## Contents

1 What’s New 2

2 Supported Operating Systems and Database Versions 3

3 Migration Methodology 5
   3.1 The Migration Process 5
   3.2 Connecting an Application to Postgres 7

4 Functionality Overview 9

5 Installing Migration Toolkit 12
   5.1 Installing Migration Toolkit on an OEL, CentOS, or RHEL 7 or 8 host 13
   5.1.1 Upgrading an RPM Installation 16
   5.2 Installing Migration Toolkit on a CentOS or RHEL 7 ppc64le host 17
   5.3 Installing Migration Toolkit on a Debian or Ubuntu Host 19
   5.4 Installing Migration Toolkit on a SLES Host 21
   5.5 Installing Migration Toolkit on Windows 23
   5.6 Installing Migration Toolkit on Mac OS X 34
   5.7 Installing Source-Specific Drivers 39

6 Building the toolkit.properties File 40
   6.1 Defining an EDB Postgres Advanced Server URL 42
   6.2 Defining a PostgreSQL URL 44
   6.3 Defining an Oracle URL 46
   6.4 Defining a MySQL URL 48
   6.5 Defining a Sybase URL 49
   6.6 Defining a SQL Server URL 50

7 Invoking Migration Toolkit 52
   7.1 Migrating a Schema from Oracle 54
Migration Toolkit is a powerful command-line tool that offers granular control of the migration process. Migration Toolkit facilitates migration of database objects and data to an EDB Postgres™ Advanced Server or PostgreSQL® database from:

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

You can also use Migration Toolkit to migrate database objects and data from Sybase® Adaptive Server® Enterprise to EDB Postgres Advanced Server, or between EDB Postgres Advanced Server and PostgreSQL.

You can install Migration Toolkit with the EDB Postgres Advanced Server installer, or via Stack Builder. Stack Builder is distributed with both the EDB Postgres Advanced Server and the PostgreSQL one-click installer available from the EnterpriseDB web site at:

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

This guide provides a high-level description of the steps involved in the migration process, as well as installation and usage instructions for Migration Toolkit. It also includes solutions to common migration problems, and details unsupported features and their potential workarounds.

Note:

This guide uses the term Postgres to refer to either an installation of EDB Postgres Advanced Server or PostgreSQL.

This guide uses the term Stack Builder to refer to either StackBuilder Plus (distributed with EDB Postgres Advanced Server) or Stack Builder (distributed with the PostgreSQL one-click installer from EnterpriseDB).

In addition, if you are using the pdf version of this document, using cut/paste to copy command may result in extra spaces or carriage returns in the pasted command. If a command fails, check the command carefully for additional characters.
Chapter 1

What’s New

The following enhancements are added to the EDB Postgres Migration Toolkit for 54.0.0 release:

- Migration Toolkit now supports:
  - Postgres and EDB Postgres Advanced Server version 13
  - Ubuntu version 20
  - Windows (64-bit)

**Note:** If you have an existing 32-bit Windows Migration Toolkit installation, you must uninstall and reinstall Migration Toolkit with the Windows 64-bit installer.

For complete list of supported platforms, see *Supported Operating Systems and Database Versions*. 
CHAPTER 2

Supported Operating Systems and Database Versions

Database Versions

The following database product versions can be used with Migration Toolkit:

- PostgreSQL versions 9.6, 10, 11, 12, and 13
- EDB Postgres Advanced Server versions 9.6, 10, 11, 12, and 13
- Oracle 10g Release 2
- Oracle 11g Release 2
- Oracle 12c Release 1
- SQL Server 2008
- SQL Server 2012
- SQL Server 2014
- MySQL 5.5.36
- Sybase Adaptive Server Enterprise 15.7

Please contact your EnterpriseDB Account Manager or sales@enterprisedb.com if you require support for other database products.

Note: The Migration Toolkit has not been tested and does not officially support use with Oracle Real Application Clusters (RAC) and Exadata. However, Migration Toolkit may work when it is connected to a single persistent node. For more information, contact your EDB representative.

Supported Operating Systems Versions
- Centos or RHEL or OEL 7 and 8
- CentOS or RHEL PPCLE 7
- SLES 12 SP5
- Debian 9 and 10
- Ubuntu 18.04 and 20
- Windows (64-bit), 2016, and 2019
- Mac OS X 10.12+

**Note:** The ojdbc7.jar can cause large data migrations to be incomplete due to a limit on records over `Integer.MAX_VALUE` (2147483647) while fetching from the ResultSet. To avoid this issue, we recommend upgrading to ojdbc8.jar.
There are many reasons to consider migrating from one database to another. Migration can allow you to take advantage of new or better technology. If your current database does not offer the right set of capabilities to allow you to scale the system, moving to a database that offers the functionality you need is the best move for your company.

Migration can also be very cost effective. Migrating systems with significant maintenance costs can save money spent on system upkeep. By consolidating the number of databases in use, you can also reduce in-house administrative costs. By using fewer database platforms (or possibly taking advantage of database compatibility), you can do more with your IT budget.

Using more than one database platform can offer you a graceful migration path should a vendor raise their pricing or change their company directive. EnterpriseDB has helped companies migrate their existing database systems to Postgres for years.

We recommend following the methodology detailed in *The Migration Process*.

### 3.1 The Migration Process

The migration path to Postgres includes the following main steps:

1. Start the migration process by determining which database objects and data will be included in the migration. Form a migration team that includes someone with solid knowledge of the architecture and implementation of the source system.

2. Identify potential migration problems. If it is an Oracle-to-EDB Postgres Advanced Server migration, consult the EnterpriseDB documentation for complete details about the compati-
bility features supported in EDB Postgres Advanced Server. Consider using EnterpriseDB’s migration assessment service to assist in this review.

3. Prepare the migration environment. Obtain and install the necessary software, and establish connectivity between the servers.

4. If the migration involves a large body of data, consider migrating the schema definition before moving the data. Verify the results of the DDL migration and resolve any problems reported in the migration summary. The Migration Errors section of this document includes information about resolving migration problems.

5. Migrate the data. For small data sets, use Migration Toolkit. If it is an Oracle migration (into EDB Postgres Advanced Server), and the data set is large or if you notice slow data transfer, take advantage of one of the other data movement methods available:
   - Use the EDB Postgres Advanced Server database link feature compatible with Oracle databases.
   - If your data has BLOB or CLOB data, use the dblink_ora style database links instead of the Oracle style database links.

   Both of these methods use the Oracle Call Interface (OCI) to connect to Oracle. After connecting, use an SQL statement to select the data from the ‘linked’ Oracle database and insert the data into the EDB Postgres Advanced Server database.

6. Confirm the results of the data migration and resolve any problems reported in the migration summary.

7. Convert applications to work with the newly migrated Postgres database. Applications that use open standard connectivity such as JDBC or ODBC normally only require changes to the database connection strings and selection of the EnterpriseDB driver. See Connecting an Application to Postgres for more information.

8. Test the system performance, and tune the new server. If you are migrating into an EDB Postgres Advanced Server database, take advantage of EDB Postgres Advanced Server’s performance tuning utilities:
   - Use Dynatune to dynamically adjust database configuration resources.
   - Use Optimizer Hints to direct the query path.
   - Use the ANALYZE command to retrieve database statistics.

The EDB Postgres Advanced Server Guide and Database Compatibility for Oracle Developer’s Guide (both available through the EnterpriseDB website) offer information about the performance tuning tools available with EDB Postgres Advanced Server.
3.2 Connecting an Application to Postgres

To convert a client application to use a Postgres database, you must modify the connection properties to specify the new target database. In the case of a Java application, change the JDBC driver name (`Class.forName`) and JDBC URL.

A Java application running on Oracle might have the following connection properties:

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

Modify the connection string to connect to a Postgres server:

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

Converting an ODBC application to connect to an instance of Postgres is a two-step process.

i. To connect an ODBC application, use an ODBC data source administrator to create a data source that defines the connection properties for the new target database.

   Most Linux and Windows systems include graphical tools that allow you to create and edit ODBC data sources. After installing ODBC, check the Administrative Tools menu for a link to the ODBC Data Source Administrator. Click the Add button to start the Create New Data Source wizard; complete the dialogs to define the new target data source.

ii. Change the application to use the new data source.

The application will contain a call to `SQLConnect` (or possibly `SQLDriverConnect`); edit the invocation to change the data source name. In the following example, the data source is named `OracleDSN`:

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

To connect to an instance of Postgres defined in a data source named `PostgresDSN`, change the data source name:
After establishing a connection between the application and the server, test the application to find any compatibility problems between the application and the migrated schema. In most cases, a simple change will resolve any incompatibility that the application encounters. In cases where a feature is not supported, use a workaround or third party tool to provide the functionality required by the application. See *Migration Errors*, for information about some common problems and their workarounds.
Migration Toolkit is a powerful command-line tool that offers granular control of the migration process. Using Migration Toolkit is a two-step process:

1. Edit the `toolkit.properties` file to specify the source and target database.
2. Invoke Migration Toolkit at the command line, specifying migration options.

Migration Toolkit facilitates migration of database objects and data to an EDB Postgres Advanced Server or PostgreSQL database from:

- Oracle
- MySQL
- SQL Server

Migration Toolkit also allows you to migrate database objects and data to an EDB Postgres Advanced Server database from Sybase. You can also use Migration Toolkit to migrate between EDB Postgres Advanced Server and PostgreSQL. Migration Toolkit includes a number of options, allowing you granular control of the migration process:

- Use the `–safeMode` option to commit each row as it is migrated.
- Use the `–fastCopy` option to bypass WAL logging to optimize migration.
- Use the `–batchSize` option to control the batch size of bulk inserts.
- Use the `–cpBatchSize` option to specify the batch size used with the COPY command.
- Use the `–lobBatchSize` option to specify the batch size used for large object data types.
• Use the `-filterProp` option to migrate only those rows that meet a user-defined condition.

• Use the `-customColTypeMapping` option to change the data type of selected columns.

• Use the `-dropSchema` option to drop the existing schema and create a new schema prior to migration.

• On EDB Postgres Advanced Server, use the `-allDBLinks` option to migrate all Oracle database links.

• On EDB Postgres Advanced Server, use the `-copyViaDBLinkOra` option to enable the `dblink_ora` module.

**Object Migration Support**

Migration Toolkit migrates object definitions (DDL), table data, or both. The following table contains a platform-specific list of the types of database objects that Migration Toolkit can migrate:

<table>
<thead>
<tr>
<th>Object</th>
<th>Oracle</th>
<th>Sybase</th>
<th>SQL Server</th>
<th>MySQL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Schemas</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Tables</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>List-Partitioned Table</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Range-Partitioned Table</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hash Partitioned Table</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Constraints</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Indexes</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Triggers</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Table Data</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Views</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Materialized Views</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Packages</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Procedures</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Functions</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sequences</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Users/Roles</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Profiles</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Object Types</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Object Type Methods</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Database Links</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Queues</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

For detailed information about the commands that offer granular control of the objects imported, please see *Schema Object Selection Options*.

**Online Migration vs. Offline Migration**
Migration Toolkit can migrate immediately and directly into a Postgres database (online migration), or you can also choose to generate scripts to use at a later time to recreate object definitions in a Postgres database (offline migration).

By default, Migration Toolkit creates objects directly into a Postgres database; in contrast, include the -offlineMigration option to generate SQL scripts you can use at a later time to reproduce the migrated objects or data in a new database. You can alter migrated objects by customizing the migration scripts generated by Migration Toolkit before you execute them. With the -offlineMigration option, you can schedule the actual migration at a time that best suits your system load.

For more information about the -offlineMigration option, see Offline Migration Options.
Before installing Migration Toolkit, you must install Java (version 1.7.0 or later). To install Migration Toolkit, you must have credentials that allow access to the EnterpriseDB repository. To request credentials for the repository, visit the EnterpriseDB website at:

https://www.enterprisedb.com/repository-access-request
5.1 Installing Migration Toolkit on an OEL, CentOS, or RHEL 7 or 8 host

You can use an RPM package to install Migration Toolkit on a CentOS or RHEL host. The following steps provide detailed information about accessing the EnterpriseDB repository and installing Migration Toolkit.

Creating a Repository Configuration File

1. To create the repository configuration file, assume superuser privileges and invoke the following command:
   - On RHEL or CentOs 7:
     ```
     yum -y install https://yum.enterprisedb.com/edb-repo-rpms/
     edb-repo-latest.noarch.rpm
     ```
   - On RHEL or CentOs 8:
     ```
     dnf -y install https://yum.enterprisedb.com/edb-repo-rpms/
     edb-repo-latest.noarch.rpm
     ```

   The repository configuration file is named `edb.repo`. The file resides in `/etc/yum.repos.d`.

2. After creating the `edb.repo` file, use your choice of editor to ensure that the value of the `enabled` parameter is 1, and replace the `username` and `password` placeholders in the `baseurl` specification with the name and password of a registered EnterpriseDB user.

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

3. After saving your changes to the configuration file, install the EPEL repository:
   - On RHEL or CentOS 7:
     ```
     yum -y install https://dl.fedoraproject.org/pub/epel/
     epel-release-latest-7.noarch.rpm
     ```
   - On RHEL or Centos 8:

4. If you are using RHEL or CentOS 8.x, you must enable an additional repository to resolve dependencies:
   • On CentOS 8 (Powertools repo):
     
     dnf config-manager --set-enabled PowerTools
   
     • On RHEL 8 (codeready-builder-for-rhel-8-*-rpms repo):
     
     ARCH=$( /bin/arch )
     subscription-manager repos --enable "codeready-builder-for-rhel-8-${ARCH}-rpms"

Installing Migration Toolkit

• On RHEL or CentOs 7:
  
  yum -y install edb-migrationtoolkit

• On RHEL or CentOs 8:
  
  dnf -y install edb-migrationtoolkit

When you install an RPM package that is signed by a source that is not recognized by your system, yum may ask for your permission to import the key to your local server. If prompted, and you are satisfied that the packages come from a trustworthy source, enter a y, and press Return to continue.

During the installation, yum may encounter a dependency that it cannot resolve. If it does, it will provide a list of the required dependencies that you must manually resolve.

After installing Migration Toolkit, you must configure the installation. Perform the following steps before invoking Migration Toolkit.

Using Migration Toolkit with IDENT Authentication

By default, the pg_hba.conf file for the RPM installer enforces IDENT authentication for remote clients. Before invoking Migration Toolkit, you must either modify the pg_hba.conf file, changing the authentication method to a form other than IDENT (and restarting the server), or perform the following steps to ensure that an IDENT server is accessible:

1. Confirm that an identd server is installed and running. You can use the yum package manager to install an identd server by invoking the command:

   yum install xinetd authd

   The command should create a file named /etc/xinetd.d/auth that contains:
service auth
{
    disable = yes
    socket_type = stream
    wait = no
    user = ident
    cps = 4096 10
    instances = UNLIMITED
    server = /usr/sbin/in.authd
    server_args = -t60 --xerror --os
}

Note: if the file includes a -E argument at the end of server_args, please erase the -E.

Then, to start the identd server, invoke the commands:

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

2. Open the pg_ident.conf file and create a user mapping:

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

Where:

- The name specified in the map_name column is a user-defined name that will identify the mapping in the pg_hba.conf file.
- The name specified in the system_username column is enterprisedb.
- The name specified in the postgres_username column is enterprisedb.

3. Open the pg_hba.conf file and modify the IDENT entries.
   On an IPv4 host:
   ```
   host all all 127.0.0.0/0 ident map=edbas
   ```
   On an IPv6 host:
   ```
   host all all ::1/128 ident map=edbas
   ```

4. Restart the EDB Postgres Advanced Server service before invoking Migration Toolkit.
5.1.1 Upgrading an RPM Installation

If you have an existing RPM installation, you can use `yum` to upgrade your repository configuration file and update to a more recent product version. To update the `edb.repo` file, assume superuser privileges and enter:

```
yum upgrade edb-repo
```

`yum` will update the `edb.repo` file to enable access to the current EnterpriseDB repository, configured to connect with the credentials specified in your `edb.repo` file. Then, you can use `yum` to upgrade Migration toolkit:

```
yum upgrade edb-migrationtoolkit
```
5.2 Installing Migration Toolkit on a CentOS or RHEL 7 ppc64le host

You can use an RPM package to install Migration Toolkit on a CentOS or RHEL ppc64le host.

Before installing Migration toolkit, you must:

1. Create the Advance Toolchain repository configuration file; assume superuser privileges and invoke the following command:

   ```
   ```

2. After creating the `advance-toolchain.repo` file, use your choice of editor to ensure that the value of the `enabled` parameter is 1, and replace the username and password placeholders in the `baseurl` specification with the name and password of a registered EnterpriseDB user.

   ```
   [advance-toolchain]
   name=Advance Toolchain IBM FTP
   baseurl=https://public.dhe.ibm.com/software/server/POWER/Linux/toolchain/at/redhat/RHEL7
   failovermethod=priority
   enabled=1
   gpgcheck=1
   ```

The following steps provide detailed information about accessing the EnterpriseDB repository and installing Migration Toolkit.

**Creating a Repository Configuration File**

1. To create the EDB repository configuration file, assume superuser privileges and invoke the following command:

   ```
   ```

   The repository configuration file is named `edb.repo`. The file resides in `/etc/yum.repos.d`.

2. After creating the `edb.repo` file, use your choice of editor to ensure that the value of the `enabled` parameter is 1, and replace the username and password placeholders in the `baseurl` specification with the name and password of a registered EnterpriseDB user.
3. After saving your changes to the configuration file, install the EPEL package:

```
```

4. If you are using RHEL 7 ppc64le, you must enable an additional repository to resolve dependencies:

```
subscription-manager repos --enable "rhel-**-optional-rpms" --enable "rhel-**-extras-rpms" --enable "rhel-ha-for-rhel-* -server-rpms"
```

5. To install Migration Toolkit, run the following command.

```
yum -y install edb-migrationtoolkit
```
5.3 Installing Migration Toolkit on a Debian or Ubuntu Host

To install a package with apt, complete the following steps. When invoking the commands, replace the username and password with the user credentials provided by EnterpriseDB.

1. Assume superuser privileges:

   ```
   sudo su -
   ```

2. Configure the EnterpriseDB repository:

   a. On Debian 9, perform the following step to configure the EnterpriseDB repository:

   ```
   sh -c 'echo "deb https://username:password@apt.enterprisedb.com/$(lsb_release -cs)-edb/ $(lsb_release -cs) main" > /etc/apt/sources.list.d/edb-$({lsb_release -cs}).list'
   ```

   b. On Debian 10, perform the following steps to configure the EnterpriseDB repository:

      a. Set up the EnterpriseDB repository:

      ```
      sh -c 'echo "deb [arch=amd64] https://apt.enterprisedb.com/$(lsb_release -cs)-edb/ $(lsb_release -cs) main" > /etc/apt/sources.list.d/edb-$({lsb_release -cs}).list'
      ```

      b. Substitute your EnterpriseDB credentials for the username and password placeholders in the following command:

      ```
      sh -c 'echo "machine apt.enterprisedb.com login <username> password <password>" > /etc/apt/auth.conf.d/edb.conf'
      ```

3. Add support to your system for secure APT repositories:

   ```
   apt-get install apt-transport-https
   ```

4. Add the EBD signing key:

   ```
   wget -q -O - https://username:password@apt.enterprisedb.com/edb-deb.gpg.key | apt-key add -
   ```

5. Update the repository metadata:
6. Install the `edb-migrationtoolkit` package:

```
apt-get -y install edb-migrationtoolkit
```
5.4 Installing Migration Toolkit on a SLES Host

You can use the zypper package manager to install a Migration toolkit on an SLES 12 host. zypper will attempt to satisfy package dependencies as it installs a package, but requires access to specific repositories that are not hosted at EnterpriseDB.

1. You must assume superuser privileges and stop any firewalls before installing Migration Toolkit. Then, use the following commands to add EnterpriseDB repositories to your system:

```bash
zypper addrepo https://zypp.enterprisedb.com/suse/edb-sles.repo
```

2. The command creates the repository configuration files in the /etc/zypp/repos.d directory. Use the following command to refresh the metadata on your SLES host to include the EnterpriseDB repository:

```bash
zypper refresh
```

When prompted, provide credentials for the repository; specify a to always trust the provided key and update the metadata to include the EnterpriseDB repository.

3. You must also add SUSEConnect and the SUSE Package Hub extension to the SLES host, and register the host with SUSE, allowing access to SUSE repositories. Use the commands:

```bash
SUSEConnect -r 'REGISTRATION_CODE' -e 'EMAIL'
SUSEConnect -p PackageHub/12.4/x86_64
SUSEConnect -p sle-sdk/12.4/x86_64
```

4. Install SUSEConnect to register the host with SUSE, allowing access to SUSE repositories:

```bash
zypper addrepo https://download.opensuse.org/repositories/Apache:/Modules/SLE_12_SP4/Apache:Modules.repo
```

5. Use the following command to refresh the metadata:

```bash
zypper refresh
```

6. Install OpenJDK (version 1.8) for Java based components:

```bash
zypper -n install java-1_8_0-openjdk
```

7. Now you can use the zypper utility to install a Migration Toolkit:

```bash
zypper -n install edb-migrationtoolkit
```

For detailed information about registering a SUSE host, visit:
5.4. Installing Migration Toolkit on a SLES Host

https://www.suse.com/support/kb/doc/?id=7016626
5.5 Installing Migration Toolkit on Windows

You can use the graphical installer to install the Migration Toolkit on Windows. You can download the installer from www.enterprisedb.com.

Once you download the installer, right-click the installer icon, and select Run As Administrator from the context menu. When the Language Selection popup opens, select an installation language and click OK to continue to the Setup window.

Note: This guide uses the term Stack Builder to refer to either StackBuilder Plus (distributed with EDB Postgres Advanced Server) or Stack Builder (distributed with the PostgreSQL one-click installer from EnterpriseDB).

You must have a Java JVM (version 1.7.0 or later) in place before Stack Builder can perform a Migration Toolkit installation.

The Java executable must be in your search path (%PATH% on Windows) Use the following commands to set the search path (substituting the name of the directory that holds the Java executable for javadir):

On Windows, use the command: SET PATH=javadir;%PATH%

After setting the search path, you can use the Stack Builder installation wizard to install Migration Toolkit into either EDB Postgres Advanced Server or PostgreSQL.

The Java executable must be in your search path (%PATH% on Windows, $PATH on Linux/Unix). Use the following commands to set the search path (substituting the name of the directory that holds the Java executable for javadir):

On Windows, use the command:

    SET PATH=javadir;%PATH%

On Linux, use the command:

    PATH=javadir:$PATH

After setting the search path, you can use the Stack Builder installation wizard to install Migration Toolkit into either EDB Postgres Advanced Server or PostgreSQL.

Launching StackBuilder Plus from EDB Postgres Advanced Server

To launch StackBuilder Plus from an existing EDB Postgres Advanced Server installation, navigate through the Start (or Applications) menu to the EDB Postgres menu; open the EDB Add-ons menu, and select the StackBuilder Plus menu option.

Launching Stack Builder from PostgreSQL

To launch Stack Builder from a PostgreSQL installation, navigate through the Start (or Applications) menu to the PostgreSQL menu, and select the Application
StackBuilder Plus menu option.

Stack Builder opens as shown below.

![StackBuilder Plus welcome window](image)

Fig. 1: The Stack Builder welcome window.

Use the drop-down listbox to select the target server installation from the list of available servers. If your network requires you to use a proxy server to access the Internet, use the Proxy servers button to open the Proxy servers dialog and specify a server; if you do not need to use a proxy server, click Next to open the application selection window.

If you are using StackBuilder Plus to add Migration Toolkit to your EDB Postgres Advanced Server installation, expand the Add-ons, tools and utilities node of the tree control, and check the box next to EnterpriseDB Migration Toolkit. Click Next to continue.

Note: If you are using Stack Builder to add Migration Toolkit to your PostgreSQL installation, expand the EnterpriseDB Tools node of the tree control (located under the Registration-required and trial products node), and select Migration Toolkit checkbox. Click Next to continue.
Confirm that Migration Toolkit is included in the Selected Packages list and that the Download directory field contains an acceptable download location.
Fig. 3: The Stack Builder selection confirmation window.

Click Next to start the Migration Toolkit download.
When the download completes, Stack Builder confirms that the installation files have been successfully downloaded. Choose Next to open the Migration Toolkit installation wizard.

When prompted by the Migration Toolkit installation wizard, specify a language for the installation and click OK to continue.

The Migration Toolkit installation wizard opens. Click Next to continue.
Carefully review the license agreement before highlighting the appropriate radio button; click Next to continue.
By default, Migration Toolkit will be installed in the `mtk` directory; accept the default installation directory as displayed, or modify the directory, and click Next to continue.
The installation wizard confirms that the Setup program is ready to install Migration Toolkit; click Next to start the installation.
A dialog confirms that the Migration Toolkit installation is complete; click Finish to exit the Migration Toolkit installer.
When Stack Builder finalizes installation of the last selected component, it displays the Installation Completed window. Click Finish to close Stack Builder.
Fig. 10: Stack Builder confirms the installation is complete.

After installing Migration Toolkit with Stack Builder, you must install the appropriate source-specific drivers before performing a migration; please see Installing Source-Specific Drivers for more information.
5.6 Installing Migration Toolkit on Mac OS X

You can use the graphical installer to install the Migration Toolkit on Mac OS X. To download the installer, visit www.enterprisedb.com.

To launch the installer:

- Navigate to the file path in the terminal:
  
  `/<User/Directory>/Downloads/edb-migrationtoolkit-54.0.0-1-osx.app/Contents/MacOS`

- Assume superuser privileges and invoke the following command:

  `.installbuilder.sh`

1. On the Language Selection popup opens, select an installation language and click OK to continue to the Setup window.

2. The Migration Toolkit installation wizard opens. Click Next to continue.

3. Carefully review the license agreement before highlighting the appropriate radio button; click Next to continue.

Fig. 11: The Migration Toolkit installation wizard.
By default, Migration Toolkit will be installed in the mtk directory; accept the default installation directory as displayed, or modify the directory, and click Next to continue.
Fig. 13: *Specify a Migration Toolkit installation directory.*

The installation wizard confirms that the Setup program is ready to install Migration Toolkit; click Next to start the installation.
A dialog confirms that the Migration Toolkit installation is complete; click Finish to exit the Migration Toolkit installer.
Fig. 15: *The Migration Toolkit wizard has finished.*
5.7 Installing Source-Specific Drivers

Before invoking Migration Toolkit, you must download and install a freely available source-specific driver. To download a driver, or for a link to a vendor download site, visit the Third Party JDBC Drivers section of the Advanced Downloads page at the EnterpriseDB website:

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

After downloading the source-specific driver, move the driver file into the <mtk_install_dir>/lib directory.
Migration Toolkit uses the configuration and connection information stored in the `toolkit.properties` file during the migration process to identify and connect to the source and target databases. On Linux, the `toolkit.properties` file is located in:

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

On Windows, the file is located in:

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

A sample `toolkit.properties` file is shown below:
Before executing Migration Toolkit commands, modify the toolkit.properties file with the editor of your choice. Update the file to include the following information:

- **SRC_DB_URL** specifies how Migration Toolkit should connect to the source database. See the section corresponding to your source database for details about forming the URL.
- **SRC_DB_USER** specifies a user name (with sufficient privileges) in the source database.
- **SRC_DB_PASSWORD** specifies the password of the source database user.
- **TARGET_DB_URL** specifies the JDBC URL of the target database.
- **TARGET_DB_USER** specifies the name of a privileged target database user.
- **TARGET_DB_PASSWORD** specifies the password of the target database user.
6.1 Defining an EDB Postgres Advanced Server URL

Migration Toolkit facilitates migration from the following platforms to EDB Postgres Advanced Server:

- Oracle
- MySQL
- Sybase
- SQL Server
- PostgreSQL

For a definitive list of the objects migrated from each database type, please refer to the Functionality Overview.

Migration Toolkit reads connection specifications for the source and the target database from the toolkit.properties file. Connection information for each must include:

- The URL of the database.
- The name of a privileged user.
- The password associated with the specified user.

The URL conforms to JDBC standards and takes the form:

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

An EDB Postgres Advanced Server URL contains the following information:

- **jdbc**
  - The protocol is always **jdbc**.

- **edb**
  - If you are using EDB Postgres Advanced Server, specify **edb** for the sub-protocol value.

- **host**
  - The name or IP address of the host where the Postgres instance is running.

- **port**
  - The port number that the EDB Postgres Advanced Server database listener is monitoring. The default port number is **5444**.

- **database_id**
  - The name of the source or target database.
{TARGET_DB_USER|SRC_DB_USER} must specify a user with privileges to CREATE each type of object migrated. If migrating data into a table, the specified user may also require INSERT, TRUNCATE, and REFERENCES privileges for each target table.

{TARGET_DB_PASSWORD|SRC_DB_PASSWORD} is set to the password of the privileged EDB Postgres Advanced Server user.

**Note:** You may specify non-superuser credentials for TARGET_DB_USER and TARGET_DB_PASSWORD while migrating to EDB Postgres Advanced Server. However, to migrate users, profiles, grants, and copy data via the dbLink_ora module (-copyViaDBLinkOra), you must use superuser credentials.
6.2 Defining a PostgreSQL URL

Migration Toolkit facilitates migration from the following platforms to PostgreSQL:

- Oracle
- MySQL
- SQL Server
- EDB Postgres Advanced Server

For a definitive list of the objects migrated from each database type, please refer to the Function-
ality Overview.

Migration Toolkit reads connection specifications for the source and the target database from the

 toolkit.properties file. Connection information for each must include:

- The URL of the database.
- The name of a privileged user.
- The password associated with the specified user.

A PostgreSQL URL conforms to JDBC standards and takes the form:

\{SRC_DB_URL|TARGET_DB_URL\}=jdbc:postgresql://host:port/
database_id

The URL contains the following information:

jdbc

The protocol is always jdbc.

postgresql

If you are using PostgreSQL, specify postgresql for the sub-protocol value.

host

The name or IP address of the host where the Postgres instance is running.

port

The port number that the Postgres database listener is monitoring. The default port

 number is 5432.

database_id

The name of the source or target database.

\{SRC_DB_USER|TARGET_DB_USER\} must specify a user with privileges to CREATE each
type of object migrated. If migrating data into a table, the specified user may also require INSERT,
TRUNCATE, and REFERENCES privileges for each target table.
{SRC_DB_PASSWORD|TARGET_DB_PASSWORD} is set to the password of the privileged PostgresQL user.

Note: You may specify non-superuser credentials for TARGET_DB_USER and TARGET_DB_PASSWORD while migrating to EDB Postgres Advanced Server. However, to migrate users, profiles, grants, and copy data via the dbLink_ora module (-copyViaDBLinkOra), you must use superuser credentials.
6.3 Defining an Oracle URL

Migration Toolkit facilitates migration from an Oracle database to a PostgreSQL or EDB Postgres Advanced Server database. When migrating from Oracle, you must specify connection specifications for the Oracle source database in the `toolkit.properties` file. The connection information must include:

- The URL of the Oracle database.
- The name of a privileged user.
- The password associated with the specified user.

When migrating from an Oracle database, `SRC_DB_URL` should contain a JDBC URL, specified in one of two forms. The first form is:

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

The second form is:

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

An Oracle URL contains the following information:

```
jdbc
  The protocol is always jdbc.
oracle
  The sub-protocol is always oracle.
thin
  The driver type. Specify a driver type of thin.
host_name
  The name or IP address of the host where the Oracle server is running.
port
  The port number that the Oracle database listener is monitoring.
database_id
  The database SID of the Oracle database.
service_name
  The name of the Oracle service.
```

`SRC_DB_USER` should specify the name of a privileged Oracle user. The Oracle user should have DBA privilege to migrate objects from Oracle to EDB Postgres Advanced Server. The DBA
privilege can be granted to the Oracle user with the Oracle `GRANT DBA TO user` command to ensure all of the desired database objects are migrated.

`SRC_DB_PASSWORD` must contain the password of the specified user.
6.4 Defining a MySQL URL

Migration Toolkit facilitates migration from a MySQL database to an EDB Postgres Advanced Server or PostgreSQL database. When migrating from MySQL, you must specify connection specifications for the MySQL source database in the toolkit.properties file. The connection information must include:

- The URL of the source database.
- The name of a privileged user.
- The password associated with the specified user.

When migrating from MySQL, SRC_DB_URL takes the form of a JDBC URL. For example:

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

The URL contains the following information:

- jdbc
  - The protocol is always jdbc.
- mysql
  - The sub-protocol is always mysql.
- //host_name
  - The name or IP address of the host where the source server is running.
- [port]
  - The port number that the MySQL database listener is monitoring.
- /database_id
  - The name of the source database.

SRC_DB_USER should specify the name of a privileged MySQL user.
SRC_DB_PASSWORD must contain the password of the specified user.
6.5 Defining a Sybase URL

Migration Toolkit facilitates migration from a Sybase database to an EDB Postgres Advanced Server database. When migrating from Sybase, you must specify connection specifications for the Sybase source database in the toolkit.properties file. The connection information must include:

- The URL of the source database.
- The name of a privileged user.
- The password associated with the specified user.

When migrating from Sybase, SRC_DB_URL takes the form of a JTDS URL. For example:

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

A Sybase URL contains the following information:

```
jdbc
jtds
sybase
host_name
port
database_id
```

The protocol is always jdbc.

The driver name is always jtds.

The server type is always sybase.

The name or IP address of the host where the source server is running.

The port number that the Sybase database listener is monitoring.

The name of the source database.

SRC_DB_USER should specify the name of a privileged Sybase user.

SRC_DB_PASSWORD must contain the password of the specified user.
6.6 Defining a SQL Server URL

Migration Toolkit facilitates migration from a SQL Server database to a PostgreSQL or EDB Postgres Advanced Server database. Migration Toolkit supports migration of the following object definitions:

- schemas
- tables
- table data
- constraints
- indexes

Migration Toolkit reads connection specifications for the source database from the toolkit.properties file. The connection information must include:

- The URL of the source database.
- The name of a privileged user.
- The password associated with the specified user.

If you are connecting to a SQL Server database, SRC_DB_URL takes the form of a JTDS URL. For example:

```
jdbc:jtds:sqlserver://server[:port]/database_id
```

A SQL Server URL contains the following information:

- **jdbc**
  - The protocol is always jdbc.
- **jtds**
  - The driver name is always jtds.
- **sqlserver**
  - The server type is always sqlserver.
- **server_name**
  - The name or IP address of the host where the source server is running.
- **port**
  - The port number that the source database listener is monitoring.
- **database_id**
  - The name of the source database.
**SRC_DB_USER** should specify the name of a privileged SQL Server user.

**SRC_DB_PASSWORD** must contain the password of the specified user.
CHAPTER 7

Invoking Migration Toolkit

After installing Migration Toolkit, and specifying connection properties for the source and target databases in the toolkit.properties file, Migration Toolkit is ready to perform migrations.

The Migration Toolkit executable is named runMTK.sh on Linux systems and runMTK.bat on Windows systems. On a Linux system, the executable is located in:

/usr/edb/migrationtoolkit/bin

On Windows, the executable is located in:

C:\Program Files\edb\mtk\bin

Note: If the following error appears upon invoking the Migration Toolkit, check the file permissions of the toolkit.properties file.

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.

The operating system user account running the Migration Toolkit must be the owner of the toolkit.properties file with a minimum of read permission on the file. In addition, there must be no permissions of any kind for group and other users. The following is an example of the recommended file permissions where user enterprisedb is running the Migration Toolkit.

-rw------- 1 enterprisedb enterprisedb 191 Aug 1 09:59 toolkit.properties
Importing Character Data with Embedded Binary Zeros (NULL characters)

Migration Toolkit properly supports importation of a column with a value of NULL. However, Migration Toolkit does not support importation of NULL character values (embedded binary zeros 0x00) with the JDBC connection protocol. If you are importing data that includes the NULL character, use the \texttt{-replaceNullChar} option to replace the NULL character with a single, non-NULL, replacement character.

\textbf{Note}:

\begin{itemize}
  \item MTK implicitly replaces NULL characters with an empty string.
  \item The \texttt{-replaceNullChar} option does not work with \texttt{-copyViaDBLinkOra} option.
\end{itemize}

Once the data has been migrated, use a SQL statement to replace the character specified by \texttt{-replaceNullChar} with binary zeros.
7.1 Migrating a Schema from Oracle

Unless specified in the command line, Migration Toolkit expects the source database to be Oracle and the target database to be EDB Postgres Advanced Server. To migrate a complete schema on Linux, navigate to the executable and invoke the following command:

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

To migrate a complete schema on Windows, navigate to the executable and invoke the following command:

```
> .\runMTK.bat <schema_name>
```

Where:

- `schema_name` is the name of the schema within the source database (specified in the toolkit.properties file) that you wish to migrate. You must include at least one `schema_name`.

**Note:** When the default database user of a migrated schema is automatically migrated, the custom profile of the default database user is also migrated if such a custom profile exists. A custom profile is a user-created profile. For example, custom profiles exclude Oracle profiles DEFAULT and MONITORING_PROFILE.

You can migrate multiple schemas by following the command name with a comma-delimited list of schema names.

On Linux, execute the following command:

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

On Windows, execute the following command:

```
> .\runMTK.bat <schema_name1>,<schema_name2>, <schema_name3>
```
7.2 Migrating from a Non-Oracle Source Database

If you do not specify a source database type and a target database type, Postgres assumes the source database to be Oracle, and the target database to be EDB Postgres Advanced Server.

To invoke Migration Toolkit, open a command window, navigate to the executable, and invoke the following command:

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

Where:

-sourcedbtype source_type

source_type specifies the server type of the source database. source_type is case-insensitive. By default, source_type is oracle. source_type may be one of the following values:

<table>
<thead>
<tr>
<th>To migrate from:</th>
<th>Specify:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oracle</td>
<td>oracle (the default value)</td>
</tr>
<tr>
<td>MySQL</td>
<td>mysql</td>
</tr>
<tr>
<td>SQL Server</td>
<td>sqlserver</td>
</tr>
<tr>
<td>Sybase</td>
<td>sybase</td>
</tr>
<tr>
<td>PostgreSQL</td>
<td>postgres or postgresql</td>
</tr>
<tr>
<td>EDB Postgres Advanced Server</td>
<td>enterprisedb</td>
</tr>
</tbody>
</table>

-targetdbtype target_type

target_type specifies the server type of the target database. target_type is case-insensitive. By default, target_type is enterprisedb. target_type may be one of the following values:

<table>
<thead>
<tr>
<th>To migrate to:</th>
<th>Specify:</th>
</tr>
</thead>
<tbody>
<tr>
<td>EDB Postgres Advanced Server</td>
<td>enterprisedb</td>
</tr>
<tr>
<td>PostgreSQL</td>
<td>postgres or postgresql</td>
</tr>
</tbody>
</table>

schema_name

schema_name is the name of the schema within the source database (specified in the toolkit.properties file) that you wish to migrate. You must include at least one schema_name.

The following example migrates a schema (table definitions and table content) named HR from a MySQL database on a Linux system to an EDB Postgres Advanced Server host. Note that the command includes the -sourcedbtype and -targetdbtype options:
$ ./runMTK.sh -sourcedbtype mysql -targetdbtype enterprise db HR

On Windows, use the following command:

> .\runMTK.bat -sourcedbtype mysql -targetdbtype enterprise db HR

You can migrate multiple schemas from a source database by including a comma-delimited list of schemas at the end of the Migration Toolkit command. The following example migrates multiple schemas (named HR and ACCTG) from a MySQL database to a PostgreSQL database:

On Linux, use the following command to migrate multiple schemas from a MySQL database:

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

On Windows, use the following command form:

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

7.2. Migrating from a Non-Oracle Source Database

56
Append migration options when you run Migration Toolkit to conveniently control details of the migration. For example, to migrate all schemas within a database, append the -allSchemas option to the command:

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

The sections that follow contain reference material for each of the command options that work with Migration Toolkit; options are grouped by their behavior. The table below lists Migration Toolkit options and the sections that they are grouped in.
<table>
<thead>
<tr>
<th>Feature:</th>
<th>Relevant Options:</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Offline Migration Options</strong></td>
<td>-offlineMigration</td>
</tr>
<tr>
<td><strong>Import Options</strong></td>
<td>-sourcedbtype, -targetdbtype, -schemaOnly, -dataOnly</td>
</tr>
<tr>
<td><strong>Schema Creation Options</strong></td>
<td>-dropSchema, -targetSchema</td>
</tr>
<tr>
<td><strong>Schema Object Selection Options</strong></td>
<td>-allTables, -tables, -constraints, -ignoreCheckConstFilter, -skipCKConst, -skipFKConst, -skipColDefaultClause, -indexes, -triggers, -allViews, -views, -allSequences, -sequences, -allProcs, -procs, -allFuncs, -funcs, -checkFunctionBodies, -allPackages, -packages, -allDomains, -allQueues, -queues, -allRules,</td>
</tr>
<tr>
<td><strong>Migration Options</strong></td>
<td>-truncLoad, -enableConstBeforeDataLoad, -retryCount, -safeMode, -fastCopy, -analyze, vacuumAnalyze, -replaceNullChar, -copyDelimiter, -batchSize, -cpBatchSize, -lobBatchSize, -fetchSize, -filterProp -customColTypeMapping, -customColTypeMappingFile</td>
</tr>
<tr>
<td><strong>Oracle Specific Options</strong></td>
<td>-allUsers, -users, -allProfiles, -profiles, -importPartitionAsTable, -objectTypes, -copyViaDBLinkOra, -allDBLinks -allSynonyms, -allPublicSynonyms, -allPrivateSynonyms, -useOraCase, -skipUserSchemaCreation</td>
</tr>
<tr>
<td><strong>Miscellaneous Options</strong></td>
<td>-help, -logDir, -logFileCount, -logFileSize, -logBadSQL -verbose, -version</td>
</tr>
</tbody>
</table>
8.1 Offline Migration Options

If you specify the `-offlineMigration` option in the command line, Migration Toolkit performs an offline migration. During an offline migration, Migration Toolkit reads the definition of each selected object and creates an SQL script that, when executed at a later time, replicates each object in Postgres.

Note: The following examples demonstrate invoking Migration Toolkit in Linux; to invoke Migration Toolkit in Windows, substitute the `runMTK.bat` command for the `runMTK.sh` command.

To perform an offline migration of both schema and data, specify the `-offlineMigration` keyword, followed by the schema name:

$ ./runMTK.sh -offlineMigration <schema_name>

Each database object definition is saved in a separate file with a name derived from the schema name and object type in the user’s home folder. To specify an alternative file destination, include a directory name after the `-offlineMigration` option:

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

To perform an offline migration of schema objects only (creating empty tables), specify the `-schemaOnly` keyword in addition to the `-offlineMigration` keyword when invoking Migration Toolkit:

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

To perform an offline migration of data only (omitting any schema object definitions), specify the `-dataOnly` keyword and the `-offlineMigration` keyword when invoking Migration Toolkit:

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

By default, data is written in COPY format; to write the data in a plain SQL format, include the `-safeMode` keyword:

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

By default, when you perform an offline migration that contains table data, a separate file is created for each table. To create a single file that contains the data from multiple tables, specify the `-singleDataFile` keyword:

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

Please note: the `-singleDataFile` option is available only when migrating data in a plain SQL format; you must include the `-safeMode` keyword if you include the `-singleDataFile` option.
8.1.1 Executing Offline Migration Scripts

You can use the edb-psql (or psql) command line to execute the scripts generated during an offline migration. The following example describes restoring a schema (named hr) into a new database (named acctg) stored in EDB Postgres Advanced Server.

1. Use the `createdb` command to create the acctg database, into which we will restore the migrated database objects:
   
   `createdb -U enterprisedb acctg`

2. Connect to the new database with edb-psql:
   
   `edb-psql -U enterprisedb acctg`

3. Use the `\i` meta-command to invoke the migration script that creates the object definitions:
   
   `acctg=# \i ./mtk_hr_ddl.sql`

4. If the `-offlineMigration` command included the `-singleDataFile` keyword, the `mtk_hr_data.sql` script will contain the commands required to recreate all of the objects in the new target database. Populate the database with the command:
   
   `acctg=# \i ./mtk_hr_data.sql`
8.2 Import Options

By default, Migration Toolkit assumes the source database to be Oracle and the target database to be EDB Postgres Advanced Server; include the `-sourcedbtype` and `-targetdbtype` keywords to specify a non-default source or target database.

By default, Migration Toolkit imports both the data and the object definition when migrating a schema; alternatively you can choose to import either the data or the object definitions.

```
-sourcedbtype <source_type>
```

The `-sourcedbtype` option specifies the source database type. `source_type` may be one of the following values: `mysql`, `oracle`, `sqlserver`, `sybase`, `postgresql` or `enterprisedb`. `source_type` is case-insensitive. By default, `source_type` is `oracle`.

```
-targetdbtype <target_type>
```

The `-targetdbtype` option specifies the target database type. `target_type` may be one of the following values: `enterprisedb`, `postgres`, or `postgresql`. `target_type` is case-insensitive. By default, `target_type` is `enterprisedb`.

```
-schemaOnly
```

This option imports the schema definition and creates all selected schema objects in the target database. This option cannot be used in conjunction with `-dataOnly` option.

```
-dataOnly
```

This option copies the data only. When used with the `-tables` option, Migration Toolkit will only import data for the selected tables (see usage details below). This option cannot be used with `-schemaOnly` option.
8.3 Schema Creation Options

By default, Migration Toolkit imports the source schema objects and/or data into a schema of the same name. If the target schema does not exist, Migration Toolkit creates a new schema. Alternatively, you may specify a custom schema name via the `-targetSchema` option. You can choose to drop the existing schema and create a new schema using the following option:

```
-dropSchema [true|false]
```

When set to true, Migration Toolkit drops the existing schema (and any objects within that schema) and creates a new schema. (By default, `-dropSchema` is false).

```
-targetSchema <schema_name>
```

Use the `-targetSchema` option to specify the name of the migrated schema. If you are migrating multiple schemas, specify a name for each schema in a comma-separated list with no intervening space characters. If the command line does not include the `-targetSchema` option, the name of the new schema will be the same as the name of the source schema.

You cannot specify `information-schema`, `dbo`, `sys`, or `pg_catalog` as target schema names. These schema names are reserved for meta-data storage in EDB Postgres Advanced Server.
8.4 Schema Object Selection Options

Use the following options to select specific schema objects to migrate:

-allTables
   Import all tables from the source schema.

-tables <table_list>
   Import the selected tables from the source schema. table_list is a comma-separated list (with no intervening space characters) of table names (e.g., -tables emp,dept,acctg).

-constraints
   Import the table constraints. This option is valid only when importing an entire schema or when you specify the -allTables or -tables <table_list> options.

-ignoreCheckConstFilter
   By default, Migration Toolkit does not implement migration of check constraints and default clauses from a Sybase database. Include the -ignoreCheckConstFilter parameter when specifying the -constraints parameter to migrate constraints and default clauses from a Sybase database.

-skipCKConst
   Omit the migration of check constraints. This option is useful when migrating check constraints that are based on built-in functions (in the source database) that are not supported in the target database.
   This option is valid only when importing an entire schema or when the -allTables or -tables <table_list> options are specified.

-skipFKConst
   Omit the migration of foreign key constraints. This option is valid only when importing an entire schema or when the -allTables or -tables <table_list> options are specified.

-skipColDefaultClause
   Omit the migration of the column DEFAULT clause.

-indexes
   Import the table indexes. This option is valid when importing an entire schema or when the -allTables or -tables <table_list> option is specified.

-triggers
Import the table triggers. This option is valid when importing an entire schema or when the allTables or -tables <table_list> option is specified.

-allViews

Import the views from the source schema. Please note that this option will migrate both dynamic and materialized views from the source. (Oracle and Postgres materialized views are supported.)

-views <view_list>

Import the specified materialized or dynamic views from the source schema. (Oracle and Postgres materialized views are supported.) view_list is a comma-separated list (with no intervening space characters) of view names (e.g., -views all_emp, mgmt_list, acct_list).

-allSequences

Import all sequences from the source schema.

-sequences <sequence_list>

Import the selected sequences from the source schema. <sequence_list> is a comma-separated list (with no intervening space characters) of sequence names.

-allProcs

Import all stored procedures from the source schema.

-procs <procedures_list>

Import the selected stored procedures from the source schema. procedures_list is a comma-separated list (with no intervening space characters) of procedure names.

-allFuncs

Import all functions from the source schema.

-funcs <function_list>

Import the selected functions from the source schema. function_list is a comma-separated list (with no intervening space characters) of function names.

-checkFunctionBodies [true/false]

When false, disables validation of the function body during function creation (to avoid errors if the function contains forward references). The default value is true.

-allPackages

Import all packages from the source schema.

-packages <package_list>

Import the selected packages from the source schema. package_list is a comma-separated list (with no intervening space characters) of package names.
-allDomains

Import all domain, enumeration and composite types from the source database; this option is only valid when both the source and target are stored on a Postgres host.

-allQueues

Import all queues from the source schema. These are queues created and managed by the DBMS_AQ and DBMS_AQADM built-in packages. When Oracle is the source database, the -objectTypes option must also be specified. When EDB Postgres Advanced Server is the source database, the -allDomains and -allTables options must also be specified. (Oracle and EDB Postgres Advanced Server queues are supported.)

-queues <queue_list>

Import the selected queues from the source schema. queue_list is a comma-separated list (with no intervening space characters) of queue names. These are queues created and managed by the DBMS_AQ and DBMS_AQADM built-in packages. When Oracle is the source database, the -objectTypes option must also be specified. When EDB Postgres Advanced Server is the source database, the -allDomains and -allTables options must also be specified. (Oracle and EDB Postgres Advanced Server queues are supported.)

-allRules

Import all rules from the source database; this option is only valid when both the source and target are stored on a Postgres host.
8.5 Migration Options

Use the migration options listed below to control the details of the migration process.

- **-loaderCount [<value>]**
  
  Use the `-loaderCount` option to specify the number of parallel threads that Migration Toolkit should use when importing data. This option is particularly useful if the source database contains a large volume of data, and the Postgres host (that is running Migration Toolkit) has high-end CPU and RAM resources. While `value` may be any non-zero, positive number, we recommend that `value` should not exceed the number of CPU cores; a dual core CPU should have an optimal `value` of 2.

  Please note that specifying too large of a `value` could cause Migration Toolkit to terminate, generating a 'Out of heap space' error.

- **-truncLoad**
  
  Truncate the data from the table before importing new data. This option can only be used in conjunction with the `-dataOnly` option.

- **-enableConstBeforeDataLoad**
  
  Include the `-enableConstBeforeDataLoad` option if a non-partitioned source table is mapped to a partitioned table. This option enables all triggers on the target table (including any triggers that redirect data to individual partitions) before the data migration. `-enableConstBeforeDataLoad` is valid only if the `-truncLoad` parameter is also specified.

- **-retryCount [<value>]**
  
  If you are performing a multiple-schema migration, objects that fail to migrate during the first migration attempt due to cross-schema dependencies may successfully migrate during a subsequent migration. Use the `-retryCount` option to specify the number of attempts that Migration Toolkit will make to migrate an object that has failed during an initial migration attempt. Specify a `value` that is greater than 0; the default value is 2.

- **-safeMode**
  
  If you include the `-safeMode` option, Migration Toolkit commits each row as migrated; if the migration fails to transfer all records, rows inserted prior to the point of failure will remain in the target database.

- **-fastCopy**
  
  Including the `-fastCopy` option specifies that Migration Toolkit should bypass WAL logging to perform the COPY operation in an optimized way, default disabled. If you choose to use the `-fastCopy` option, migrated data may not be recoverable (in the target database) if the migration is interrupted.
-replaceNullChar <value>

The Migration Toolkit properly supports importation of a column with a value of NULL.

However, the Migration Toolkit does not support importation of NULL character values (embedded binary zeros 0x00) with the JDBC connection protocol. If you are importing data that includes the NULL character, use the -replaceNullChar option to replace the NULL character with a single, non-NUL, replacement character. Do not enclose the replacement character in quotes or apostrophes.

Once the data has been migrated, use a SQL statement to replace the character specified by -replaceNullChar with binary zeros.

-analyze

Include the -analyze option to invoke the Postgres ANALYZE operation against a target database. The optimizer consults the statistics collected by the ANALYZE operation, utilizing the information to construct efficient query plans.

-vacuumAnalyze

Include the -vacuumAnalyze option to invoke both the VACUUM and ANALYZE operations against a target database. The optimizer consults the statistics collected by the ANALYZE operation, utilizing the information to construct efficient query plans. The VACUUM operation reclaims any storage space occupied by dead tuples in the target database.

copyDelimiter

Specify a single character to be used as a delimiter in the copy command when loading table data. The default value is ' \t ' (tab).

-batchSize

Specify the batch size of bulk inserts. Valid values are 1-1000. The default batch size is 1000; reduce the value of -batchSize if Out of Memory exceptions occur.

-cpBatchSize

Specify the batch Size in MB to be used in the COPY command. Any value greater than 0 is valid; the default batch size is 8 MB.

-lobBatchSize

Specify the number of rows to be loaded in a batch for LOB data types. The data migration for a table containing a large object type (LOB) column such as BYTEA, BLOB, or CLOB, etc., is performed one row at a time by default. This is to avoid an out of heap space error in case an individual LOB column holds hundreds of megabytes of data. In case the LOB column average data size is at a lower end, you can customize the LOB batch size by specifying the number of rows in each batch with any value greater than 0.
-fetchSize

Use the -fetchSize option to specify the number of rows fetched in a result set. If the designated -fetchSize is too large, you may encounter Out of Memory exceptions; include the -fetchSize option to avoid this pitfall when migrating large tables. The default fetch size is specific to the JDBC driver implementation, and varies by database.

MySQL users note: By default, the MySQL JDBC driver will fetch all of the rows that reside in a table into the client application (Migration Toolkit) in a single network round-trip. This behavior can easily exceed available memory for large tables. If you encounter an 'out of heap space' error, specify -fetchSize 1 as a command line argument to force Migration Toolkit to load the table data one row at a time.

-filterProp <file_name>

file_name specifies the name of a file that contains constraints in key=value pairs. Each record read from the database is evaluated against the constraints; those that satisfy the constraints are migrated. The left side of the pair lists a table name; please note that the table name should not be schema-qualified. The right side specifies a condition that must be true for each row migrated. For example, including the following constraints in the property file.

countries=country_id<>'AR'

migrates only those countries with a country_id value that is not equal to AR; this constraint applies to the countries table.

You can also specify conditions for multiple tables. However, the condition for each table must be on a new line in the property file.

Example:

Following entries in the properties file will only migrate the relevant data from EMPLOYEES and the DEPARTMENTS tables.

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

    DEPARTMENTS=(DEPARTMENT_ID BETWEEN 10 AND 30)

-customColTypeMapping <column_list>

Use custom type mapping to change the data type of migrated columns. The left side of each pair specifies the columns with a regular expression; the right side of each pair names the data type that column should assume. You can include multiple pairs in a semi-colon separated column_list. For example, to map any column whose name ends in ID to type INTEGER, use the following custom mapping entry:

    .*ID=INTEGER
Custom mapping is applied to all table columns that match the criteria unless the column is table-qualified.

The '\\' characters act as an escape string; since ' .' is a reserved character in regular expressions, on Linux use ' \\.' to represent the ' .' character. For example, to use custom mapping to select rows from the EMP_ID column in the EMP table, specify the following custom mapping entry:

EMP\.EMP_ID=INTEGER

On Windows, use '. ' to represent the ' .' character:

EMP\.EMP_ID=INTEGER

-cust omColTypeMappingFile <property_file>

You can include multiple custom type mappings in a property_file; specify each entry in the file on a separate line, in a key=value pair. The left side of each pair selects the columns with a regular expression; the right side of each pair names the data type that column should assume.
8.6 Oracle Specific Options

The following options apply only when the source database is Oracle.

-o objectTypes

    Import the user-defined object types from the schema list specified at the end of the runMTK.sh command.

-a allUsers

    Import all users and roles from the source database. Please note that the -allUsers option is only supported when migrating from an Oracle database to an EDB Postgres Advanced Server database.

-u <user_list>

    Import the selected users or roles from the source Oracle database. user_list is a comma-separated list (with no intervening space characters) of user/role names (e.g., -users MTK, SAMPLE, acctg). Please note that the -users option is only supported when migrating from an Oracle database to an EDB Postgres Advanced Server database.

-allProfiles

    Import all custom (that is, user-created) profiles from the source database. Other Oracle non-custom profiles such as DEFAULT and MONITORING_PROFILE are not imported.

    For the imported profiles, only the following password parameters associated with the profiles are imported:

        FAILED_LOGIN_ATTEMPTS
        PASSWORD_LIFE_TIME
        PASSWORD_REUSE_TIME
        PASSWORD_REUSE_MAX
        PASSWORD_LOCK_TIME
        PASSWORD_GRACE_TIME
        PASSWORD_VERIFY_FUNCTION

    All other profile parameters such as the Oracle resource parameters are not imported. The Oracle database user specified by SRC_DB_USER must have SELECT privilege on the Oracle data dictionary view DBA_PROFILES.

    Please note that the -allProfiles option is only supported when migrating from an Oracle database to an EDB Postgres Advanced Server database.
-profiles <profile_list>

Import the selected, custom (that is, user-created) profiles from the source Oracle database. profile_list is a comma-separated list (with no intervening space characters) of profile names (e.g., -profiles ADMIN_PROFILE,USER_PROFILE). Oracle non-custom profiles such as DEFAULT and MONITORING_PROFILE are not imported.

As with the -allProfiles option, only the password parameters are imported. The Oracle database user specified by SRC_DB_USER must have SELECT privilege on the Oracle data dictionary view DBA_PROFILES.

Please note that the -profiles option is only supported when migrating from an Oracle database to an EDB Postgres Advanced Server database.

-importPartitionAsTable <table_list>

Include the -importPartitionAsTable parameter to import the contents of a partitioned table that resides on an Oracle host into a single non-partitioned table. table_list is a comma-separated list (with no intervening space characters) of table names (e.g., -importPartitionAsTable emp,dept,acctg).

copyViaDBLinkOra

The dblink_ora module provides EDB Postgres Advanced Server-to-Oracle connectivity at the SQL level. dblink_ora is bundled and installed as part of the EDB Postgres Advanced Server database installation. dblink_ora utilizes the COPY API method to transfer data between databases. This method is considerably faster than the JDBC COPY method.

The following example uses the dblink_ora COPY API to migrate all tables from the HR schema:

$./runMTK.sh -copyViaDBLinkOra -allTables HR

The target EDB Postgres Advanced Server database must have dblink_ora installed and configured. For information about dblink_ora, please see the Database Compatibility for Oracle Developer’s Guide.

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

Choose this option to migrate Oracle database links. The password information for each link connection in the source database is encrypted, so unless specified, a dummy password (edb) is substituted.

To migrate all database links using edb as the dummy password for the connected user:

$./runMTK.sh -allDBLinks HR

You can alternatively specify the password for each of the database links through a comma-separated list (with no intervening space characters) of name=value pairs.
Specify the link name on the left side of the pair and the password value on the right side.

To migrate all database links with the actual passwords specified on the command-line:
```
./runMTK.sh -allDBLinks LINK_NAME1=abc,LINK_NAME2=xyz HR
```

Migration Toolkit migrates only the database link types that are currently supported by EnterpriseDB; this includes fixed user links of public and private type.

- **-allSynonyms**

  Include the `-allSynonyms` option to migrate all public and private synonyms from an Oracle database to an EDB Postgres Advanced Server database. If a synonym with the same name already exists in the target database, the existing synonym will be replaced with the migrated version.

- **-allPublicSynonyms**

  Include the `-allPublicSynonyms` option to migrate all public synonyms from an Oracle database to an EDB Postgres Advanced Server database. If a synonym with the same name already exists in the target database, the existing synonym will be replaced with the migrated version.

- **-allPrivateSynonyms**

  Include the `-allPrivateSynonyms` option to migrate all private synonyms from an Oracle database to an EDB Postgres Advanced Server database. If a synonym with the same name already exists in the target database, the existing synonym will be replaced with the migrated version.

- **-useOraCase**

  Include the `-useOraCase` option to preserve the Oracle default, uppercase naming convention for all database objects when migrating from an Oracle database to an EDB Postgres Advanced Server database.

  The uppercase naming convention is preserved for tables, views, sequences, procedures, functions, triggers, packages, etc. For these database objects, the uppercase naming convention is applied to a) the names of the database objects, b) the column names, key names, index names, constraint names, etc., of the tables and views, c) the `SELECT` column list for a view, and d) the parameter names that are part of the procedure or function header.

  **Note:** Within the procedural code body of a procedure, function, trigger or package, identifier references may have to be manually edited in order for the program to execute properly without an error. Such corrections are in regard to the proper case conversion of identifier references that may or may not have occurred.
When you specify the `--useOraCase` option, you may need to specify the `--skipUserSchemaCreation` option as well. For information, see the description of the `--skipUserSchemaCreation` option in this section.

The default behavior of the Migration Toolkit (without using the `--useOraCase` option) is that database object names are extracted from Oracle without enclosing quotation marks (unless the database object was explicitly created in Oracle with enclosing quotation marks). The following is a portion of a table command generated by the Migration Toolkit with the `--offlineMigration` option:

```sql
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);
```

When this table is then migrated to, and created in EDB Postgres Advanced Server, all unquoted object names are converted to lowercase letters, so the table appears in EDB Postgres Advanced Server as follows:

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

If your EDB Postgres Advanced Server applications are referencing the migrated database objects using quoted uppercase identifiers, the applications will fail since the database object names are now in lowercase.

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

If your application uses quoted upper-case identifiers, perform the migration with the `--useOraCase` option. The DDL will enclose all database object names in quotes:

```sql
CREATE TABLE "DEPT" (  
    "DEPTNO" NUMBER(2) NOT NULL,  
```

(continues on next page)
When this table is migrated to, and created in EDB Postgres Advanced Server, all object names are maintained in uppercase letters, so the table appears in EDB Postgres Advanced Server as follows:

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Modifiers</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEPTNO</td>
<td>numeric(2,0)</td>
<td>not null</td>
</tr>
<tr>
<td>DNAME</td>
<td>character varying(14)</td>
<td></td>
</tr>
<tr>
<td>LOC</td>
<td>character varying(13)</td>
<td></td>
</tr>
</tbody>
</table>

Indexes:
- "DEPT_PK" PRIMARY KEY, btree ("DEPTNO")
- "DEPT_DNAME_UQ" UNIQUE CONSTRAINT, btree ("DNAME")

Applications can then access the object using quoted uppercase names.

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

Skipping User Schema Creation

When an Oracle user is migrated, a role (that is, a user name) is created in the target database server for the Oracle user if the role does not already exist. The role name is created in lowercase letters. When a new role is created, a schema with the same name is also created in lowercase letters.

Specification of the -skipUserSchemaCreation option prevents the automatic schema creation for a migrated Oracle user name. This option is particularly useful when the -useOraCase option is specified in order to prevent creation of two schemas with the same name except for one schema name in lowercase letters and the other in uppercase letters. Specifying the -useOraCase option results in the creation of a schema in the Oracle naming convention of uppercase letters for the source schema specified following the options list when Migration Toolkit is invoked.

Thus, if the -useOraCase option is specified without the -skipUserSchemaCreation option, the target database results in having
two identically named schemas with one in lowercase letters and the other in uppercase letters. If the `-useOraCase` option is specified along with the `-skipUserSchemaCreation` option, the target database will have only the schema in uppercase letters.
8.7 Miscellaneous Options

Use the migration options listed below to view Migration Toolkit help and version information; you can also use the options in this section to control Migration Toolkit feedback and logging options.

- **help**
  Display the application command-line usage information.

- **logDir <log_path>**
  Include this option to specify where the log files will be written; log_path represents the path where application log files are saved. By default, on Linux log files are written to:

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

  On Windows, the log files are saved to:

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

- **LogFileCount <file_count>**
  Include this option to specify the number of files used in log file rotation. Specify a value of 0 to disable log file rotation and create a single log file (it will be truncated when it reaches the value specified using the logFileSize option). file_count must be greater than or equal to 0; the default is 20.

- **logFileSize <file_size>**
  Include this option to specify the maximum file size limit (in MB) before rotating to a new log file. file_size must be greater than 0; the default is 50 MB.

- **logBadSQL**
  Include this option to have the schema definition (DDL script) of any failed objects saved to a file. The file is saved under the same path that is used for the error logs and is named in the format mtk_bad_sql_<schema_name_timestamp>.sql where schema_name is the name of the schema and timestamp is the timestamp of the Migration Toolkit run.

- **verbose [on|off]**
  Display application log messages on standard output (By default, verbose is on).

- **version**
  Display the Migration Toolkit version.
8.8 Example

The following example demonstrates performing an Oracle to EDB Postgres Advanced Server migration.

The following is the content of the `toolkit.properties` file.

```
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
```

The following command invokes Migration Toolkit:

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

```
Running EnterpriseDB Migration Toolkit (Build 48.0.0) ...
Source database connectivity info...
conn =jdbc:oracle:thin:@192.168.2.6:1521:xe
user =edb
password=******\*
Target database connectivity info...
conn =jdbc:edb://localhost:5444/edb
user =enterprisedb
password=******\*
Connecting with source Oracle database server...
Connected to Oracle, version 'Oracle Database 10g Express Edition Release 10.2.0.1.0 - Production'
Connecting with target EnterpriseDB database server...
Connected to EnterpriseDB, version '9.4.0.0'
Importing redwood schema EDB...
Creating Schema...edb
Creating Sequence: NEXT_EMPNO
Creating Tables...
Creating Table: BAD_TABLE
MTK-15013: Error Creating Table BAD_TABLE
DB-42704: ERROR: type "binary_double" does not exist at position 58
-- CREATE TABLE BAD_TABLE (  
-- F1 NUMBER NOT NULL,  
-- Line 3: F2 BINARY_DOUBLE  
-- ^
Creating Table: DEPT
Creating Table: EMP
Creating Table: JOBHIST
(continues on next page)
```
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] Table Data Load Summary: Total Time(s): 0.077 Total Rows: 14
Loading Table: JOBHIST ...
[JOBHIST] Migrated 17 rows.
[JOBHIST] Table Data Load Summary: Total Time(s): 0.042 Total Rows: 17
Total Size(MB): 9.765625E-4
Loading Table: "MixedCase" ...
["MixedCase"] Table Data Load Summary: Total Time(s): 0.098 Total Rows: 0
Loading Table: "lowercase" ...
["lowercase"] Table Data Load Summary: Total Time(s): 0.066 Total Rows: 0
Data Load Summary: Total Time (sec): 0.806 Total Rows: 35 Total Size(MB): 0.001
Creating Constraint: DEPT_PK
Creating Constraint: DEPT_DNAME_UQ
Creating Constraint: EMP_PK
Creating Constraint: JOBHIST_PK
Creating Constraint: SYS_C008958
MTK-15001: Error Creating Constraint SYS_C008958
DB-42P01: com.edb.util.PSQLException: ERROR: relation "bad_table" does not exist
Creating Constraint: EMP_REF_DEPT_FK
Creating Constraint: EMP_SAL_CK
Creating Constraint: JOBHIST_REF_DEPT_FK
Creating Constraint: JOBHIST_REF_EMP_FK
Creating Constraint: JOBHIST_DATE_CHK
Creating Trigger: USER_AUDIT_TRIG
Creating Trigger: EMP_SAL_TRIG
MTK-13009: Warning! Skipping migration of trigger DROP_TRIGGER, currently non-table triggers are not supported in target database.
Creating View: SALESEMP
Creating Function: EMP_COMP
Creating Package: EMP_ADMIN
MTK-16005: Package Body is Invalid, Skipping...
Schema EDB imported with errors.

(continues on next page)
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
=======================
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)

**********************************************************************

8.8. Example
Note the omission of skipped and unsupported database objects. The migration information is summarized in the Migration Summary at the end of the run.
During the migration process, the migration summary displays the progress of the migration. If an error occurs, the summary will display information about the error. The migration summary is also written to a log file. The default locations for the log files are:

**On Linux:**

$HOME/.enterprisedb/migration-toolkit/logs

**On Windows,** the log files are saved to:

%HOMEDRIVE%%HOMEPATH%\.enterprisedb\migration-toolkit\logs

You can specify an alternate log file directory with the `-logdir <log_path>` option in Migration Toolkit.

### 9.1 Connection Errors

Migration Toolkit uses information from the `toolkit.properties file` to connect to the source and target databases. Most of the connection errors that occur when using Migration Toolkit are related to the information specified in the `toolkit.properties file`. Use the following section to identify common connection errors, and learn how to resolve them.
9.1.1 Invalid username/password

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

```
MTK-11009: Error Connecting Database "Oracle"
DB-1017: java.sql.SQLException: ORA-01017: invalid username/password;
logon denied
The user name or password specified in the toolkit.properties file is not valid to use to connect to the Oracle source database.
```

To resolve this error, edit the `toolkit.properties` file, specifying the name and password of a valid user with sufficient privileges to perform the migration in the `SRC_DB_USER` and `SRC_DB_PASSWORD` properties.

9.1.2 Connection rejected: FATAL: password

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

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

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

To resolve this error, edit the `toolkit.properties` file, specifying the name and password of a valid user with sufficient privileges to perform the migration in the `TARGET_DB_USER` and `TARGET_DB_PASSWORD` properties.

9.1.3 Exception: ORA-28000: the account is locked

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

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

To resolve this error, you can either unlock the user account on the Oracle server or edit the `toolkit.properties` file, specifying the name and password of a valid user with sufficient privileges to perform the migration in the `SRC_DB_USER` and `SRC_DB_PASSWORD` parameters.
9.1.4 Exception: oracle.jdbc.driver.OracleDriver

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

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

Before using Migration Toolkit, you must download and install the appropriate JDBC driver for the database that you are migrating from. See Installing Source-Specific Drivers for complete instructions.

9.1.5 I/O exception: The Network Adapter could not establish the connection

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

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

The JDBC URL for the source database specified in the toolkit.properties file contains invalid connection properties.

To resolve this error, edit the toolkit.properties file specifying valid connection information for the source database in the SRC_DB_URL property. Please note that the SRC_DB_URL value is database-specific; consult the table of contents for detailed information for your source database type.

9.1.6 Exception: The URL specified for the “target” database is invalid

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

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

The JDBC URL for the target database (EDB Postgres Advanced Server) specified in the toolkit.properties file contains invalid connection properties.

To resolve this error, edit the toolkit.properties file, specifying valid connection information for the target database in the TARGET_DB_URL property. For information about forming a JDBC URL for EDB Postgres Advanced Server, see Defining an EDB Postgres Advanced Server URL.
9.2 Migration Errors

The following errors may occur after Migration Toolkit has successfully connected to the target and source database servers.

9.2.1 ERROR: Extra Data after last expected column

When migrating a table online, I get the error message:

```
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"
```

This error occurs when the data in a column in `table_name` includes the delimiter character. To correct this error, change the delimiter character to a character not found in the table contents.

**Note:** In this example, the pipe character (`|`) occurs in the text, `"LOS|ANGELES"`, intended for insertion into the last column, and the Migration Toolkit is run using the `-copyDelimiter '|'` option, which results in the error.

9.2.2 Error Loading Data into Table: TABLE_NAME

When performing a data-only migration, I get the following error:

```
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
```

You must create a table to receive the data in the target database before you can migrate the data. Verify that a table (with a name of `TABLE_NAME`) exists in the target database; create the table if necessary and re-try the data migration.
9.2.3 Error Creating Constraint CONS_NAME_FK

When I perform a table migration that includes indexes and constraints, I get the following error message:

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

One or more schema objects could not be imported during the migration process. Please review the migration output for more details.

Migration logs have been saved to
/home/user/.enterprisedb/migration-toolkit/logs

Migration Summary

Tables: 1 out of 1
Constraints: 4 out of 6
Total objects: 7
Successful count: 5
Failed count: 2
Invalid count: 0

List of failed objects

Constraints

1. HR.EMPLOYEES.EMP.DEPT.FK
2. HR.EMPLOYEES.EMP.JOB.FK

The table you are migrating includes a foreign key constraint on a table that does not exist in the target database. Migration Toolkit creates the table, omitting the foreign key constraint.

You can avoid generating the error message by including the -skipFKConst option in the Migration Toolkit command.

9.2.4 Error Loading Data into Table

I've already migrated the table definition; when I try to migrate the data into the table, I get an error:

MTK-17001: Error Loading Data into Table: DEPARTMENTS
DB-22P04: com.edb.util.PSQLException: ERROR: extra data after last expected column

(continues on next page)
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.

The table definition (in the target database) does not match the migrated data. If you’ve altered the target or source table, confirm that the table definition and the data are compatible.

**ERROR: value too long for type**

*I’ve already migrated the table definition; when I try to migrate the data into the table, I get the following error:*

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)

A column in the target database is not large enough to receive the migrated data; this problem could occur if the table definition is altered after migration. The column name (in our example, location_id) is identified in the line that begins with Where:.

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

To correct the problem, adjust the column size and re-try the migration.

9.2. Migration Errors 86
ERROR: Exception in thread:OutOfMemoryError

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

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

By default, the MySQL JDBC driver will fetch all of the rows that reside in a table into the client application (Migration Toolkit) in a single network round-trip. This behavior can easily exceed available memory for large tables.

To correct the problem, specify `-fetchSize 1` as a command line argument when you retry the migration.

9.2. Migration Errors
When Migration Toolkit encounters a problem, it issues a notification in the form of an error code and a message.

This chapter describes the error codes, messages, and their resolutions.

Each error code begins with the prefix MTK- followed by five digits. The first two digits denote the error class, which is a general classification of the error. The last three digits represent the specific error condition.

The following table lists the error classes.

<table>
<thead>
<tr>
<th>Error Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>Successful Completion</td>
</tr>
<tr>
<td>01</td>
<td>Information</td>
</tr>
<tr>
<td>02</td>
<td>Warning</td>
</tr>
<tr>
<td>03</td>
<td>General Error</td>
</tr>
<tr>
<td>0x</td>
<td>Reserved</td>
</tr>
<tr>
<td>10</td>
<td>Invalid User Input</td>
</tr>
<tr>
<td>11</td>
<td>Configuration Error</td>
</tr>
<tr>
<td>12</td>
<td>Insufficient Privileges</td>
</tr>
<tr>
<td>13</td>
<td>Unsupported Feature</td>
</tr>
<tr>
<td>14</td>
<td>Missing Object</td>
</tr>
<tr>
<td>15</td>
<td>Schema Migration</td>
</tr>
<tr>
<td>16</td>
<td>Procedural Language Migration</td>
</tr>
<tr>
<td>17</td>
<td>Data Loading</td>
</tr>
</tbody>
</table>
If there is an error reported back by a specific database server, this error message is prefixed with `DB-`. For example, if table creation fails due to an existing table in a Postgres database server, the error code `42P07` is returned by the database server. The specific error in the Migration Toolkit log appears as `DB-42P07`.

# 10.1 Error Code Summary

The following sections summarize the Migration Toolkit error codes. In the following tables, column Error Code lists the Migration Toolkit error codes. The Message and Resolution column contains the message displayed with the error code. The message explains the cause of the error and how it is to be resolved.

In the Message and Resolution column, `$NAME` is a placeholder for information that is substituted at run time with the appropriate value.

## 10.1.1 Class 02 - Warning

This class represents the warning messages.

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Message and Resolution</th>
</tr>
</thead>
<tbody>
<tr>
<td>MTK-02000</td>
<td>All the warnings that relate to this class and do not have a specific error code binding will use this error code.</td>
</tr>
<tr>
<td>MTK-02001</td>
<td>Run <code>runMTK -help</code> to see the usage details.</td>
</tr>
<tr>
<td>MTK-02002</td>
<td>Warning! The offline migration path <code>$OFFLINE_PATH</code> does not exist, the scripts will be created under the user home folder.</td>
</tr>
</tbody>
</table>

## 10.1.2 Class 10 - Invalid User Input

This class represents invalid user input values.

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Message and Resolution</th>
</tr>
</thead>
<tbody>
<tr>
<td>MTK-10000</td>
<td>All the errors that relate to this class and do not have a specific error code binding will use this error code.</td>
</tr>
<tr>
<td>MTK-10001</td>
<td>You cannot select <code>information_schema</code>, <code>dbo</code>, <code>sys</code>, or <code>pg_catalog</code> as target schemas. These are used to store metadata information in <code>$DATABASE</code>.</td>
</tr>
<tr>
<td>MTK-10002</td>
<td>The <code>-schemaOnly</code> and <code>-dataOnly</code> options cannot be specified at the same time.</td>
</tr>
<tr>
<td>MTK-10003</td>
<td>The <code>&quot;\t&quot;</code> is required as a <code>copyDelimiter</code>, when using <code>escapeTabDelimiter</code> option.</td>
</tr>
<tr>
<td>MTK-10004</td>
<td>The -truncLoad option can only be given with the -dataOnly option.</td>
</tr>
</tbody>
</table>

continues on next page
### Table 1 – continued from previous page

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Message and Resolution</th>
</tr>
</thead>
<tbody>
<tr>
<td>MTK-10005</td>
<td>The <code>-dataOnly</code> option is applicable only for <code>-allTables/-tables</code> option. Schema DDL cannot be copied when this option is in place.</td>
</tr>
<tr>
<td>MTK-10006</td>
<td>The <code>-constraints</code>, <code>-indexes</code> and <code>-triggers</code> options are applicable only in the context of <code>-allTables/-tables</code> option.</td>
</tr>
<tr>
<td>MTK-10007</td>
<td>The <code>'-vacuumAnalyze'</code> and <code>'-analyze'</code> options cannot be specified at the same time.</td>
</tr>
<tr>
<td>MTK-10008</td>
<td>The <code>-skipFKConst</code> option can only be given with <code>-constraints</code> option.</td>
</tr>
<tr>
<td>MTK-10009</td>
<td>The <code>-skipCKConst</code> option can only be given with <code>-constraints</code> option.</td>
</tr>
<tr>
<td>MTK-10010</td>
<td>The <code>-fastCopy</code> option cannot be specified with <code>-schemaOnly</code> option.</td>
</tr>
<tr>
<td>MTK-10011</td>
<td>The <code>-skipColDefaultClause</code> cannot be specified with <code>-dataOnly</code> option.</td>
</tr>
<tr>
<td>MTK-10012</td>
<td>The <code>'-customColTypeMapping'</code> and <code>'-customColTypeMappingFile'</code> options cannot be specified at the same time.</td>
</tr>
<tr>
<td>MTK-10013</td>
<td>Provided default date time must be in following format ‘yyyyMMdd_HH:mm:ss’. Time portion is optional, to specify time, the underscore symbol ‘_’ is necessary.</td>
</tr>
<tr>
<td>MTK-10014</td>
<td>You cannot select MySQL and Sybase as the target database for migration.</td>
</tr>
<tr>
<td>MTK-10015</td>
<td>Copy delimiter should be a single character.</td>
</tr>
<tr>
<td>MTK-10017</td>
<td>Invalid number for batch size, use a number from 1-50000.</td>
</tr>
<tr>
<td>MTK-10018</td>
<td>Copy Batch Size should be greater than 0.</td>
</tr>
<tr>
<td>MTK-10019</td>
<td>Invalid number for Copy Batch Size, use a number &gt; 0.</td>
</tr>
<tr>
<td>MTK-10020</td>
<td>Fetch Size should be greater than 0.</td>
</tr>
<tr>
<td>MTK-10021</td>
<td>Invalid number for Fetch Size, use a number &gt; 0.</td>
</tr>
<tr>
<td>MTK-10022</td>
<td>One or more of the command-line arguments is invalid.</td>
</tr>
<tr>
<td>MTK-10023</td>
<td>The <code>-targetDBVersion</code> value should be specified using <code>major.minor</code> pattern.</td>
</tr>
<tr>
<td>MTK-10024</td>
<td>The <code>-targetDBVersion</code> can only be used with <code>-offlineMigration</code> mode.</td>
</tr>
<tr>
<td>MTK-10025</td>
<td>Options (<code>-constraints</code></td>
</tr>
<tr>
<td>MTK-10026</td>
<td>The <code>-allSchemas</code> option should be specified as the last option in the command-line options list.</td>
</tr>
<tr>
<td>MTK-10027</td>
<td>You have specified invalid command-line arguments. Run <code>runMTK -help</code> to see the usage details.</td>
</tr>
<tr>
<td>MTK-10028</td>
<td>The <code>-replaceNullChar</code> cannot be specified with <code>-schemaOnly</code> option.</td>
</tr>
<tr>
<td>MTK-10029</td>
<td>The <code>-nullReplacementChar</code> can only be specified with <code>-replaceNullChar</code> option.</td>
</tr>
<tr>
<td>MTK-10030</td>
<td>The <code>-ignoreCheckConstFilter</code> can only be given with <code>-constraints</code> option.</td>
</tr>
<tr>
<td>MTK-10031</td>
<td>The <code>-enableConstBeforeDataLoad</code> can only be given with <code>-truncLoad</code> option.</td>
</tr>
<tr>
<td>MTK-10032</td>
<td>The <code>retryCount</code> value should be greater than 0.</td>
</tr>
<tr>
<td>MTK-10033</td>
<td>Invalid number for retryCount, use a number &gt; 0.</td>
</tr>
<tr>
<td>MTK-10034</td>
<td>The <code>loaderCount</code> value should be greater than 0.</td>
</tr>
</tbody>
</table>

continues on next page
### Table 1 – continued from previous page

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Message and Resolution</th>
</tr>
</thead>
<tbody>
<tr>
<td>MTK-10035</td>
<td>Invalid number for loaderCount, use a number &gt; 0.</td>
</tr>
<tr>
<td>MTK-10036</td>
<td>Cannot use singleDataFile option for offline data migration in COPY format.</td>
</tr>
<tr>
<td>MTK-10037</td>
<td>The offline data migration in COPY format is supported only when PostgreSQL or EnterpriseDB (EDB Postgres Advanced Server) is the target database.</td>
</tr>
<tr>
<td>MTK-10038</td>
<td>The ( $SHEMA ) cannot be used as schema name in ( $DATABASE ). Choose a different schema name via -targetSchema option.</td>
</tr>
<tr>
<td>MTK-10039</td>
<td>Log file size should be greater than 0.</td>
</tr>
<tr>
<td>MTK-10040</td>
<td>Log file count should be greater than or equal to 0.</td>
</tr>
<tr>
<td>MTK-10041</td>
<td>Offline migration can only be used with schema only option.</td>
</tr>
<tr>
<td>MTK-10042</td>
<td>The copyViaDBLinkOra option can only be used for copying data from Oracle to EnterpriseDB.</td>
</tr>
<tr>
<td>MTK-10043</td>
<td>The -recreateConst option can only be given with the -dataOnly option.</td>
</tr>
<tr>
<td>MTK-10044</td>
<td>The ( $DATABASE ) database type is not supported by Migration Toolkit. Specify a valid database type string (i.e., EnterpriseDB, Postgres, Oracle, SQLServer, Sybase, or MySQL).</td>
</tr>
<tr>
<td>MTK-10045</td>
<td>The URL specified for the ( $DATABASE ) database is invalid. Check the connectivity credentials.</td>
</tr>
<tr>
<td>MTK-10046</td>
<td>The -escapeKeywords value may be true or false.</td>
</tr>
<tr>
<td>MTK-10047</td>
<td>The -useOraCase option can only be used for migration from Oracle.</td>
</tr>
<tr>
<td>MTK-10048</td>
<td>The Postgres exported snapshot id is invalid.</td>
</tr>
<tr>
<td>MTK-10049</td>
<td>The URL specified for the Oracle database is not supported by dblink_ora. Check the connectivity credentials and provide a valid URL.</td>
</tr>
<tr>
<td>MTK-10050</td>
<td>The schema ( $SHEMA ) not found on source database.</td>
</tr>
<tr>
<td>MTK-10051</td>
<td>The skipUserSchemaCreation option can only be used for user migration from Oracle to EnterpriseDB.</td>
</tr>
<tr>
<td>MTK-10052</td>
<td>The -truncLoad option cannot be used with offline data migration.</td>
</tr>
</tbody>
</table>

### 10.1.3 Class 11 - Configuration Issues

This class represents invalid configuration settings provided in the toolkit.properties file or in any other configuration file used by the Migration Toolkit.
### 10.1.4 Class 12 - Insufficient Privileges

This class represents insufficient privilege errors for loading database user/role information.

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Message and Resolution</th>
</tr>
</thead>
<tbody>
<tr>
<td>MTK-12000</td>
<td>All the errors that relate to this class and do not have a specific error code binding will use this error code.</td>
</tr>
<tr>
<td>MTK-12001</td>
<td>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.</td>
</tr>
<tr>
<td>MTK-12002</td>
<td>The migration of privileges failed due to insufficient privileges. Grant the user SELECT privilege on the following Oracle catalog: dba_tab_privs.</td>
</tr>
</tbody>
</table>
10.1.5 Class 13 - Unsupported Features

This class represents errors related with the migration of unsupported objects and clauses. Either the target database has not provided the implementation for the given object, or the Migration Toolkit does not handle its migration.
<table>
<thead>
<tr>
<th>Error Code</th>
<th>Message and Resolution</th>
</tr>
</thead>
<tbody>
<tr>
<td>MTK-13000</td>
<td>All the errors that relate to this class and do not have a specific error code binding will use this error code.</td>
</tr>
<tr>
<td>MTK-13001</td>
<td>Skipping index as Oracle does not allow multiple indexes against same column list.</td>
</tr>
<tr>
<td>MTK-13002</td>
<td>For $DATABASE views migration is not supported.</td>
</tr>
<tr>
<td>MTK-13003</td>
<td>For $DATABASE roles migration is not supported.</td>
</tr>
<tr>
<td>MTK-13004</td>
<td>You cannot migrate triggers, sequences, procedures, functions, packages and synonyms for MySQL, SQL Server, and Sybase databases.</td>
</tr>
<tr>
<td>MTK-13005</td>
<td>You cannot migrate sequences and packages for SQL Server database.</td>
</tr>
<tr>
<td>MTK-13006</td>
<td>Given trigger is not migrated, the trigger is owned by a different user schema and cross-schema triggers are not supported by EnterpriseDB.</td>
</tr>
<tr>
<td>MTK-13007</td>
<td>The given trigger is not migrated, the trigger has WHEN clause which is not supported by EnterpriseDB.</td>
</tr>
<tr>
<td>MTK-13008</td>
<td>Skipping Database Link $DATABASE_LINK. EnterpriseDB currently does not support this type of Database Link.</td>
</tr>
<tr>
<td>MTK-13009</td>
<td>Warning! Skipping migration of trigger $TRIGGER, currently non-table triggers are not supported in target database.</td>
</tr>
<tr>
<td>MTK-13010</td>
<td>You cannot migrate procedures, packages, synonyms and database links to PostgreSQL database.</td>
</tr>
<tr>
<td>MTK-13011</td>
<td>Domain objects are not supported in target database.</td>
</tr>
<tr>
<td>MTK-13012</td>
<td>Rules are not supported in $DATABASE database.</td>
</tr>
<tr>
<td>MTK-13013</td>
<td>The database type is not supported.</td>
</tr>
<tr>
<td>MTK-13015</td>
<td>$TYPE is Not Supported by COPY.</td>
</tr>
<tr>
<td>MTK-13016</td>
<td>The migration to PostgreSQL is supported only when Oracle or PostgreSQL is the source database.</td>
</tr>
<tr>
<td>MTK-13017</td>
<td>Groups are not supported in $DATABASE database.</td>
</tr>
<tr>
<td>MTK-13018</td>
<td>Profile migration is not supported in $SRC_DB to $TARGET_DB permutation.</td>
</tr>
<tr>
<td>MTK-13019</td>
<td>Cannot migrate unknown Profile $PROFILE.</td>
</tr>
<tr>
<td>MTK-13020</td>
<td>Profiles cannot be migrated to database version $VERSION.</td>
</tr>
<tr>
<td>MTK-13021</td>
<td>Password Profile verify function $MY_VERIFICATION_FUNCTION must be explicitly migrated.</td>
</tr>
<tr>
<td>MTK-13022</td>
<td>Advanced Queues migration is not supported from $SRC_DB database to $TARGET_DB database.</td>
</tr>
<tr>
<td>MTK-13023</td>
<td>Advanced Queues cannot be migrated to database version $VERSION.</td>
</tr>
<tr>
<td>MTK-13024</td>
<td>The INTERVAL partition is not supported in $DATABASE, the table will be migrated without INTERVAL definition.</td>
</tr>
<tr>
<td>MTK-13025</td>
<td>Warning! User profile migration is not supported in target database version $VERSION. The profile “$PROFILE” for user “$USER” will be skipped.</td>
</tr>
<tr>
<td>MTK-13026</td>
<td>Object Type migration is not supported from $SRC_DB to $TARGET_DB.</td>
</tr>
</tbody>
</table>
10.1.6 Class 14 - Missing Objects

This class represents failures to find metadata information in the source database.

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Message and Resolution</th>
</tr>
</thead>
<tbody>
<tr>
<td>MTK-14000</td>
<td>All the errors that relate to this class and do not have a specific error code binding will use this error code.</td>
</tr>
<tr>
<td>MTK-14001</td>
<td>One or more tables are missing from the source $DATABASE database.</td>
</tr>
<tr>
<td>MTK-14002</td>
<td>One or more tables couldn’t be found in the source $DATABASE database. With -tables mode, the table name should be in uppercase unless it is case-sensitive.</td>
</tr>
<tr>
<td>MTK-14003</td>
<td>One or more users couldn’t be found in the source $DATABASE database. With -users mode, the user name should be in uppercase unless it is case-sensitive.</td>
</tr>
</tbody>
</table>

10.1.7 Class 15 - Schema Migration

The class represents migration issues related to non-procedural database objects such as tables, constraints, indexes, synonyms, views, users, and roles.
<table>
<thead>
<tr>
<th>Error Code</th>
<th>Message and Resolution</th>
</tr>
</thead>
<tbody>
<tr>
<td>MTK-15000</td>
<td>All the errors that relate to this class and do not have a specific error code binding will use this error code.</td>
</tr>
<tr>
<td>MTK-15001</td>
<td>Error creating constraint $CONSTRAINT.</td>
</tr>
<tr>
<td>MTK-15002</td>
<td>Error creating role $ROLE.</td>
</tr>
<tr>
<td>MTK-15003</td>
<td>Error granting given privilege $PRIVILEGE on $OBJECT to $USER.</td>
</tr>
<tr>
<td>MTK-15004</td>
<td>Error granting $ROLE to $USER.</td>
</tr>
<tr>
<td>MTK-15005</td>
<td>Error granting privilege $REASON.</td>
</tr>
<tr>
<td>MTK-15006</td>
<td>Error creating user $USER.</td>
</tr>
<tr>
<td>MTK-15007</td>
<td>Error creating index $INDEX.</td>
</tr>
<tr>
<td>MTK-15008</td>
<td>Error creating view $VIEW.</td>
</tr>
<tr>
<td>MTK-15009</td>
<td>Error creating materialized view $MVIEW.</td>
</tr>
<tr>
<td>MTK-15010</td>
<td>Error creating public synonym $SYNONYM.</td>
</tr>
<tr>
<td>MTK-15011</td>
<td>Error creating private synonym $SYNONYM.</td>
</tr>
<tr>
<td>MTK-15012</td>
<td>Error creating sequence $SEQUENCE.</td>
</tr>
<tr>
<td>MTK-15013</td>
<td>Error creating table $TABLE.</td>
</tr>
<tr>
<td>MTK-15014</td>
<td>Error creating database link $DATABASE_LINK.</td>
</tr>
<tr>
<td>MTK-15015</td>
<td>Error creating given object type $OBJECT_TYPE.</td>
</tr>
<tr>
<td>MTK-15016</td>
<td>The table $TABLE could not be created in $DATABASE database.</td>
</tr>
<tr>
<td>MTK-15017</td>
<td>The linked schema $LINKED_SCHEMA doesn’t exist in the target database. Create the schema and then retry.</td>
</tr>
<tr>
<td>MTK-15018</td>
<td>Error creating domain $DOMAIN.</td>
</tr>
<tr>
<td>MTK-15019</td>
<td>Error creating custom data type $DATA_TYPE.</td>
</tr>
<tr>
<td>MTK-15020</td>
<td>Error creating membership for group $GROUP.</td>
</tr>
<tr>
<td>MTK-15021</td>
<td>Table name $TABLE does not have a schema qualifier. With multiple schema migration context, each table should be schema qualified.</td>
</tr>
<tr>
<td>MTK-15022</td>
<td>Schema qualifier $SCHEMA does not match the schema list.</td>
</tr>
<tr>
<td>MTK-15023</td>
<td>The table metadata information is not available.</td>
</tr>
<tr>
<td>MTK-15024</td>
<td>Tables list is not initialized yet.</td>
</tr>
<tr>
<td>MTK-15025</td>
<td>Error while getting database metadata information.</td>
</tr>
<tr>
<td>MTK-15026</td>
<td>Error creating group $GROUP.</td>
</tr>
<tr>
<td>MTK-15027</td>
<td>Error creating Profile $PROFILE.</td>
</tr>
</tbody>
</table>

10.1. Error Code Summary
10.1.8 Class 16 - Procedural Language Migration

The class represents migration issues related to database objects that are based on procedural languages such as procedures, functions, packages, anonymous blocks, triggers, and rules.

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Message and Resolution</th>
</tr>
</thead>
<tbody>
<tr>
<td>MTK-16000</td>
<td>All the errors that relate to this class and do not have a specific error code binding will use this error code.</td>
</tr>
<tr>
<td>MTK-16001</td>
<td>Error Creating Trigger $TRIGGER.</td>
</tr>
<tr>
<td>MTK-16002</td>
<td>Error Creating Rule $RULE.</td>
</tr>
<tr>
<td>MTK-16003</td>
<td>Error Creating Package Spec $PACKAGE.</td>
</tr>
<tr>
<td>MTK-16004</td>
<td>Error Creating Package Body $PACKAGE.</td>
</tr>
<tr>
<td>MTK-16005</td>
<td>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.</td>
</tr>
<tr>
<td>MTK-16006</td>
<td>Error Creating Procedure $PROCEDURE.</td>
</tr>
<tr>
<td>MTK-16007</td>
<td>Error Creating Function $FUNCTION.</td>
</tr>
<tr>
<td>MTK-16008</td>
<td>Error Creating Anonymous Block $BLOCK.</td>
</tr>
<tr>
<td>MTK-16009</td>
<td>Error creating Queue $QUEUE.</td>
</tr>
</tbody>
</table>

10.1.9 Class 17 - Data Loading

This class represents errors that may occur while copying data from the source to the target database.

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Message and Resolution</th>
</tr>
</thead>
<tbody>
<tr>
<td>MTK-17000</td>
<td>All the errors that relate to this class and do not have a specific error code binding will use this error code.</td>
</tr>
<tr>
<td>MTK-17001</td>
<td>Error Loading Data into Table: $TABLE</td>
</tr>
<tr>
<td>MTK-17002</td>
<td>Encoding Conversion encountered some characters that will be loaded using Bulk Inserts instead.</td>
</tr>
<tr>
<td>MTK-17003</td>
<td>Error in copy tables $REASON.</td>
</tr>
<tr>
<td>MTK-17004</td>
<td>Invalid Data Type.</td>
</tr>
<tr>
<td>MTK-17005</td>
<td>This Table Contains CLOB data, Marked for Bulk Insert Loading.</td>
</tr>
</tbody>
</table>
EDB Postgres Advanced Server offers complete support for some Oracle features and partial support for others. Migration Toolkit cannot migrate any object that uses an unsupported feature.

In some cases, Migration Toolkit can migrate objects that use features that offer partial compatibility. In other cases, EDB Postgres Advanced Server supports suitable workarounds.

Full-text search is an example of functionality that is not fully compatible with Oracle. The EDB Postgres Advanced Server database has included support for full-text search for quite some time, but the implementation is quite different than Oracle’s; Migration Toolkit is unable to migrate objects that utilize this feature.

There are also features that EDB Postgres Advanced Server does not yet support. Features in this category include Automated Storage Management, table compression, and external tables. You can often orchestrate a successful workaround:

- Automated Storage Management can be replaced with (system specific) volume management software.
- Table compression can be implemented by storing data in a tablespace that resides on a compressed filesystem.
- External tables don’t exist in EDB Postgres Advanced Server, but you can load flat text files into staging tables in the database. We recommend using the EDB*Loader utility to load the data into an EDB Postgres Advanced Server database quickly.
- When migrating multiple profiles from an Oracle database into EDB Postgres Advanced Server, you must manually assign the profile to a user (or users) when the migration completes.
11.1 Unsupported Postgres Features

Migration Toolkit does not support migration of the following Postgres features:

- OPERATOR CLASS
- OPERATOR FAMILY

For information about OPERATOR CLASS and OPERATOR FAMILY, see the PostgreSQL core documentation available at:

https://www.postgresql.org/docs/current/indexes-opclass.html
11.2 Frequently Asked Questions

Does Migration Toolkit support the migration of packages?

Migration Toolkit supports the migration of packages from an Oracle database into EDB Postgres Advanced Server. See the Functionality Overview Section for information about the migration support offered by EDB Postgres Advanced Server.

Is there a way to transfer only the data from the source database?

Yes. Data transfer is supported as part of an online or offline migration.

Does Migration Toolkit support migration of tables that contain data of the CLOB data type?

Migration Toolkit does support migration of tables containing data of the CLOB type.

Does EDB Postgres Advanced Server support the enum data type?

EDB Postgres Advanced Server does not currently support the enum data type, but will support them in future releases. Until then, you can use a check constraint to restrict the data added to an EDB Postgres Advanced Server database. A check constraint defines a list of valid values that a column may take.

The following code sample includes a simple example of a check constraint that restricts the value of a column to one of three dept types; sales, admin or technical.

```sql
CREATE TABLE emp (
    emp_id INT NOT NULL PRIMARY KEY,
    dept VARCHAR(255) NOT NULL,

    CHECK (dept IN ('sales', 'admin', 'technical'))
);
```

If we test the check constraint by entering a valid dept type, the INSERT statement works without error:

```sql
test=# INSERT INTO emp VALUES (7324, 'sales');
INSERT 0 1
```

If we try to insert a value not included in the constraint (support), EDB Postgres Advanced Server throws an error:

```sql
test=# INSERT INTO emp VALUES (7325, 'support');
ERROR: new row for relation "emp" violates check constraint "emp_dept_check"
```

Does EDB Postgres Advanced Server support materialized views?
Postgres does not support materialized views compatible with Oracle databases. To setup a materialized view/summary table in Postgres you must manually create the triggers that maintain the summary table. Automatic query rewrite is not currently supported; the application must be made aware of the summary table’s existence.

**When I try to migrate from a MySQL database that includes a TIME data type, I get the following error: Error Loading Data into Table: Bad format for Time. Does Postgres support MySQL ``TIME`` data types?**

Postgres will have no problem storing `TIME` data types as long as the value of the hour component is not greater than 24.

Unlike Postgres, the MySQL `TIME` data type will allow you to store a value that represents either a `TIME` or an `INTERVAL` value. A value stored in a MySQL `TIME` column that represents an `INTERVAL` value could potentially be out of the accepted range of a valid Postgres `TIMESTAMP` value. If, during the migration process, Postgres encounters a value stored in a `TIME` data column that it perceives as out of range, it will return an error.
Conclusion

EDB Migration Toolkit Guide

Copyright © 2007 - 2021 EnterpriseDB Corporation.

All rights reserved.

EnterpriseDB® Corporation 34 Crosby Drive, Suite 201, Bedford, MA 01730, USA

T +1 781 357 3390 F +1 978 467 1307 E info@enterprisedb.com www.enterprisedb.com

- EnterpriseDB and Postgres Enterprise Manager are registered trademarks of EnterpriseDB Corporation. EDB and EDB Postgres are trademarks of EnterpriseDB Corporation. Oracle is a registered trademark of Oracle, Inc. Other trademarks may be trademarks of their respective owners.

- EDB designs, establishes coding best practices, reviews, and verifies input validation for the logon UI for EDB Postgres products where present. EDB follows the same approach for additional input components, however the nature of the product may require that it accepts freeform SQL, WMI or other strings to be entered and submitted by trusted users for which limited validation is possible. In such cases it is not possible to prevent users from entering incorrect or otherwise dangerous inputs.

- EDB reserves the right to add features to products that accept freeform SQL, WMI or other potentially dangerous inputs from authenticated, trusted users in the future, but will ensure all such features are designed and tested to ensure they provide the minimum possible risk, and where possible, require superuser or equivalent privileges.

- EDB does not warrant that we can or will anticipate all potential threats and therefore our
process cannot fully guarantee that all potential vulnerabilities have been addressed or considered.
<table>
<thead>
<tr>
<th>Index</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>B</strong></td>
</tr>
</tbody>
</table>
| **C** | **CentOS Installation, 12**  
**CentOS ppcle Installation, 16**  
**Conclusion, 102**  
**Configuring IDENT Authentication, 14**  
**Connection Errors, 81** |
| **D** | **Debian Installation, 18**  
**Defining a MySQL URL, 48**  
**Defining a PostgreSQL URL, 44**  
**Defining a SQL Server URL, 50**  
**Defining a Sybase URL, 49**  
**Defining an Advanced Server URL, 42**  
**Defining an Oracle URL, 46** |
| **E** | **Error Codes, 88**  
**Executing Offline Migration Scripts, 60** |
| **F** | **FAQ, 100**  
**Functionality Overview, 9** |
| **I** | **Import Options, 61** |
| **L** | **Linux Installation, 12** |
| **M** | **MacOSX Installation, 33**  
**Migration Errors, 81, 84**  
**Migration Methodology, 5**  
**Migration Options, 66**  
**Migration Toolkit Command Options, 57** |
| **O** | **Offline Migration Options, 59**  
**Oracle Specific Options, 70** |
| **R** | **RHEL Installation, 12**  
**RHEL ppcle Installation, 16** |
| **S** | **Schema Creation Options, 62**  
**Schema Object Selection Options, 63**  
**SLES Installation, 20**  
**Supported Operating Systems and Database Versions, 3** |
| **U** | **Ubuntu Installation, 18** |
Unsupported Features, 98
Unsupported PostgreSQL Features, 99
Upgrading an RPM Installation, 16

W
What's New, 2
Windows Installation, 22