペアのコンマ区切りリストを使用して、各データベースリンクのパスワードを指定できます。ペアの左側にリンク名、右側にパスワードの値を指定します。

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 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

-allSynonyms オプションを含めて、すべてのパブリックおよびプライベートシノニムを Oracle データベースから EDB Postgres Advanced Server データベースに移行します。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

-allPublicSynonyms オプションを含めて、すべてのパブリックシノニムを Oracle データベースから EDB Postgres Advanced Server データベースに移行します。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

-allPrivateSynonyms オプションを含めて、すべてのプライベートシノニムを Oracle データベースから EDB Postgres Advanced Server データベースに移行します。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

-useOraCase オプションを含めて、Oracle データベースから EDB Postgres 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 デフォルトの大文字の命名規則を保持し

ます。

大文字の命名規則は、テーブル、ビュー、シーケンス、プロシージャ、ファンクション、トリガー、パッケージなどで保持されます。これらのデータベースオブジェクトでは、大文字の命名規則が次に適用されます。

- データベースオブジェクトの名前
- テーブルおよびビューのカラム名、キー名、インデックス名、制約名など
- ビューの SELECT 列リスト
- プロシージャまたはファンクションヘッダーの一部であるパラメーター名

注釈: プロシージャ、ファンクション、トリガーまたはパッケージのプロシージャコード本体では、プログラムをエラーなしで実行するための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、発生した可能性の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。

注釈: `-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、`-skipUserSchemaCreation` オプションの説明を参照してください。

`-useOraCase` オプションを指定しない場合の Migration Toolkit のデフォルトの動作は、データベースオブジェクトが Oracle で引用符を囲んで明示的に作成された場合を除き、引用符を囲まずにデータベースオブジェクト名が Oracle から抽出されることです。以下は、`-offlineMigration` オプションを使用して Migration Toolkit によって生成された table 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);
```

次に、このテーブルを移行して EDB Postgres Advanced Server で作成すると、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、テーブルは次のように EDB Postgres Advanced Server に表示されます。

```
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)
```

(次のページに続く)

(前のページからの続き)

```
"dept_dname_uq" UNIQUE CONSTRAINT, btree (dname)
```

EDB Postgres 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";
```

アプリケーションで引用符で囲まれた大文字の識別子を使用する場合は、`-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");
```

次に、このテーブルを移行して EDB Postgres Advanced Server で作成すると、すべてのオブジェクト名が大文字で維持されるため、テーブルは次のように EDB Postgres Advanced Server に表示されます。

```
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、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（つまり、ユーザー名）が作成されます（ロールがまだ存在しない場合）。ロール名は小文字で作成されます。新しいロールを作成すると、小文字で同じ名前のスキーマも作成されます。

`-skipUserSchemaCreation` オプションを指定すると、移行した Oracle ユーザー名のスキーマの自動作成が防止されます。このオプションは、`-useOraCase` オプションが指定されて、大文字と小文字のみが異なる同じ名前の 2 つのスキーマが作成されるのを防ぐ場合に特に役立ちます。`-useOraCase` オプションを指定すると、Migration Toolkit が呼び出されたときにオプションリストに続い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` オプションが指定された場合、ターゲットデータベースには、1 つが小文字でもう 1 つが大文字の 2 つの同名のスキーマがあります。`-useOraCase` オプションが `-skipUserSchemaCreation` オプションとともに指定されている場合、ターゲットデータベースには大文字のスキーマのみがあります。

その他のオプション

これらの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。これら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>`

ログファイルを書き込む場所を指定するには、このオプションを含めます。`<log_path>` はログファイルを保存する場所です。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す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。0 の値を指定して、ログファイルのローテーションを無効にし、単一のログファイルを作成します。このファイルは、`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>`

新しいログファイルにローテーションする前の最大ファイルサイズ制限を MB 単位で指定するには、このオプションを含めます。`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 です。

`-logBadSQL`

このオプションを含めて、失敗したオブジェクトのスキーマ定義（DDL スクリプト）をファイルに保存します。ファイルは、エラーログに使用されたのと同じパスの下に保存され、次の形式の名前が付けられます。

`mtk_bad_sql_<schema_name_timestamp>.sql`

`schema_name` はスキーマの名前であり、`timestamp` は Migration Toolkit 実行のタイムスタンプです。

`-verbose [on|off]`

アプリケーションログメッセージを標準出力に表示します。デフォルトでは、冗長はオンです。

`-version`

Migration Toolkit のバージョンを表示します。

例

この例では、Oracle から EDB Postgres Advanced Server への移行を実行します。

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
__OUTPUT__
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
```

(次のページに続く)

(前のページからの続き)

```

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.
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_CHK
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.

```

(次のページに続く)

(前のページからの続き)

```

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
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
=====

```

(次のページに続く)

(前のページからの続き)

```

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は省略されています。移行情報は、実行の最後に Migration Summary に要約されます。

第8章 Migration errors

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。移行の概要はログファイルにも書き込まれます。

Linux では、ログファイルのデフォルトの場所は次のとおりです。

```
$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
```

Migration Toolkit の `-logdir <log_path>` オプションを使用して、代替のログファイルの場所を指定できます。

8.1 接続エラー

Migration Toolkit は、

`toolkit.properties` file からの情報を使用して、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。

8.1.1 無効なユーザー名/パスワード

- Migration Toolkit を使用して Oracle データベースから移行を実行しようとする、次のエラーが表示されます。*

```
MTK-11009: Error Connecting Database "Oracle"
DB-1017: java.sql.SQLException: ORA-01017: invalid username/password;
logon denied
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` ファイルを編集し、`SRC_DB_USER` および `SRC_DB_PASSWORD` プロパティで移行を実行する特権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。

8.1.2 接続が拒否されました：FATAL：パスワード

- Migration Toolkit で移行を実行しようとする、次のエラーが表示されます。*

```
MTK-11009: Error Connecting Database "EDB Postgres EDB Postgres 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するのに有効ではありません。

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特権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。

8.1.3 例外：ORA-28000：アカウントはロックされています

- Migration Toolkit を使用して Oracle データベースから移行を実行しようとする、次のエラーメッセージが表示されます。*

```
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 サーバーの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 および SRC_DB_PASSWORD パラメーターで移行を実行する特権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。

8.1.4 例外：oracle.jdbc.driver.OracleDriver

- Migration Toolkit で移行を実行しようとする、移行が失敗し、次のエラーメッセージが表示されます。*

```
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切な 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。完全な手順については、

Installing a JDBC driver を参照してください。

8.1.5 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

- Migration Toolkit で移行を実行しようとする、次のエラーが表示されます。*

```
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 には、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 file を編集し、 SRC_DB_URL プロパティに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す。ソースデータベースタイプの詳細については、

Building the toolkit-properties file を参照してください。

8.1.6 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

- Migration Toolkit で移行を実行しようとする、次のエラーが表示されます。*

```
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（EDB Postgres Advanced Server）の JDBC URL には、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 プロパティに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。 EDB Postgres Advanced Server の JDBC URL の形成については、 Defining an EDB Postgres Advanced Server URL を参照してください。

8.2 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可能性があります。

8.2.1 エラー：予想される最後の列の後に余分なデータ

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見つからない文字に変更します。

注釈: この例では、パイプ文字 (`|`) がテキスト ``LOS|ANGELES`` に発生し、最後の列に挿入し、`-copyDelimiter` オプションを使用して Migration Toolkit が実行され、エラーが発生します。

8.2.2 テーブルにデータをロード中にエラーが発生しました: **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にデータを受け取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ます。

8.2.3 制約 **CONS_NAME_FK** の作成中にエラーが発生しました

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
```

(次のページに続く)

(前のページからの続き)

```

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。

8.2.4 テーブルにデータをロード中にエラーが発生しました

テーブル定義はすでに移行しています。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
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。ターゲットテーブル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ます。

8.2.5 エラー：タイプに対して値が長すぎます

※テーブル定義は移行済みです。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。

ここで: COPY departments, line 1, column location_id: "1700"

問題を解決するには、列サイズを調整して、移行を再試行します。

8.2.6 ERROR：スレッドの例外：OutOfMemoryError

- MySQL データベースから移行するときに、次のエラーが発生します。*

```
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)
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 回のネットワークラウンドトリップでテーブル内のすべての行を 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。

問題を解決するには、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9章 Error codes

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。

各エラーコードは、プレフィックス MTK-で始まり、その後に 5 桁の数字が続きます。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。

この表は、エラークラスをリストしています。

Error Class	Description
00	正常に完了しました
01	インフォメーション
02	警告
03	一般的な恐怖
0x	予約済み
10	無効なユーザー入力
11	構成エラー
12	十分な権限がありません
13	サポートされていない機能
14	オブジェクトがありません
15	スキーマの移行
16	手続き型言語の移行
17	データ読み込み

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にはプレフィックス DB-が付きます。たとえば、Postgres データベースサーバー内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、データベースサーバーからエラーコード 42P07 が返されます。Migration Toolkit ログに特定のエラーは、DB-42P07 として表示されます。

9.1 エラーコードの概要

次の表の、**エラーコード** の列に、Migration Toolkit のエラーコードをリストします。**メッセージと解決策** 列には、エラーコードとともに表示されるメッセージが含まれています。メッセージには、エラーの原因と解決方法が説明されています。

メッセージと解決 列の **\$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。

9.1.1 クラス 02 - 警告

このクラスは、警告メッセージを表します。

Error code	Message and resolution
MTK-02000	All the warnings that relate to this class and don't have a specific error code binding use this error code.
MTK-02001	Run 'runMTK -help' to see the usage details.
MTK-02002	Warning! The offline migration path <i>\$OFFLINE_PATH</i> does not exist, the scripts will be created under the user home folder.

9.1.2 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。

Error Code	Message and Resolution
MTK-10000	All the errors that relate to this class and don't have a specific error code binding use this error code.
MTK-10001	You cannot select information_schema, dbo, sys, or pg_catalog as target schemas. These are used to store metadata information in <i>\$DATABASE</i> .
MTK-10002	The '-schemaOnly' and '-dataOnly' options cannot be specified at the same time.
MTK-10003	The "\t" is required as a copyDelimiter, when using escapeTabDelimiter option.
MTK-10004	The -truncLoad option can only be given with the -dataOnly option.
MTK-10005	The '-dataOnly' option is applicable only for -allTables/-tables option. Schema DDL cannot be copied when this option is in place.
MTK-10006	The -constraints, -indexes and -triggers options are applicable only in the context of -allTables/-tables option.
MTK-10007	The '-vacuumAnalyze' and '-analyze' options cannot be specified at the same time.
MTK-10008	The -skipFKConst option can only be given with -constraints option.
MTK-10009	The -skipCKConst option can only be given with -constraints option.
MTK-10010	The -fastCopy option cannot be specified with -schemaOnly option.
MTK-10011	The -skipColDefaultClause cannot be specified with -dataOnly option.
MTK-10012	The '-customColTypeMapping' and '-customColTypeMappingFile' options cannot be specified at the same time.
MTK-10013	Provided default date time must be in following format 'yyyy-MM-dd_HH:mm:ss'. Time portion is optional, to specify time, the underscore symbol '_' is necessary.
MTK-10014	You cannot select MySQL and Sybase as the target database for migration.
MTK-10015	Copy delimiter should be a single character.
MTK-10016	Batch size should be between 1 - 50000.
MTK-10017	Invalid number for batch size, use a number from 1-50000.
MTK-10018	Copy Batch Size should be greater than 0.
MTK-10019	Invalid number for Copy Batch Size, use a number > 0.

次のページに続く

表 3 – 前のページからの続き

Error Code	Message and Resolution
MTK-10020	Fetch Size should be greater than 0.
MTK-10021	Invalid number for Fetch Size, use a number > 0.
MTK-10022	One or more of the command-line arguments is invalid.
MTK-10023	The -targetDBVersion value should be specified using <i>major.*minor*</i> pattern.
MTK-10024	The -targetDBVersion can only be used with -offlineMigration mode.
MTK-10025	Options (-constraints I -indexes I -triggers I -tables I -views I -sequences I -procs I -funcs I -packages I -synonyms) cannot be used with multiple schemas option.
MTK-10026	The -allSchemas option should be specified as the last option in the command-line options list.
MTK-10027	You have specified invalid command-line arguments. Run 'runMTK -help' to see the usage details.
MTK-10028	The -replaceNullChar cannot be specified with -schemaOnly option.
MTK-10029	The -nullReplacementChar can only be specified with -replaceNullChar option.
MTK-10030	The -ignoreCheckConstFilter can only be given with -constraints option.
MTK-10031	The -enableConstBeforeDataLoad can only be given with -truncLoad option.
MTK-10032	The retryCount value should be greater than 0.
MTK-10033	Invalid number for retryCount, use a number > 0.
MTK-10034	The loaderCount value should be greater than 0.
MTK-10035	Invalid number for loaderCount, use a number > 0.
MTK-10036	Cannot use singleDataFile option for offline data migration in COPY format.
MTK-10037	The offline data migration in COPY format is supported only when PostgreSQL or EnterpriseDB (EDB Postgres Advanced Server) is the target database.
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	Log file size should be greater than 0.
MTK-10040	Log file count should be greater than or equal to 0.
MTK-10041	Offline migration can only be used with schema only option.
MTK-10042	The copyViaDBLinkOra option can only be used for copying data from Oracle to EnterpriseDB.
MTK-10043	The -recreateConst option can only be given with the -dataOnly option.
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	The -escapeKeywords value may be true or false.
MTK-10047	The -useOraCase option can only be used for migration from Oracle.
MTK-10048	The Postgres exported snapshot id is invalid.
MTK-10049	The URL specified for the Oracle database is not supported by dblink_oracle. Check the connectivity credentials and provide a valid URL.
MTK-10050	The schema <i>\$SCHEMA</i> not found on source database.
MTK-10051	The skipUserSchemaCreation option can only be used for user migration from Oracle to EnterpriseDB.

次のページに続く

表 3 – 前のページからの続き

Error Code	Message and Resolution
MTK-10052	The -truncLoad option cannot be used with offline data migration.
MTK-10053	The -parallelLoadRowLimit should be greater than 0.
MTK-10054	Invalid number for -parallelLoadRowLimit, use an integer in the range of 1 to \$MAX_LIMIT.
MTK-10055	The -parallelLoadRowLimit cannot be specified with -schemaOnly option.
MTK-10056	The -loaderCount cannot be specified with -schemaOnly option.
MTK-10057	The -tableLoaderLimit should be greater than 0.
MTK-10058	Invalid number for -tableLoaderLimit, use an integer in the range of 1 to \$MAX_LIMIT.
MTK-10059	The -tableLoaderLimit cannot be specified with -schemaOnly option.
MTK-10060	The parallel loading options (-loaderCount, -parallelLoadRowLimit, -tableLoaderLimit) are only supported for Oracle, EnterpriseDB and PostgreSQL source databases.
MTK-10061	The -fastCopy cannot be specified with table-level parallel loading context (-tableLoaderLimit > 1).
MTK-10062	The '-copyViaDBLinkOra' option cannot be used with offline data migration.
MTK-10063	The permutation of source database type \$DATABASE and target database type \$DATABASE is not supported.

9.1.3 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 によって使用されるその他の構成ファイル内の無効な構成設定を表します。

Error code	Message and resolution
MTK-11000	All the errors that relate to this class and don't have a specific error code binding use this error code.
MTK-11001	The properties file containing table filter clause cannot be loaded.
MTK-11002	The custom type mapping file couldn't be loaded. Reason: \$REASON.
MTK-11003	The custom type mapping file contains an invalid mapping entry.
MTK-11004	The custom type mapping file doesn't contain any mapping entry.
MTK-11005	Connectivity information for source database is not available.
MTK-11006	Connectivity information for target database is not available.
MTK-11007	Unable to create the logs directory. \$PATH.
MTK-11008	The properties file containing table columns filter list cannot be loaded.
MTK-11009	Error Connecting Database \$DATABASE.
MTK-11010	Error loading \$DBLINKORA_FILE file.
MTK-11011	Error while loading DBLink Ora module. \$DBLINKORA_MODULE. Verify that dblink_ora is installed/configured on target EnterpriseDB server. Please see the Database Compatibility for Oracle Developer's Guide for more information about installing and configuring the dblink_ora module.

次のページに続く

表 4 – 前のページからの続き

Error code	Message and resolution
MTK-11012	Error while loading given DBLink_Ora module. <i>\$DBLINKORA_MODULE</i> . Verify that dblink_ora is installed/configured on target EnterpriseDB server. Please see the Database Compatibility for Oracle Developer's Guide for more information about installing and configuring the dblink_ora module.
MTK-11013	Could not load DBLinkOra Module.
MTK-11014	Error connecting to DBLinkOra.
MTK-11015	The connection credentials file <i>\$TOOLKIT_PROP_FILE</i> is not secure and accessible to group/others users. This file contains plain passwords and should be restricted to Migration Toolkit owner user only.
MTK-11016	Data loader task failed to initialize. Reason: <i>\$REASON</i>

9.1.4 クラス 12 - 不十分な特権

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

Error code	Message and resolution
MTK-12000	All the errors that relate to this class and don't have a specific error code binding use this error code.
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, DBA_SYS_PRIVS.
MTK-12002	The migration of privileges failed due to insufficient privileges. Grant the user SELECT privilege on the following Oracle catalog: dba_tab_privs.

9.1.5 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と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を実装していないか、Migration Toolkit がその移行を処理していません。

Error code	Message and resolution
MTK-13000	All the errors that relate to this class and don't have a specific error code binding use this error code.
MTK-13001	Skipping index as Oracle does not allow multiple indexes against same column list.
MTK-13002	For <i>\$DATABASE</i> views migration is not supported.
MTK-13003	For <i>\$DATABASE</i> roles migration is not supported.

次のページに続く

表 6 – 前のページからの続き

Error code	Message and resolution
MTK-13004	You cannot migrate triggers, sequences, procedures, functions, packages and synonyms for MySQL, SQL Server, and Sybase databases.
MTK-13005	You cannot migrate sequences and packages for SQL Server database.
MTK-13006	Given trigger is not migrated, the trigger is owned by a different user schema and cross-schema triggers are not supported by EnterpriseDB.
MTK-13007	The given trigger is not migrated, the trigger has WHEN clause which is not supported by EnterpriseDB.
MTK-13008	Skipping Database Link <i>\$DATABASE_LINK</i> . EnterpriseDB currently does not support this type of Database Link.
MTK-13009	Warning! Skipping migration of trigger <i>\$TRIGGER</i> , currently non-table triggers are not supported in target database.
MTK-13010	You cannot migrate procedures, packages, synonyms and database links to PostgreSQL database.
MTK-13011	Domain objects are not supported in target database.
MTK-13012	Rules are not supported in <i>\$DATABASE</i> database.
MTK-13013	The database type is not supported.
MTK-13015	<i>\$TYPE</i> is Not Supported by COPY.
MTK-13016	The migration to PostgreSQL is supported only when Oracle or PostgreSQL is the source database.
MTK-13017	Groups are not supported in <i>\$DATABASE</i> database.
MTK-13018	Profile migration is not supported in <i>\$SRC_DB</i> to <i>\$TARGET_DB</i> permutation.
MTK-13019	Cannot migrate unknown Profile <i>\$PROFILE</i> .
MTK-13020	Profiles cannot be migrated to database version <i>\$VERSION</i> .
MTK-13021	Password Profile verify function <i>\$MY_VERIFICATION_FUNCTION</i> must be explicitly migrated.
MTK-13022	Advanced Queues migration is not supported from <i>\$SRC_DB</i> database to <i>\$TARGET_DB</i> database.
MTK-13023	Advanced Queues cannot be migrated to database version <i>\$VERSION</i> .
MTK-13024	The INTERVAL partition is not supported in <i>\$DATABASE</i> , the table will be migrated without INTERVAL definition.
MTK-13025	Warning! User profile migration is not supported in target database version <i>\$VERSION</i> . The profile " <i>\$PROFILE</i> " for user " <i>\$USER</i> " will be skipped.
MTK-13026	Object Type migration is not supported from <i>\$SRC_DB</i> to <i>\$TARGET_DB</i> .
MTK-13027	Migration is not supported from <i>\$DATABASE</i> database for PDB Admin schemas/users with PDB_DBA role.

9.1.6 クラス 14 - 見つからないオブジェクト

このクラスは、ソースデータベースでメタデータ情報を見つけることができないことを表します。

Error code	Message and resolution
MTK-14000	All the errors that relate to this class and don't have a specific error code binding use this error code.
MTK-14001	One or more tables are missing from the source <i>\$DATABASE</i> database.
MTK-14002	One or more tables couldn't be found in the source <i>\$DATABASE</i> database. With -tables mode, the table name should be in uppercase unless it is case-sensitive.
MTK-14003	One or more users couldn't be found in the source <i>\$DATABASE</i> database. With -users mode, the user name should be in uppercase unless it is case-sensitive.

9.1.7 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、シノニム、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。

Error code	Message and resolution
MTK-15000	All the errors that relate to this class and don't have a specific error code binding use this error code.
MTK-15001	Error creating constraint <i>\$CONSTRAINT</i> .
MTK-15002	Error creating role <i>\$ROLE</i> .
MTK-15003	Error granting given privilege <i>\$PRIVILEGE</i> on <i>\$OBJECT</i> to <i>\$USER</i> .
MTK-15004	Error granting <i>\$ROLE</i> to <i>\$USER</i> .
MTK-15005	Error granting privilege <i>\$REASON</i> .
MTK-15006	Error creating user <i>\$USER</i> .
MTK-15007	Error creating index <i>\$INDEX</i> .
MTK-15008	Error creating view <i>\$VIEW</i> .
MTK-15009	Error creating materialized view <i>\$MVIEW</i> .
MTK-15010	Error creating public synonym <i>\$SYNONYM</i> .
MTK-15011	Error creating private synonym <i>\$SYNONYM</i> .
MTK-15012	Error creating sequence <i>\$SEQUENCE</i> .
MTK-15013	Error creating table <i>\$TABLE</i> .
MTK-15014	Error creating database link <i>\$DATABASE_LINK</i> .
MTK-15015	Error creating given object type <i>\$OBJECT_TYPE</i> .
MTK-15016	The table <i>\$TABLE</i> could not be created in <i>\$DATABASE</i> database.
MTK-15017	The linked schema <i>\$LINKED_SCHEMA</i> doesn't exist in the target database. Create the schema and then retry.
MTK-15018	Error creating domain <i>\$DOMAIN</i> .
MTK-15019	Error creating custom data type <i>\$DATA_TYPE</i> .
MTK-15020	Error creating membership for group <i>\$GROUP</i> .

次のページに続く

表 8 – 前のページからの続き

Error code	Message and resolution
MTK-15021	Table name <i>\$TABLE</i> does not have a schema qualifier. With multiple schema migration context, each table should be schema qualified.
MTK-15022	Schema qualifier <i>\$SCHEMA</i> does not match the schema list.
MTK-15023	The table metadata information is not available.
MTK-15024	Tables list is not initialized yet.
MTK-15025	Error while getting database metadata information.
MTK-15026	Error creating group <i>\$GROUP</i> .
MTK-15027	Error creating Profile <i>\$PROFILE</i> .

9.1.8 クラス 16 - 手続き型言語の移行

このクラスは、プロシージャ、ファンクション、パッケージ、匿名ブロック、トリガー、ルールなどの手続き型言語に基づいた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。

Error code	Message and resolution
MTK-16000	All the errors that relate to this class and don't have a specific error code binding use this error code.
MTK-16001	Error Creating Trigger <i>\$TRIGGER</i> .
MTK-16002	Error Creating Rule <i>\$RULE</i> .
MTK-16003	Error Creating Package Spec <i>\$PACKAGE</i> .
MTK-16004	Error Creating Package Body <i>\$PACKAGE</i> .
MTK-16005	Package Body is invalid, skipping... Note: This error message also appears when a package specification is successfully migrated, but there is no corresponding package body in the source database. In this case, the package specification should function properly in the target database despite the appearance of the error message.
MTK-16006	Error Creating Procedure <i>\$PROCEDURE</i> .
MTK-16007	Error Creating Function <i>\$FUNCTION</i> .
MTK-16008	Error Creating Anonymous Block <i>\$BLOCK</i> .
MTK-16009	Error creating Queue <i>\$QUEUE</i> .

9.1.9 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中に発生する可能性のあるエラーを表します。

Error code	Message and resolution
MTK-17000	All the errors that relate to this class and don't have a specific error code binding use this error code.
MTK-17001	Error Loading Data into Table: <i>\$TABLE</i>
MTK-17002	Encoding Conversion encountered some characters that will be loaded using Bulk Inserts instead.
MTK-17003	Error in copy tables <i>\$REASON</i> .
MTK-17004	Invalid DataType.
MTK-17005	This Table Contains CLOB data, Marked for Bulk Insert Loading.
MTK-17006	One of the table loader threads failed while copying data for table <i>\$TABLE</i> .

第10章 FAQ

MigrationToolkit 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 EDB Postgres Advanced Server へのパッケージの移行をサポートします。EDB Postgres Advanced Server が提供する移行サポートについては、

Functionality overview を参照してください。

ソースデータベースから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移行の一部としてサポートされています。

MigrationToolkit 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せん。

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

EDB Postgres Advanced Server は現在 enum データ型をサポートしていませんが、将来のリリースでサポートする予定です。それまでは、チェック制約を使用して、EDB Postgres 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（support）に含まれていない値を挿入しようとすると、EDB Postgres Advanced Server はエラーを返します。

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

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

EDBPostgresAdvancedServer 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設定するには、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。自動クエリリライトは現在サポートされていません。アプリケーションにサマリーテーブルの存在を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と、次のエラーが表示されます。ErrorLoadingDataintoTable:BadformatforTime。Postgres は MySQL ``TIME`` データ型をサポートしていますか？

Postgres は、時間コンポーネントの値が 24 以下である限り、TIME データ型の保存に問題はありません。

Postgres とは異なり、MySQL TIME データ型では、TIME または INTERVAL 値を表す値を格納できます。INTERVAL 値を表す MySQL TIME カラムに格納されている値は、有効な Postgres TIMESTAMP 値の許容範囲外であ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ている値を範囲外と認識した場合、エラーを返します。