Oracle® Database Migration Assistant for Unicode

User's Guide

Release 23.1
F77360-01
June 2024

ORACLE®

Oracle Database Migration Assistant for Unicode User's Guide, Release 23.1

F77360-01

Contents

Preface

1   Overview of Migrating Databases to Unicode

ORACLE

2   Getting Started with the DMU

3   Viewing and Setting Object Properties in the DMU

4   Performing Basic DMU Tasks

5   Advanced Topics in the DMU

6   Using the DMU to Cleanse Data

Preface

This guide introduces you to the Oracle Database Migration Assistant for Unicode (DMU).

This preface contains the following topics:

- Audience
- Documentation Accessibility
- Diversity and Inclusion
- Related Documents
- Conventions

Audience

This guide is intended for database administrators who perform the following tasks:

- Migrate character sets to Unicode
- Verify that there are no character set data problems in a current Unicode database

To use this guide, you must be familiar with Oracle databases.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Diversity and Inclusion

Oracle is fully committed to diversity and inclusion. Oracle respects and values having a diverse workforce that increases thought leadership and innovation. As part of our initiative to build a more inclusive culture that positively impacts our employees, customers, and partners, we are working to remove insensitive terms from our products and documentation. We are also mindful of the necessity to maintain compatibility with our customers' existing technologies and the need to ensure continuity of service as Oracle's offerings and industry standards evolve. Because of these technical constraints, our effort to remove insensitive terms is ongoing and will take time and external cooperation.

Related Documents

For more information, see these Oracle resources:

- *Oracle Database Globalization Support Guide*

Many of the examples in this guide use sample schemas, which are installed by default when you select the Basic Installation option with an Oracle Database installation. See *Oracle Database Sample Schemas* for information about how these schemas were created and how you can use them.

To download free release notes, installation documentation, white papers, or other collateral, visit the Oracle Help Center (OHC). You must register online before using OHC; registration is free and can be done at:

http://docs.oracle.com

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

1

Overview of Migrating Databases to Unicode

This chapter discusses the process of database character set migration.

Besides explaining what a database character set is and what it means to migrate one, the chapter discusses the basic issues related to migration and the tools available to help with these tasks.

1.1 What Is a Database Character Set?

A **database character set** defines a set of characters that can be stored in the database and defines how each character maps to a sequence of bytes stored in memory or disk to represent this character.

A character set is also known in the industry as a **character encoding**. You can think of a character encoding as a set of characters plus their encoding in bytes.

A database character set is used for the following:

- `VARCHAR2`

- `CHAR`

- `LONG`

- `CLOB`

- SQL, PL/SQL, and Java stored code in the database

The `CLOB` data is stored in the database character set only if the character set is single-byte. Otherwise, it is stored in AL16UTF16, which is Unicode UTF-16 encoding in big-endian form, abbreviated as UTF-16BE.

A database character set has either of the following characteristics:

- Single-byte

  Each character has one byte. Single-byte character sets are easier and faster to process but have a limited capability of 256 codes, which is too few even for all European languages.

- Multibyte

  Each character may be 1, 2, 3, or 4 bytes. The maximum depends on the particular character set. Multibyte character sets require more complex string processing but can accommodate thousands of characters.

Almost all character sets other than Unicode, which are frequently called **legacy character sets**, were designed for a specific language or group of languages. For single-byte character sets, this design was logical because of the limited number of available codes. For multibyte character sets, this was because of limited requirements of isolated local databases, the desire to limit the maximum required width of a character code, or expertise required to design a character set for multiple unrelated language groups.

If character set A contains or defines all characters of set B plus some other characters, then A is a superset of B, and B is a subset of A. If A contains or defines all characters of B, and if

each character has identical byte representation in both A and B, then A is a strict or binary superset of B, and B is a strict or binary subset of the A. A binary subset or superset is also called a strict subset or superset.

When character data is transmitted from an environment where it is encoded in one character set to an environment where it is encoded in another character set, for example, from one database to another or from client to server, the data must undergo character set conversion. The code of a character in the source character set must be replaced with the code of the same character in the target.

Characters that are not available in the target character set are converted to replacement characters. The replacement of a given source character may be defined explicitly in the target character set. For example, ä (a with an umlaut) is replaced by a when converted to the character set US7ASCII. If no explicit replacement character is defined, then the default replacement character of the target character set is used. This character is usually the question mark ? or the inverted question mark ¿. The default replacement character in Unicode character sets is the character U+FFFD.

By definition, if data is converted from a character set to its superset, no replacement characters are used because all characters are available in the superset. Also, if data is converted from a character set to its binary (strict) superset, no character codes actually change. Oracle Database usually performs the conversion even to a binary superset. Consequently, if the text to be converted contains byte sequences that do not form valid character codes in the source character set, then the database converts these sequences to the replacement characters in the target character set.

1.2 Introduction to Character Set Migration

A database defined in a legacy character set can store data only in the languages determined by its character set. To store data in other languages, the character set must be changed to a superset that defines both the existing and new required characters.

You may have various reasons to migrate character sets, including the following:

- Internationalization of operations

- Moving to an online sales model

- Consolidating databases of international subsidiaries or acquired companies

If the superset is a binary superset, then the database character set declaration must change only in metadata (schema information) in the data dictionary. Existing data remains valid in the binary superset. The change is simple and fast. If the identified superset is not a binary superset, however, then binary character codes (byte representation) must be re-encoded. The representation in the old character set must be converted to the representation in the new character set.

Even if the old database character set is not a binary subset of the new character set, many characters may have the same binary codes. In particular, standard ASCII characters with codes between 0 and 127 form a binary subset of every Oracle database character set supported on ASCII-based platforms (that is, all platforms except EBCDIC-based IBM z/OS and Fujitsu BS2000). For example, if WE8ISO8859P1 must change to AL32UTF8 (which are not in a binary subset-superset relationship), and if all characters have codes between 0 and 127, then no conversion is necessary. All characters will have the same codes after the change.

When no conversion is needed for the database character data to be converted, it is called **changeless**. Data that requires conversion is called **convertible**. You can save significant resources by skipping the conversion of changeless data.

When character codes are converted, issues may arise. Most commonly, the new representation may have more bytes, causing the converted value to no longer satisfy the containing column's length constraint or even the maximum length of its data type. For conversion to succeed, you must identify and clean up these issues. For example, you may need to lengthen columns, shorten text, or migrate to a larger data type.

Character set migration is a process that includes the following tasks:

- Identifying the suitable superset

- Classifying character data as changeless or convertible

- Checking for issues and conversion requirements

- Fixing any issues

- Changing the declaration in the data dictionary and converting data as needed

You can identify the issues and the subset/superset relationship of existing data in the database with the help of the Database Character Set Scanner utility or the Database Migration Assistant for Unicode (DMU).

You can perform an **A-to-B migration**, where data is copied from a source database into a new database defined with the target character set. Alternatively, you can perform an **inline migration**, where data is converted inside a database and the character set information is changed.

The character set migration is distinct from the repair process of updating the database character set declaration to match the encoding of data in the database. In certain incorrect configurations of Oracle client software, known as **pass-through** or garbage-in garbage-out configurations, the character data in the database may be interpreted by applications in a different character set from the one declared for the database. The repair goal is to match the database character set declaration with the expectations of the applications. This type of repair does not involve conversion of character codes.

> ✎ **See Also:**
>
> *Oracle Database Globalization Support Guide* to learn more about the Database Character Set Scanner utility

1.3 Why Unicode Is the Right Choice

Oracle recommends migrating to Unicode for its universality and compatibility with contemporary and future technologies and language requirements.

An important step in the character set migration process is gathering requirements for what you need to achieve. A typical situation is a company that is expanding worldwide and must support data storage of languages unavailable with the existing character set. Because many legacy systems initially supported only one or possibly a few languages, the original character set might have had a limited repertoire of supported characters. Common cases were America, where 7-bit ASCII was satisfactory for supporting English data exclusively, and Europe, where a variety of 8-bit character sets supported European languages plus English. These choices were reasonable and provided a balance of economy and performance.

Today, a limited character set can be a barrier to supporting the global market, thus necessitating character set migration. It is common to need to consolidate multiple servers into

a single instance, with the existing servers having various character sets. In this case, Unicode is the logical standard.

The character set defined in the Unicode standard supports all contemporary written languages with significant use and a few historical scripts. It also supports symbols such as those used in technical, scientific, and musical notations. Unicode is the native or recommended character set of many technologies, such as Java, Windows, HTML, or XML. There is no character set more universal than Unicode. Adoption of Unicode is increasing rapidly with great support from within the industry.

Oracle supports two encodings of Unicode as the database character set:

*   UTF-8 through the AL32UTF8 character set

    UTF-8 is a multibyte varying width Unicode encoding using 1 to 4 bytes per character. AL32UTF8 encodes ASCII characters in 1 byte, characters from European, and Middle East languages in 2 bytes, characters from South and East Asian languages in 3 bytes. Therefore, storage requirements of Unicode are usually higher than storage requirements of a legacy character set for the same language. This is valid for all languages except those requiring only ASCII (English, Dutch, Indonesian, Swahili, and so on).

*   CESU-8 through the UTF8 character set

    CESU-8 is a compatibility-only encoding, discouraged for information exchange by the Unicode standard. CESU-8 is similar to UTF-8 except that so-called supplementary characters encoded by UTF-8 in 4 bytes, CESU-8 encodes as pairs of 3-byte codes. Oracle Database has deprecated the UTF8 character set, so it should not be used as a database character set except when explicitly required by an application, such as Oracle E-Business Suite Release 11i.

> **✎ Note:**
>
> A hyphen appears in the name of Unicode encoding (for example, UTF-8) and does not appear in the name of Oracle character set (for example, UTF8). This differentiation is used throughout Oracle documentation.

When looking for a superset to satisfy business requirements, you might find that there is a legacy character set that fulfills those requirements. This could happen, for example, if the database is US7ASCII storing English, which can also be stored in any other Oracle database character set, and the new languages to be added to the database all belong to a single group of compatible languages, supported by some legacy character set. For example, all the new languages are Western European and can be stored in WE8MSWIN252. Another example is if a few new characters must be stored in the database, such as the Euro sign and *smart quotes* ("), and the new characters are supported by an extended variant of the existing database character set. An example of such an extended variant is WE8MSWIN1252, which is a single-byte binary superset of WE8ISO8859P1.

If more than one language group is to be stored, then Unicode is the only choice.

Migration from one legacy character set to another legacy character set should be considered only if all of the following statements are true for the environment to be migrated:

*   The new character set is a binary superset of the old character set, requiring only a simple declaration change, and no data conversion.

*   The new character sets is single-byte, thus offering better performance and more compact storage than Unicode.

- No future language requirements are expected in a foreseeable future that will not be compatible with the new character set.

- Migration to AL32UTF8 would require significant resources due to availability and/or compatibility requirements.

Two more reasons to consider when choosing the new character set are:

- If the database grows fast, a conversion in the future will be much more expensive than today. Therefore, as Unicode seems to be the ultimate goal in the industry, it is best to migrate to it as soon as possible.

- If the application is an Internet application, or it uses Java, or some other Unicode-based technology, then Unicode as the database character set will provide a more consistent, safer and more efficient environment, where there is no risk of losing data because of the inability of the system to store characters that may unexpectedly come from the application, and where character set conversions are limited to efficient transformations between Unicode character encodings.

Another Unicode encoding, UTF-16BE, named AL16UTF16 by Oracle Database, is used as the national character set for `NCHAR`, `NVARCHAR2`, and `NCLOB` data types. Data requiring the Unicode character set may be stored in columns of those data types, even if the database character set is not migrated to AL32UTF8. But those data types have some restrictions, for example, they are not supported by Oracle Text and require changes to client API calls (Oracle Call Interface (OCI) or Java Database Connectivity (JDBC)). Oracle does not recommend the use of national character set data types, and `NCHAR` data types are not discussed in this guide.

> **See Also:**
>
> *Oracle Database Globalization Support Guide* for more coverage of `NCHAR`

1.4 About Character Set Migration Tools

Oracle Database provides various tools to help you change the database character set of an existing database.

1.4.1 Identification of Migration Issues with the Database Character Set Scanner (CSSCAN)

The Database Character Set Scanner (`CSSCAN`) is a command-line utility that assesses the feasibility of migrating an Oracle database to a new database character set.

> **Note:**
>
> The Database Character Set Scanner (`CSSCAN`) is not available in Oracle Database 12c.

The Database Character Set Scanner checks all character data in the database and tests for the effects and problems of changing the character set encoding. A summary report is generated at the end of the scan that shows the scope of work required to convert the database to a new character set.

ORACLE

The Database Character Set Scanner is available with all database editions except the Express Edition. Most of its scanning functionality has been included in the Database Migration Assistant for Unicode, discussed in this guide, but the Database Character Set Scanner must still be used if any of the following is true:

- Migration is to a legacy character set.
- The national (NCHAR) character set is being migrated.
- The scan results are required by the CSALTER script.

1.4.2 Changing Character Set Metadata Using the CSALTER Script

The CSALTER script changes character set metadata in the data dictionary to the new character set. If the character set changes from single-byte to multibyte (or vice versa), then the script converts CLOB column data in the data dictionary tables.

> **Note:**
>
> The CSALTER script is part of the Database Character Set Scanner utility. CSALTER is not available in Oracle Database 12c.

The CSALTER script does not limit the data dictionary to SYS tables. The scripts also considers most schemas in default template databases, which are created during database installation, as the data dictionary.

You can run CSALTER only if no character data in the database, except CLOB data in the data dictionary tables, needs re-encoding to remain valid in the new character set. To assure the integrity of the database, CSALTER requires that you confirm that no data requires conversion by running Database Character Set Scanner. If the scan results contain information about "exceptional" data anywhere in the database or "convertible" data outside of the data dictionary CLOB columns, then CSALTER fails with an error.

CLOB columns in the data dictionary are treated specially because AL16UTF16, which stores CLOB data in multibyte databases, is never a binary superset or subset of any single-byte character set. Even if the target character set is a superset, CLOB data must be converted. Because data dictionary tables cannot be dropped or truncated, no single-byte database character set could be changed to any multibyte character set without CSALTER converting the data dictionary CLOB data.

The CLOB columns in user tables are not converted by CSALTER; the CSALTER script will not run if CLOB data is in user tables and the character set change is from single-byte to multibyte.

Most CSALTER functionality has been included in the Database Migration Assistant for Unicode, but you must still use CSALTER in the following situations:

- You migrate to a legacy character set.
- The national (NCHAR) character set is being migrated.

1.4.3 Converting Character Data Using the Export/Import and Data Pump Utilities

When the Database Character Set Scanner finds data requiring conversion, the `CSALTER` script cannot change the database character set. Oracle Data Pump solves this problem.

A common way to convert character data is to use the Export utility to export database objects, and then the Import utility to import them to the same or to another database. If the database character set differs during import and export, then the Import utility automatically converts the data as required. The advantage of using this technique, as opposed to techniques such as spooling data and loading it back with SQL*Loader, is that it automatically takes care of most steps needed to re-create exported objects in their original form.

> ✎ **See Also:**
>
> *Oracle Database Globalization Support Guide* for information about how to change the database character set using a full export and import process

1.4.3.1 Full Versus Selective Export/Import

You can migrate character sets either through full database export and export, or selective export and import.

The migration techniques are as follows:

- Migration through full database export and import

  In this technique, you first create an empty database using the target character set. Afterward, you export the contents of the source database, and then import it into the target database. The created copy of the original database has all character data converted as required. You drop the source database, making the copy the new production database.

- Migration through selective export and import

  In this technique, you export only tables identified by the Database Character Set Scanner utility as containing convertible data, and then truncate them. You run the Database Character Set Scanner again. This time, the scan finds no convertible data. You can now change the database character set by executing `CSALTER`. Finally, you import the previously exported data back into the database, undergoing the required character set conversion.

The following circumstances favor the full export/import method:

- The data dictionary of the database contains convertible data. For example, table and/or column names (identifiers) contain non-ASCII characters requiring conversion. As data dictionary tables cannot be truncated, the selective export/import approach cannot be used.

- The number of tables requiring conversion is high, making manual selection of tables to export and truncate a costly task.

- The character set migration process is combined with database upgrade to a higher release and/or migration to another platform.

- A lot of tables in the database are fragmented and the character set migration process is an opportunity to perform defragmentation and cleanup.

The following circumstances favor the selective export/import method:

- The full export/import converts all text in the database, more than may be needed. It also must insert all non-textual data. Therefore, the full export/import method is usually much too slow for the short maintenance windows of large production databases.

- The selective export/import method requires temporary disk space for the exported tables only. The full method requires temporary storage for the database copy and for the full database export file. This amounts to additional space almost twice the size of the source database.

1.4.3.2 Data Pump Utility

The Export and Import utilities come in two versions: original Export (`exp`) and Import (`imp`), and Data Pump Export (`expdp`) and Import (`impdp`).

The Data Pump versions were introduced in Oracle Database Release 10g. The original versions are desupported for general use as of Oracle Database Release 11g. You can use both versions for migration.

The original utilities are not compatible with Data Pump utilities. You cannot import files produced by `exp` using `impdp`, and you cannot import files produced by `expdp` using `imp`.

Original Export and Import are client-side utilities. Export retrieves data over an Oracle Net connection and creates the export file on the client. Data Pump Export contains a small client-control utility, but the export is performed by a database server process and the export file is produced on the server host. Similarly, original Import reads the export file on the client and inserts data over Oracle Net into the target database, whereas Data Pump reads the export file directly on the server host. Its architecture makes the Data Pump export and import process much faster, but if the export file must be created or read on a host other than the source, then the target system must have a dummy database installed.

> **See Also:**
>
> *Oracle Database Utilities* for detailed information about the original and Data Pump Export/Import utilities

1.4.3.3 Conversion Issues

Conversion is associated with possible issues, mainly involving length problems.

Those issues should have been identified by the Database Character Set Scanner and handled by the user before the import. Otherwise, in both techniques discussed, the import might fail.

When using the Export and Import, you must consider restrictions regarding objects not supported by the given utility version. For example, Oracle Data Pump Release 10g does not support exporting an XML schema. You might need to move such objects manually.

> **See Also:**
>
> *Oracle Database Globalization Support Guide* for more information about the database migration process with Export/Import utilities

1.4.4 Migrating a Database Using the Database Migration Assistant for Unicode

The Database Migration Assistant for Unicode (DMU) offers an intuitive and user-friendly GUI that helps you streamline the migration process.

The DMU interface minimizes the workload and ensures that all migration issues are addressed. The DMU guarantees that the data conversion is carried out correctly and efficiently.

Advantages of the DMU include the following:

- Guides you through the workflow

    An important advantage of the DMU is that it offers a logical workflow to guide you through the entire process of migrating character sets.

- Offers suggestions for handling certain problems

    The DMU can help you when you run into certain problems, such as errors or failures during the scanning or cleansing of the data.

- Supports selective conversion of data

    The DMU enables you to convert only the data that must be converted, at the table, column, and row level.

- Offers progress monitoring

    The DMU provides a GUI to visualize how the steps are progressing.

- Offers interactive visualization features

    The DMU enables you to analyze data and see the results in the GUI in an interactive way. It also enables you to see the data itself in the GUI and cleanse it interactively from identified migration issues.

- Provides the only supported tool for inline conversion

    With the DMU, Oracle Database supports inline conversion of database contents. This offers performance and security advantage over other existing conversion methods.

- Allows cleansing actions to be scheduled for later execution during the conversion step

    Postponing of cleansing actions, such as data type migration, ensures that the production database and applications are not affected until the actual migration downtime window.

This release of the Database Migration Assistant has a few restrictions with respect to what databases it can convert. For example, in most cases, it does not convert databases with convertible data in the data dictionary. The export/import migration methods could be used to overcome these limitations.

> **✎ See Also:**
>
> Performing Basic DMU Tasks for more information

1.5 Overview of Character Set Migration Considerations

Before migrating your current character set using the DMU, consider data integrity, dependent objects, downtime and failure recovery, and several other important issues.

1.5.1 Character Set Migration: Data Integrity

Several data integrity issues might arise from re-encoding character data, including data expansion, invalid binary representation, and several others.

1.5.1.1 Data Expansion

When you migrate from a legacy encoding to Unicode, character values may expand because their encodings have more bytes, resulting in increased space consumption.

Also, widths for `CHAR` and `VARCHAR2` columns may be insufficient after the migration. Thus, there is a risk of truncated data. The column length constraints must be increased. Alternatively, if the constraints are at the data type limit, you might need to migrate the columns to the `CLOB` type.

Non-ASCII object names have a related issue. Typically, these names have a 30-byte limit in releases before Oracle Database 12c Release 2 (12.2). In both the database and affected applications, you must manually truncate the names of such objects or change them to use only ASCII characters because it is impossible to increase the limit of non-ASCII object names.

> **Note:**
>
> Starting with Oracle Database 12c Release 2 (12.2), the maximum limit for an identifier is 128 bytes. For the earlier Oracle Database versions, the maximum limit for an identifier is 30 bytes.

1.5.1.2 Invalid Binary Storage Representation of Data

Column data may not be in the declared database character set: it may be binary, in another character set, or in multiple character sets.

Examples include column data that it is composed of images, documents in proprietary word processor formats, or text encrypted with custom methods.

In a *pass-through* configuration, the client character set is defined (usually through the `NLS_LANG` client setting) as equal to the database character set. The Oracle communication protocol does not attempt client/server character set conversion because conversion is assumed to be unnecessary. This configuration is correct when data from a client application is in the declared character set. But because no character set conversion or character validation is performed, any sequence of bytes can be stored and retrieved unchanged. If only the client application interprets the data, then the stored byte sequences could correspond to a character encoding different from the database character set, or not correspond to readable text at all.

If default character set conversion is applied to such incorrect data during migration, some of the data does not survive the conversion, and produces garbage. This results from the default application of a code mapping from the old database character set to the new character set to convert the stored sequences of bytes. This mapping is certainly incorrect for binary data,

ORACLE®

which should not change at all, and usually incorrect for data interpreted by applications as being in a character set different from the old database character set.

To avoid the problem, there are a few things you can do. One is to change the data type of the column to `RAW` or `LONG RAW`. This approach is recommended for binary data. Another possibility is to skip conversion of a column with suspected problems and let the application continue using the pass-through configuration after updating the client character set declaration to the new database character set. This approach is generally not recommended but might be the only choice if data in a single column is stored in multiple actual character sets and discovering these character sets automatically is not possible. Finally, the DMU enables you to declare the actual character set of any column as being different from the source database character set. In this way, you can tell DMU how to perform the conversion according to correct code mapping for data in the column.

1.5.1.3 Partitioning

To distribute rows, range partitioning uses the binary storage representation of character values to compare key values with boundary values. When converted to Unicode, the binary representation of these values could change, failing to preserve the ordering.

This lack of order preservation could lead to three problems:

1. A given partitioning key might not sort between the lower and the upper boundaries of the partition that the key is in.

2. Partitions are internally numbered in the same order as the order of their boundary values. If the boundary values change their sort order during conversion to Unicode, the internal numbering would become invalid.

3. In rare cases, two partition boundaries could become equal because in some multibyte Eastern Asian character sets two different characters may map to the same Unicode character. Unicode does not separately encode certain shape variants of certain Chinese characters, which are encoded in legacy character sets.

If its partitioning boundaries use non-ASCII characters, then a partitioned table might need to be re-created using modified DDL. If only the partitioning key values were affected, then rows would need to be moved between partitions, so row movement must be enabled for the table.

1.5.1.4 Maximum Index Key Size

In some cases, a large index key might exceed the maximum supported size.

The index key size is limited by the database block size. When you convert a database to Unicode, character values may expand because their encodings have more bytes. If an index is defined on a column that contains expanded data, its index key size may increase as follows:

- For databases in which the `COMPATIBLE` initialization parameter is set to a value of `20.0.0` or higher, the index key size is calculated using the maximum lengths of column data stored in the key columns. An increase in column data size may cause the index key size to increase beyond the maximum supported size.

- For all other databases, the index key size is calculated using the maximum lengths of key columns. To ensure that expanded data can reside in a column after conversion to Unicode, users may try to expand the column byte length. Alternatively, they may alter the column length semantics from `BYTE` to `CHAR`. The larger column lengths may cause the index key size to increase beyond the maximum supported size.

For all databases, you can try to solve this problem by:

- Increasing the database block size

- Dropping the index

- Moving the index to a tablespace with a larger block size

- Removing some columns from a composite index

For databases in which `COMPATIBLE` is set to a value of `20.0.0` or higher, you can also try to solve the problem by:

- Truncating data from the key columns

> **✎ Note:**
>
> Starting with DMU 23.1, you can view and resolve potential problems with index key sizes prior to converting a database. When scanning the database, you can instruct the DMU to perform an index check and generate an index check report. This report lists all indexes whose key sizes will exceed the maximum supported size as a result of column modifications required for the conversion or an increased post-conversion column data size. You can then fix the problematic indexes before proceeding with the database conversion. See "Database Scan Report: Index Check Report" for more details.

1.5.1.5 Unique Keys and Primary Keys

Uniqueness of the key values of a unique or primary constraint might be violated after conversion to Unicode, even though the preconversion key values satisfy the constraint.

The situations are as follows:

- Two different text values might become the same value after conversion to AL32UTF8 because characters in some character sets map to the same Unicode code point. For example, two JA16SJIS code points 0xED40 and 0xFA5C map to U+7E8A.

- Two text values differing only at their ends might become the same if, during conversion, they were truncated because of their post-conversion length exceeding the maximum column length.

Usually, the key values must be modified manually to solve these issues.

1.5.1.6 Derived or Encrypted Data

Applications might store information derived from character data that depends on its binary storage representation. Such derived values must be resynchronized after the migration.

For example, an application might store text length separately from numerical column text. If the text expands, then this column is not updated automatically. Alternatively, an application might calculate and store a checksum or an encrypted form of a character value. If the checksum or cipher text is based on the binary storage representation in the database character set, then it becomes invalid when the representation changes because of character set conversion.

For encrypted text, you must decrypt it just before the conversion, convert it, and then immediately re-encrypt it. Keep the time of the decrypted text to a minimum. To prevent a security breach, the database server might require extra protection.

Oracle Transparent Data Encryption functionality is generally transparent for the migration process. Manual decryption and re-encryption are not needed.

1.5.1.7 Character Data Stored in Binary Data Types

Applications might store character data in binary data types such as `RAW`, `LONG RAW`, and `BLOB`, and then use the database character set to interpret and process it. If the database character set is migrated but the data in the binary types is not re-encoded to match the new character set, then the applications fail.

Automatic migration tools such as the DMU have no information about which binary columns contain character data and might need character set conversion. Furthermore, character data may be intermixed with binary data in binary structures, so that simple conversion of entire column values is not possible. If required, you must manually locate and convert character data in binary data types.

1.5.2 Character Set Migration: Dependent Objects

Converting table data affects all objects that store a copy of it. Depending on the conversion method, you may need to drop and re-create dependent objects, or they will be synchronized automatically.

Dependent objects include indexes (standard, functional, and domain) and materialized views. Instead of relying on the automatic synchronization, performance may improve if you drop the dependent objects, and then re-create them after migration.

Triggers are also affected because they might react to modifications in a way that is not acceptable for business requirements. For example, a business rules requires that only user-initiated modifications are audited. However, a trigger sets the last modification date and the modifying user ID to the date and ID of the migration process. Therefore, you must disable triggers before converting the underlying table.

Most of the preceding issues are automatically handled by the DMU.

1.5.3 Character Set Migration: Read-Only and Inaccessible Objects

Read-only tables cannot be converted before they are made read/write.

Read-only tables are tables explicitly marked as read-only, external tables, and tables stored in read-only tablespaces. Except for the explicit marking of a table as read-only, which the migration process can temporarily remove, you must manage the other cases manually.

External tables do not usually require conversion. If the character set of the source file for an external table (SQL*Loader or Data Pump) is properly declared, then the database automatically converts character data to the new database character set when data is fetched from the file. To improve performance, consider converting the file permanently to avoid this conversion. If the character set of the source file is not defined correctly, and if the file content can be fetched only due to a pass-through configuration, similar to what is described in "Invalid Binary Storage Representation of Data", then you must correct the character set declaration of the file. For Data Pump files, this is not a trivial task.

Read-only tablespaces may be stored on read-only media. If convertible data is present in such a tablespace, then perform the following steps:

1. Move the tablespace to standard disk storage.

2. Make the tablespace read/write.

3. Convert the database.

4. Make the tablespace read-only again.

5. Copy it back to the read-only media.

Many such tablespaces may exist in a database. Making them read/write before conversion may not be feasible due to disk storage requirements. In this case, consider the following strategy:

1. Create a dedicated archival database in the old database character set.

2. Move the read-only tablespaces one-by-one to the new database using the transportable tablespace feature.

3. Migrate the original database.

4. Access the read-only data in the archival database through a database link. This takes care of the necessary character set conversion when the data is retrieved.

Tables may be stored in tablespaces or data files that have been put into the offline state. Data in such tables is not accessible. You can neither scan the data for migration issues nor actually convert it. Before the character set migration can begin, you must bring the offline tablespaces and data files containing tables with character columns online.

1.5.4 Character Set Migration: Downtime

A downtime window is required to perform the conversion of the database.

In the downtime window, applications accessing the production database are shut down, and the database is accessible only to migration tools. You can scan for issues and cleanse them parallel to standard production work. Preferably, the downtime window occurs during off-peak hours.

Keep the downtime windows as short as possible during migration. Consider the possibility of something going wrong, and how long resolving the problem might take. The tolerated downtime can be anything from a matter of hours for mission-critical applications to several days for a common data store.

The migration methods usually offer some flexibility regarding the accepted time between the last scan for data issues and the conversion of the database. If the last scan is performed in a production database before the downtime window starts, then the downtime window is smaller. However, there is some risk that new issues might be introduced in the time between this last scan and the downtime window. By analyzing the downtime requirements and the database access patterns, you can find the balance between the risk of introducing data integrity issues and the inaccessibility of the database for production use.

The DMU offers the flexibility to scan only selected tables. This feature enables you to scan only those tables in the downtime window for which the risk of introducing new issues is significant.

1.5.5 Character Set Migration: Failure Recovery

When you perform a migration, the process may abruptly terminate, leaving the database in an inconsistent state. Dealing with failures during the migration process is a necessary part of migration planning.

Many possible events may trigger termination, including a software defect or hardware malfunction. If the DMU cannot resume the migration, then you may need to bring your database back to a consistent state. Typical responses are recovering from a backup or using Oracle Flashback Database.

> ✎ **See Also:**
>
> *Oracle Database Backup and Recovery User's Guide* for more information about how to recover a database

1.5.6 Character Set Migration: Application Impact

The character set migration of a database, especially from a single-byte to a multibyte character set, may have a significant effect on applications.

The following issues could affect the application:

- If table columns are lengthened or migrated to other data types to solve length expansion issues, then you may need to enlarge buffer variables in the application code and adapt application logic to process `CLOB` data.

- An application could make an assumption about maximum byte width of a character code in the database character set.

- An application could rely on a specific binary ordering of character values. In Unicode, the ordering of non-ASCII values frequently differs from the ordering in a legacy character set.

- To take full advantage of a new multibyte database character set, you may need to adapt single-byte applications to process multibyte text correctly. This is not a trivial change.

You must handle the preceding application issues manually. Application developers must review and test their applications with a migrated test copy of the production database.

2

Getting Started with the DMU

This chapter introduces the Database Migration Assistant for Unicode (DMU) and describes the basic workflow of migrating a character set using this utility.

2.1 Using the Database Migration Assistant for Unicode: A Roadmap

This section provides an overview of the tasks involved in working with the DMU.

To work with the Database Migration Assistant for Unicode:

1. Review the requirements.

   Ensure that your environment meets the requirements to work successfully with the DMU.

   See "Overview of Database Requirements", "Overview of Java Runtime Requirements", and "Overview of DMU Security Considerations".

2. Perform basic preparatory tasks, such as installing the DMU, reviewing the main user interface, creating a database connection, reviewing the user interface, and installing the DMU repository.

   See "Installing the DMU", "Creating a Database Connection", "Installing the DMU Repository", and "Introduction to the DMU User Interface".

3. Scan the database to discover any possible problems that could prevent a successful migration to Unicode.

   See "Scanning the Database".

4. Cleanse the database of the possible problems that you discovered during the scanning phase.

   See "Cleansing the Data".

5. Convert the database to Unicode.

   See "Converting the Database".

6. Validate the database to verify that the character set is now Unicode and contains no exceptional data.

   See "Validating Data as Unicode".

2.2 Introduction to the DMU Interface and Navigation

From the DMU home page, you can perform the basic tasks associated with migrating a database to Unicode.

DMU Home Page

This is the page that loads when you click the DMU icon. See "Introduction to the DMU User Interface".

Online Help

You can access context-sensitive online help by clicking the Help link displayed at the top of every page. On any help page, click **Help** or press the **F1** key.

Navigation

Navigational features of the DMU include the following:

- Subpage links at the top of the page. These links take you to the various subpages that organize management tasks into distinct categories.

- Drill-down links that provide increasing levels of detail.

2.3 Overview of Requirements and Security Considerations

When using the Database Migration Assistant for Unicode, you must consider several requirements and restrictions.

This section also describes the first steps you must take to start the utility.

2.3.1 Overview of Database Requirements

To be supported by the DMU, a database must meet several requirements, including release version, type of character set, open mode, and so on.

The requirements are as follows:

- The release of Oracle Database must be 10.2.0.4, 10.2.0.5, 11.1.0.7, 11.2.0.1, or later. Check the latest version of the DMU release notes for any additional patches that must be installed in the database. The release notes also list further database releases that are supported with appropriate patches.

- The database character set must be ASCII-based. For example, databases running on the EBCDIC-based platforms IBM z/OS and Fujitsu BS2000 are not supported.

- The `SYS.DBMS_DUMA_INTERNAL` package must be installed in the database.

  This package is available as part of the database installation. You must create it manually by running the script `?/rdbms/admin/prvtdumi.plb` from the Oracle home. You must log in as `SYSDBA` to run the script.

- You must disable Oracle Database Vault before starting the migration. The DMU has not been certified to work with Database Vault enabled.

- The database must be opened in read/write mode.

Additional requirements pertain to databases that the DMU should convert. Without meeting these requirements, you can still use the DMU for scanning and cleansing the database. These requirements are as follows:

- All database objects, including objects created by standard PL/SQL packages, such as `DBMS_RULE`, `DBMS_DATA_MINING`, or `DBMS_WM`, must be named using only standard characters from the ASCII character set. Similarly, expressions of `CHECK` constraints, virtual columns and other database features must be specified only using those characters. In other words, the data dictionary of the database cannot contain non-ASCII characters. However, there is some character data in the data dictionary in which non-ASCII characters are allowed.

  See "Data Dictionary Tables That Are Converted" for more details.

- No OLAP analytical workspaces, other than predefined system workspaces and certain predefined Oracle Applications workspaces, can exist in the database.

- No flashback data archives can exist in the database.

- No data to be converted can reside in a read-only or offline tablespace.

- Neither cluster key columns nor partitioning key columns can be defined with character length semantics.

- No convertible data can be present in tables in the recycle bin.

- No convertible data can be present in a reference partitioning key column.

This release of the DMU supports the migration of pluggable databases (PDBs). To convert a PDB using the DMU in a container database (CDB), the root container's character set must be the same as the migration target character set (AL32UTF8 or UTF8). When this requirement is not met, you can still use the DMU for scanning and cleansing PDBs, but the conversion operation is disallowed.

This release of the DMU also supports the migration of Oracle E-Business Suite and PeopleSoft databases to Unicode. The migration workflow is mostly the same as that of a regular database instance. Note that you should not perform any data cleansing actions that require modifying the database schema of Oracle E-Business Suite or PeopleSoft applications because such modifications will not be supported by Oracle. In addition, this release of the DMU does not support migrating databases for PeopleSoft Application versions older than 9.0 or PeopleTools versions older than 8.48.

2.3.2 Overview of Java Runtime Requirements

The Java SE Development Kit (JDK) version 8 or version 11 must be available on the system running the DMU.

When it is first started, the DMU asks for the location of the Java Runtime executable file. Depending on the platform, this file is either `java.exe` or `java`.

From the Java SE download page on `http://www.oracle.com`, you can download the JDK for Linux, Solaris, and Microsoft Windows operating systems for Intel x86, Intel x64, Intel Itanium, and SPARC processors. Look for the installation instructions for your platform on the relevant Oracle Technology Network documentation page at `http://www.oracle.com/technetwork/java/javase/downloads/index.html`.

For other platforms, such as Hewlett-Packard HP-UX or IBM AIX, visit the vendor's website.

2.3.3 Overview of DMU Security Considerations

Before you start the migration, be aware of the DMU security considerations listed in this section.

- Protect DMU software integrity

  In this release, the DMU requires that you connect to a database with the `SYSDBA` privilege. This highest privilege is needed to analyze and convert all data in the database. But this also means that once a connection is established, the program—if maliciously modified—could read and change all database content, create database users and grant privileges to them, and so on.

  To ensure that this power is not abused, the DMU software files must be modifiable only by users who are also authorized to connect as `SYSDBA`. This implies that all files in the DMU installation directory must be owned and writable only by an operating system user to

which only the trusted administrators can connect. If writable by a malicious user, the DMU program code in JAR files, the repository maintenance SQL scripts, or configuration files could be abused to inject malicious SQL statements.

Unless you install the DMU on a host to which only authorized users have access, you must protect the DMU installation and configuration files using the operating system privileges. After you have uncompressed the archive file with the DMU installation, ensure that all uncompressed files and directories are writable only by you and other authorized operating system users. The DMU does not have an installer that can automatically set file permissions.

- Protect connection details

  If you select the **Save Password** check box when creating a database connection, the password you specify is saved in an obfuscated form in a password file named `cwallet.sso` in your user directory. Because obfuscation is a reversible operation, use this feature only for passwords to test databases with no production data or only if the DMU is installed on a very well-protected host. Ensure that the password file is readable only by you.

  The `cwallet.sso` file is in the `$HOME/.dmu/` directory on Linux and UNIX systems, and in the `%APPDATA%\DMU\` directory on Microsoft Windows systems.

- Protect DMU repository objects

  To avoid unauthorized access to potentially sensitive metadata and functionality, do not grant any privileges for the DMU repository database objects to any database users. This guideline applies to any object with a name starting with `DUM$` or `DBMS_DUMA`.

- Avoid information leakage

  To avoid unauthorized access to potentially sensitive information, ensure that the DMU configuration, log, and report files are readable and modifiable only by authorized operating system users. By default, these files are in the `$HOME/.dmu/` directory on Linux and UNIX systems, and in the `%APPDATA%\DMU\` directory on Microsoft Windows systems. However, DMU users can often select a file path for generated logs and reports.

- Protect operating system files

  The DMU lets its users specify file paths to access specific files. In a remote terminal setup where a DMU user does not have full shell access to the shared host, this functionality could be abused to read or overwrite sensitive files. The host's operating system administrator should protect all sensitive operating system files from inadvertent or malicious overwriting. This precaution is not specific to the DMU.

2.3.4 Review Your Preparations for Migration

Before you start the migration, collect information such as whether your database is supported, the target character set, which languages can be stored in the character set, and so on.

To help you set up the DMU properly, answer the following questions about your database and its use by applications:

- Is your database supported by the DMU?

  Verify that the version of your database software is supported by the DMU as described in "Overview of Database Requirements". The platform on which the database is installed must be ASCII-based (that is, not IBM z/OS or Fujitsu BS2000). Your database must not have Oracle Database Vault installed.

- What is the target character set?

You have to decide if the target character set of the migration is AL32UTF8 or UTF8. Oracle recommends AL32UTF8, which is a proper implementation of the UTF-8 encoding form of the Unicode Standard. The DMU supports migration to the older character set UTF8 for databases that must support database client software based on Oracle 8*i* Client libraries, or that must support applications certified with UTF8 but not with AL32UTF8, for example, Oracle Applications Release 11*i*. The UTF8 character set, despite its name, is an implementation of the CESU-8 compatibility encoding form of the Unicode Standard. CESU-8 is very similar to UTF-8 except for the way the supplementary characters are stored (that is, characters with Unicode code points U+010000 and higher). Oracle does not generally update the definition of UTF8 to synchronize it with the newest versions of the Unicode Standard, while Oracle does update the definition of AL32UTF8.

- Which languages and character sets can be stored in the database?

  The very nature of character set encodings and the very nature of character storage in an Oracle database make it impossible to automatically and precisely recognize what language in what character set is represented by a given sequence of bytes in a column value. The DMU scanning process can tell if there are bytes in a character value that are invalid in the declared character set of this value, but not more. If almost all bytes of the declared character set are assigned to single-byte codes of some characters, such as the case of CL8MSWIN1251, which defines all bytes except 0x98, then virtually any sequence of bytes declared as encoded in this character set will appear as valid to the scanning process. Therefore, even if an application uses the pass-through configuration to store data in the database that it interprets in a character set different from the database character set (see "Invalid Binary Storage Representation of Data") the DMU might not see any invalid codes and it might not signal the mismatch between the real (application) character set of the data and the declared database character set. Hence, it is very important that you research, for example, by asking application developers and administrators, what languages or character sets are most probably stored in your database. This will help you to manually analyze the contents of the database, if necessary, to supplement the automatic analysis by the DMU.

- Which columns contain data in foreign languages?

  If your database contains data mainly in one language and only a few table columns might contain other languages, for example foreign customer names and street addresses, a list of all such columns facilitates the manual search for invalid data stored in the pass-through configuration, if the existence of such data becomes probable after analysis of the information gathered in the previous list item.

- Which character columns might contain binary data?

  Some applications might connect to your database in the pass-through configuration and use character columns to store data that is binary in nature, such as text encrypted without using the Transparent Data Encryption feature of the database, images, text in a binary format of a word processor, and so on. Obtain a list of those applications from application developers or administrators to further help you analyze the contents of your database.

  See "Cleansing Scenario 3: Cleansing Invalid Representation Issues" for information about resolving invalid binary representation issues using the collected information.

- What is the real character set of your database?

  Determine if the character data in your database is actually to be interpreted in a particular common character set of the clients connecting to database and not in the declared database character set. To improve the effectiveness and accuracy of the analysis that you perform with the help of the DMU to verify that the database can be converted to Unicode.

  One way to determine if the real character set of your database differs from the declared database character set is to look for the following characteristics:

- The database (declared) character set is US7ASCII, or WE8ISO8859P1, or WE8ISO8859P15.

- The client character set, declared in the `NLS_LANG` client setting, is the same as the database character set. If the `NLS_LANG` is not specified at all, the client character set defaults to US7ASCII. `NLS_LANG` affects only C or C++ clients connecting through an OCI API. Java clients connecting through a JDBC API are always UTF8 clients.

- All database client software runs on the Microsoft Windows platform and works in one of the Windows character sets, which are also known as ANSI Code Pages. This character set depends on the language version of Windows: WE8MSWIN1252 – US and Western European versions, EE8MSWIN1250 – Central European versions (for example, Polish), CL8MSWIN1251 – Cyrillic versions (for example, Russian), AR8MSWIN1256 – Arabic version, and so on.

  This Microsoft Windows character set common to all clients is the real database character set that was to be identified.

  If your database is used in a correct character set configuration, that is, the `NLS_LANG` setting always correctly corresponds to the real character set of the database clients, the real character set of the database is its declared character set.

- What is the connection information for your database?

  To migrate a database with the DMU, you must have the `SYSDBA` privilege and the required connection credentials (user name and password). Starting in Oracle Database 12c, to perform scanning and cleansing operations on a PDB, you can connect as a user who has the `SYSDBA` privilege in the local PDB. However, to convert a PDB to Unicode, you must connect as the `SYS` user.

  You need to know the host name or IP address of the database server, the port number on which the database listener listens for connection requests, and the service name of the database. For performance reasons, Oracle recommends that you connect to a service that is configured for dedicated connections, that it, not using the a shared server configuration.

After you have collected the preceding information, you can install the DMU.

> ✎ **See Also:**
>
> *Oracle Database Security Guide* for further security considerations

2.4 Performing First Tasks with the DMU

Before you can begin using the DMU, you must perform installation and creation tasks.

2.4.1 Installing the DMU

Oracle recommends that you run the DMU on the database host or on a workstation connected to the database host with a fast and reliable local area network.

Before installing the DMU, always download the current release notes available for the DMU version you are using.

To install the DMU:

The DMU does not come with an installer. Install the utility by uncompressing the installation file into a directory of your choice.

1. To install the DMU, uncompress the downloaded archive file to any directory on the host on which you want to run the DMU.

2. After you have uncompressed the archive file, ensure that the DMU files are writable only to you and other authorized operating system users. This is very important because unprivileged users with access to the DMU host could modify the DMU files to make the DMU execute arbitrary SQL statements when the DMU is later started with SYSDBA credentials. Such SQL statements could compromise database security.

3. After you have installed the DMU, initialize the database for the migration process with the DMU.

> ✎ **See Also:**
>
> • *Oracle Database Migration Assistant for Unicode: Getting Started* page on the Oracle Technology Network (OTN) website for the prerequisites and the detailed steps for installing the DMU software:
>
>   http://www.oracle.com/technetwork/database/database-technologies/globalization/dmu/learnmore/start-334681.html
>
> • *Oracle Database Migration Assistant for Unicode Release Notes*

2.4.2 Creating a Database Connection

You must create a connection to the database that you want to analyze or migrate.

To create a database connection:

1. In the Navigator pane, right-click the **Databases** node to open the context menu. Alternatively, open the **File** menu, and then select **New Database Connection**.

   The Create Database Connection dialog box appears.
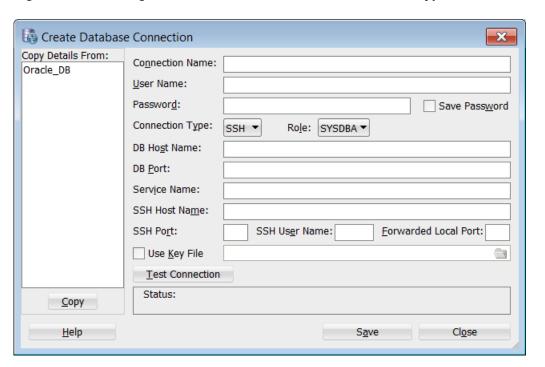
Figure 2-1   Creating a Database Connection - Basic Connection Type

In this dialog, the connection type of `Basic` is selected by default.

If you select the connection type of `SSH` in this dialog, then the dialog shows additional fields to connect to the database securely using the Secure Shell (SSH) protocol:

Figure 2-2   Creating a Database Connection - SSH Connection Type

2. Enter the following information to define a connection:

- Connection Name

  Give a descriptive name to the connection so that you can easily identify to which database the DMU is connected.

- User Name

  Specify a database user name to use for authentication with the database. In this release, DMU requires the user to have the `SYSDBA` privilege.

- Password

  Specify the password for the database user.

- Save Password

  Select this check box if you want to save the password on the local system, so that you are not prompted for password each time you open the database connection.

  > **Note:**
  >
  > If you select the **Save Password** check box, then the password will be saved along with other connection details in an obfuscated form in a configuration file in your user directory. Because obfuscation is a reversible operation, use this feature only for passwords to test databases with no production data or only on very well protected hosts. Ensure that the configuration files storing your preferences and connection information are readable only by you. On Linux and UNIX systems, the configuration files are in the `$HOME/.dmu/` directory. On Microsoft Windows systems, the configuration files are in the `%APPDATA%\DMU\` directory. Otherwise, you risk compromising the security of your database. If you deselect the **Save Password** check box, you will be prompted for the password each time you open the connection.

- Connection Type

  Specify one of these two connection types to connect to the database – `Basic` or `SSH`.

- Role

  In this DMU release, only the `SYSDBA` role is supported.

- DB Host Name

  Specify the DNS name or the IP address of the database host.

- DB Port

  Specify the TCP/IP port on which the Net Services (TNS) listener listens for database connection requests.

- Service Name

  Specify the service name of the database instance to which you want to connect.

- If you select the connection type of SSH, then the following additional fields are displayed:

  - SSH Host Name

    Specify the DNS name or the IP address of the SSH server.

  - SSH Port

Specify the port for the SSH server.

– SSH User Name

Specify the SSH server user name.

– Forwarded Local Port

Specify local port of the system where the DMU application is running.

– Use Key File

Select this option to connect to the SSH server using key file authorization. If you select this option, then specify the key file with the complete directory path.

> **✎ Note:**
>
> * The supported key types are: DSA, RSA, and ECDSA.
>
> * The supported key formats are: OpenSSH and Putty.
>
> * For the key file validation rules, refer to the documentation for the Apache Mina SSHD API `org.apache.sshd.common.config.keys.KeyUtils.validateStrictK eyFilePermissions()`.

You can test the connection by clicking **Test Connection**. For the Basic connection type, if the database administrator password is not already stored in the DMU, then you will be prompted to enter the database administrator password. For the SSH connection type, if a key file is not used, then you will be prompted to enter password for the SSH server user. For the SSH connection type, if a key file is used and a passphrase is set for the key, then you will be prompted to enter the passphrase.

If the connection can be established, the Status box will show "Success". Otherwise, it will show an error message describing the connection problem. You can create the connection and close the dialog box by clicking **Save**.

After a connection has been created, a new database node with the name of the connection is added to the Navigator pane. You can open the node, that is, connect to the database, by right-clicking it and choosing **Connect** from the context menu. Only one connection can be open in the DMU at the same time, thus the DMU will ask you for permission to close any currently open connection before it will open a new one.

You can change the details of an existing connection, such as user name or host name, by right-clicking the corresponding database node and choosing **Connection Details** from the context menu. The Modify Database Connection dialog box that is shown has the same elements as the Create Database Connection dialog box.

You can rename a connection by selecting **Rename Connection** or delete a connection by selecting **Delete** from the context menu of the corresponding database node. The Connection Details, Rename Connection, and Delete menu items are available in the context menu of a database node only if the node is not currently connected.

2.4.3 Installing the DMU Repository

The **DMU repository** manages the information necessary for each step of the migration.
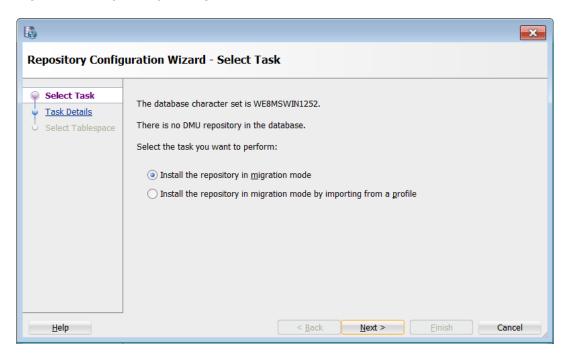
The DMU repository contents include items such as objects to be processed, details on data that had an error flagged, and the progress of a scan or a conversion. For any database

without a DMU repository, an automatic repository creation wizard begins each time you connect to that database. You can also start the wizard from the DMU user interface.

To install the DMU repository:

1. If you connect to a database for the first time, the DMU automatically prompts you to install the repository. If you are not prompted to install the repository, you can install it by right-clicking the database you want to use, and selecting **Configure DMU Repository**. You can also select **Configure DMU Repository** from the Migration menu. In all of these cases, the Repository Configuration Wizard appears. Figure 2-3 shows how it appears.
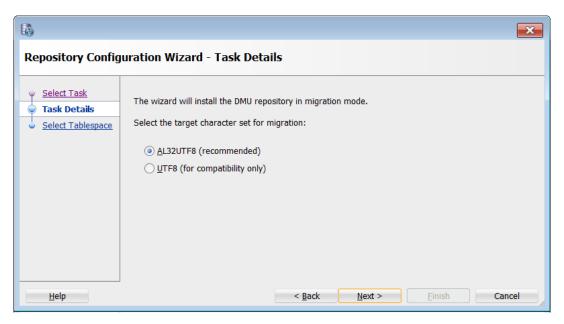
Figure 2-3   Repository Configuration Wizard - Select Task

2. On the first page of the wizard, the only choice available is Install the repository in migration mode. After selecting this, click **Next**.
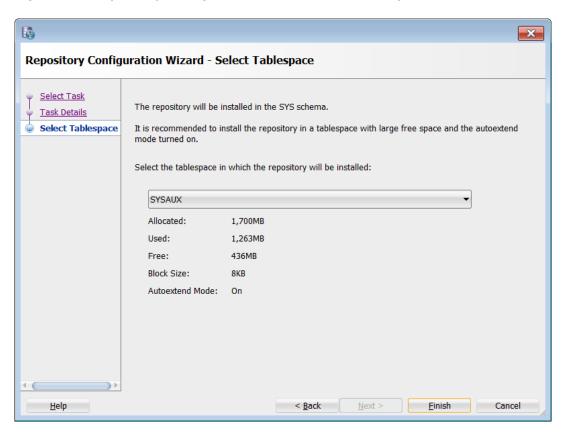
   After you click **Next**, the second page of the Repository Configuration Wizard is shown. Figure 2-4 is an example of this page.

Figure 2-4   Repository Configuration Wizard - Task Details

3. Here, you select the target character set for the migration. You can choose between AL32UTF8, the normally recommended character set, and UTF8, which might be required for compatibility with certain applications, such as Oracle Applications 11*i*. After selecting the character set, click **Next**. Figure 2-5 appears.

Figure 2-5   Repository Configuration Wizard - Select Tablespace

4. On the third page, you can select the tablespace in which you want to install the DMU repository. The default tablespace is `SYSAUX`.

Oracle recommends that you create a separate, empty tablespace to be used for the DMU repository. This helps to avoid fragmentation of the production tablespaces. Refer to "Creating a Tablespace for the DMU Repository" for more information.

Click **Finish** to install the repository.

After you have the repository installed, you can begin working on the data.

2.4.4 Following the Status of a Migration

After a database connection has been established, you can follow the progress of the migration by looking at the Migration Status tab.

This tab, which is displayed in the Client pane of the DMU main window, shows the important milestones in the DMU process:

- Installing the repository
- Scanning the database for problems
- Resolving any problems encountered
- Converting the data

The Migration Status tab shows the completion status of each milestone, suggests a next action, and lists issues that prevented a successful conversion. All this information indicates the progress in the workflow. Return to the Migration Status tab periodically to enable the Database Migration Assistant for Unicode to guide you through the migration process.
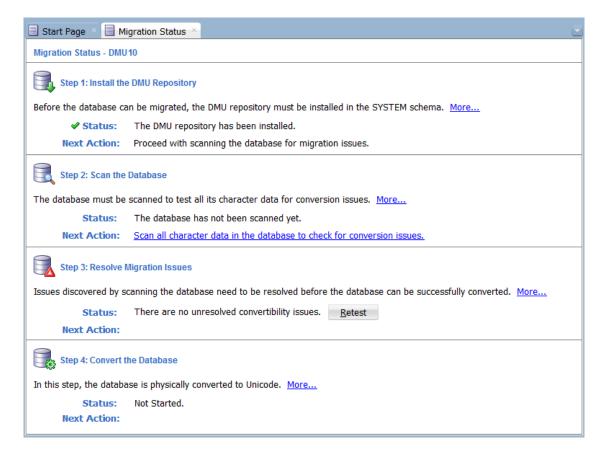
An example of Migration Status tab content is shown in Figure 2-6. The Migration Status tab describes each of the four main steps of the migration:

- Installation of the DMU repository
- Scanning of the database
- Resolution of migration issues
- Conversion of the database

The Migration Status tab shows the status of each of these steps and suggests the next action to take. You can click the **More** links to open help pages that expand the presented information.

Figure 2-6   Migration Status Tab

- **Step 1: Install the DMU Repository**

    This section informs you if the connected database contains a DMU repository. It can also report that the installed repository is in an incompatible version. Until the repository is installed in the correct version, you cannot proceed to the following steps of the migration process. Follow the advice given in the Next Action area. If the text in this area is a link, click the link to start the recommended action. See "Installing the DMU Repository".

- **Step 2: Scan the Database**

    This section shows the scan status of the database. The database might be in any of the following conditions:

    – Not yet scanned – no scan has been executed yet after the DMU repository has been created

    – Being scanned – a scan is running in the database just now

    – Partially scanned – only part of the database has been scanned

    – Entirely scanned – all tables in the database have valid scan results

    – Contained invalidated results – scanning results for some tables have been invalidated because the table structure has been modified since the last scan

    If the database has not been entirely scanned, see "Scanning the Database" and scan all those tables that have missing or invalid scan results.

- **Step 3: Resolve Migration Issues**

This section shows the list of issues that must be resolved before you can start the actual conversion of the database content. Click the plus icon to the left of the text "Unresolved convertibility issues found" to see the issues. The issues are classified as warnings and blocking issues.

Warnings, marked with a yellow color, describe features and configuration details of the database that might cause problems during conversion in some situations and not cause any problems in other situations. The DMU is not able to automatically analyze the probability of these problems to arise. Some warnings describe configuration details of the database that might decrease the performance of the database conversion, but which the DMU cannot change itself because of the possible side effects. You might see the following warnings:

– User-defined OLAP analytical workspaces are present in the database.

– The database is a standby database.

– There are external tables with convertible data in the database.

– There are convertible primary key-based object identifiers (OIDs) in the database.

– The current setting rules out the CTAS conversion method for tables with row movement disabled.

– Turning off the `FORCE LOGGING` mode of the database or tablespaces might improve conversion performance, though at the expense of the ability to perform media recovery.

– The conversion of some partitioned tables might fail due to error ORA-14402, because the tables do not have the row movement option enabled.

– Some tables have been excluded from conversion.

– Convertibility issues in some of the columns will be ignored and the columns will be converted despite the resulting data.

You can choose to ignore some of the warnings and start the conversion, but ensure that you understand all the implications.

Blocking issues, marked with red color, are issues that prevent the DMU to enter the conversion phase. They are known to cause problems in the database, if ignored. You must resolve all blocking issues before you can proceed with converting the database. After all issues are resolved, the status area in this section shows the message "No unresolved convertibility issue found" and proceeding with the conversion process is considered safe.

• **Step 4: Convert the Database**

This section provides the information about the status of the database conversion process.

To view the status of a migration:

1. Click the **Migration Status** tab in the Client pane.

2. If you have modified the database outside of the DMU to resolve some of the convertibility issues, click **Retest** to let the DMU recheck the database and update the status information.

3. To reopen the tab at any time, select **Migration Status Panel** from the Migration menu.

2.5 Introduction to the DMU User Interface

The default layout of the main window of the DMU user interface contains a menu bar and a toolbar at the top, and a status bar at the bottom. The remaining area of the window is divided into panes.

The default layout is shown in Figure 2-7.

The *Navigator pane* in the top left-hand part of the window shows a tree of database objects on which the DMU operates. The first level of nodes in the tree represents databases to which the DMU can connect. Only one of the database nodes can be associated with an open connection. For further information about connections, see "Creating a Database Connection".

A database node with a currently open connection can be expanded to a subtree of schemas, tables, and columns contained in the database that are relevant to the migration process. Relevant columns are columns that contain character data and might, therefore, require character set conversion when migrated to the Unicode character set. Due to differences in the way the DMU processes them, predefined Oracle schemas comprising the data dictionary are grouped together and displayed separately under the Oracle-Maintained Schemas group node. Remaining schemas are displayed under the Application Schemas group node. Similarly, tables in a schema are grouped into materialized views, under the Materialized Views group node, and other tables, under the Tables group node.

A status icon might be displayed to the left of a node name in the Navigator tree. Table 2-1 shows the possible icons and their meanings for particular node types.
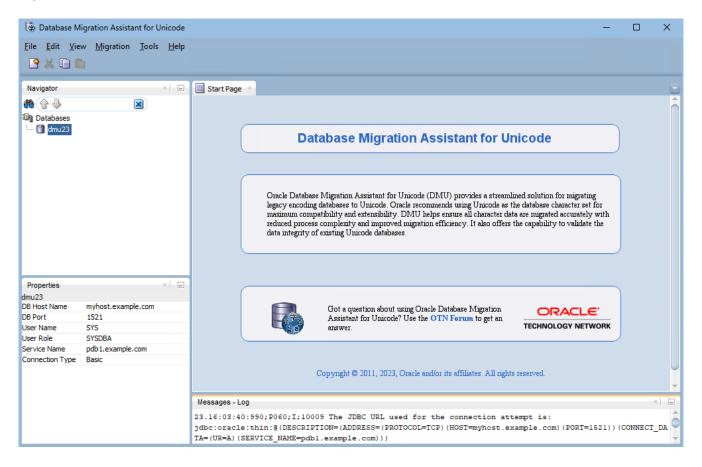
The *Client pane* in the upper right-hand part of the window is an area in which various tabs are opened in the process of migration. These tabs display object properties, scanning results, cleansing tools, and progress status of tasks. All the tabs are described in the remaining sections of this chapter. When you start the DMU, before a database connection is made, the client pane contains only the Start page, which is a collection of links to various sources of information about the DMU.

The *Properties pane* in the bottom left-hand part of the window displays selected properties of the most recently clicked node in the Navigator pane or in the Database Scan Report (see "Scanning the Database"). The properties displayed depend on the type of node.

The *Log pane* in the lower right-hand part of the window displays error and warning messages reported by the DMU during the migration process.

Figure 2-7   DMU User Interface

You can customize the layout of the main DMU window by dragging panes and tabs to new positions, but the customized layout is not preserved across program runs.

Table 2-1   Database Migration Assistant for Unicode Icons

Icon	Column Node	Table, Schema, Database, and Grouping Nodes
No icon	The column has never been scanned.	No column in the table/schema/ database/group has ever been scanned.
Check mark	The column has been successfully scanned and no issues have been found.	All columns in the table/schema/ database/group have been successfully scanned and no issues have been found.
Red circle with a white X	The last attempt to scan the column failed.	A recent attempt to scan one or more columns of the table/schema/database/ group failed.
Warning triangle in yellow	The column has been successfully scanned, but some convertibility issues have been found.	All columns in the table/schema/ database/ group have either been successfully scanned, or they have never been scanned or their scan results have been invalidated; some convertibility issues have been found in the scanned columns.

Table 2-1   (Cont.) Database Migration Assistant for Unicode Icons

Icon	Column Node	Table, Schema, Database, and Grouping Nodes
Round red circle with diagonal line	The column has been previously scanned, but the scan results have been invalidated due to the containing table being altered or a cleansing action being applied.	One or more columns in the table/ schema/database/group have had their scan results invalidated while all other columns in this table/schema/database/ group have never been scanned.
Magnifying glass	Not applicable.	Some columns in the table/schema/ database/ group have been successfully scanned, and no issues have been found, but other columns have never been scanned or their scan results have been invalidated.
Moving magnifying glass	The column is being scanned; this status overrides all other statuses.	One or more columns in the table/ schema/ database/group are being scanned; this status overrides all other statuses.

2.6 Overview of Data Preparation

Data preparation ensures that no data to be migrated causes problems during or after conversion. The elements of data preparation are scanning and cleansing.

2.6.1 Data Preparation: Scanning

In this step, the DMU scans tables for problems, assessing the feasibility of migrating the data to Unicode.

The most common types of problems are with your data. Issues include values expanding during conversion beyond column or data type limits, data in a mislabeled character set, or binary data stored in character data types.

During scanning, the DMU reads specified character columns and performs a test conversion of each column value to the target character set, AL32UTF8 or UTF8. Depending on the result of this conversion, the DMU classifies data as follows:

- Needs no conversion

  The data is fine, because the binary representation of the data does not change in the conversion.

- Needs conversion

  The data must be converted, because the binary representation of the data does change, but no other data issues have been found.

- Invalid binary representation

  The binary representation of the data is invalid under the current database character set. If you do the conversion in this state, the resulting data will usually not make sense to applications and users.

- Exceeds column limit

The data will not fit into a column after migration.

- Exceeds data type limit

    The data will exceed a data type limit after migration.

Each column value is assigned to only one of the preceding categories. Values that have invalid binary representation are classified only as such even if their lengths exceed column or data type limit after conversion. As conversion of invalid character codes usually yields the default replacement character, which has a three-byte representation in AL32UTF8 and UTF8, the length expansion issues are not rare among values with invalid representation. Because the DMU let you ignore invalid representation issues and force conversion of a column, you should be aware that forcefully converted values with invalid binary representation may be additionally truncated. You can compare the value of the Maximum Post-Conversion Length property of the column with the column and data type length limits to see if the truncation will take place.

This release of the DMU supports Oracle Database 12*c* extended data type limit where the maximum size limit for VARCHAR2 has been increased to 32767 bytes. The extended type limit guarantees that VARCHAR2 data created in earlier releases up to the original type limit of 4000 bytes can be migrated to Unicode on Database 12.1 with no over data type limit expansion exceptions.

After you have run a scan, the DMU creates a Database Scan Report, which can be found under the Migration drop-down menu. This scan report shows the statistics for the current data under each of the preceding categories. The counts of values are presented for each character data type column, and also summed up at the table, schema, and database levels. The scan report enables you to filter and customize the output, for example, so that only columns with selected potential problems such as exceeding the column limit are displayed.

In addition, you can use the report to iteratively go through the steps of taking the data that is not clean, reviewing this data in the Cleansing Editor, fixing the data, either immediately or at your convenience later, and rescanning to verify that the data is now clean. While scanning the data, you can generate an index check report, which can be found on the Indexes subtab of the Database Scan Report. The index check report lists the indexes defined on character columns whose key lengths will exceed the maximum allowed key size due to column modifications made with the Cleansing Editor or the increased post-conversion data size of the columns. This enables you to fix any problematic indexes before proceeding to the data conversion phase.

You can perform scans on the following objects:

- All tables in the database

- All tables in the data dictionary of a database

- All application schema tables in a database

- All tables in an application schema

- A table column in an application schema

- An arbitrary set of application schema tables and columns in a database

As mentioned in "Overview of Character Set Migration Considerations", certain data issues cannot be discovered automatically. For example, the DMU does not analyze character data in binary data types. Also, some single-byte character sets define almost all byte values as valid codes. For example, the only byte value that is not a valid CL8MSWIN1251 character code is 0x98. Therefore, if the incorrectly stored data happens not to contain any of the undefined byte values, then the test conversion during a scan might not show invalid binary representation problems. Even if the database scan reports no issues, collect and analyze information about use of the database by applications and try to identify these types of hidden problems before

attempting the database conversion. Creating a test copy of the database, migrating it to Unicode, and thoroughly testing it with your applications is also a way to discover many problems.

> ✎ **See Also:**
>
> - "Scanning the Database"
> - "Column Properties: Scanning"
> - "Ignoring Convertibility Issues"

2.6.2 Data Preparation: Cleansing

In this step, the DMU cleans the data in tables based on identified issues.

You can define cleansing actions for immediate or for delayed execution (immediate versus scheduled cleansing mode). You can perform certain types of cleansing in a production environment with no side effects to applications. Other cleansing operations involve metadata changes and require that you adapt applications to these changes. The scheduled mode enables you to define a cleansing action at any time but delay its execution to the conversion phase of the migration process. This is usually the most convenient moment to introduce metadata changes because the database is not used in production and new application versions can be easily deployed at the same time.

Immediate changes to metadata are defined in the Modify Column and Modify Attribute dialog boxes. Scheduled changes are defined in the Schedule Column Modification and Schedule Attribute Modification dialog boxes. All these dialog boxes can be invoked from the context menu of a column or attribute in the Cleansing Editor. Immediate editing changes of user data are performed directly in the Cleansing Editor.

In the immediate mode, any change entered is performed immediately after an appropriate **Save** button is selected. In this case, all SQL statements are issued to the database and the transaction is committed.

In the scheduled mode, clicking **Save** puts the cleansing action into the DMU repository. The corresponding statements are executed during the conversion step. You can change the scheduled action or remove it, provided the conversion step has not yet been started, by reopening the Schedule Column Modification or Schedule Attribute Modification dialog box and selecting **No Modification**.

You apply the editing changes performed in the Cleansing Editor on table data to the database by clicking **Save** on the Cleansing Editor tab. You can revert the changes that have not yet been saved by clicking the **Revert** button.

The Cleansing Editor can also be used to set the assumed character set of a column. This property tells the DMU to interpret the contents of the column in a character set different from the database character set. The selected character set is applied to the test conversion during scanning, actual conversion occurs in the conversion step, and the set is used to interpret data for display in the Cleansing Editor. If column data in the Cleansing Editor is not legible, the character set of the column might be selected incorrectly.

After cleansing, rescan the database to verify that your changes have successfully handled the potential problems. This is an iterative process where you scan and cleanse until there are no more conversion issues before you proceed to the conversion phase.

> ✎ **See Also:**
>
>   Using the DMU to Cleanse Data for details

2.7 Overview of Data Conversion

The data conversion phase is where the actual modification of the database contents occurs. After the conversion is complete, the database character set will be Unicode.

2.7.1 Data Conversion: Preparing

When you select **Convert Database** in the Migration menu, the DMU first executes a conversion feasibility test.

This test checks that:

*   All database convertibility requirements listed in "Overview of Database Requirements" are met

*   All data in the database has been scanned and has valid scan results

    Cleansing actions might invalidate scan results, so cleansed tables might need to be rescanned.

*   No data in the database has binary representation or length issues

If the test succeeds, the DMU presents a plan for conversion. This plan shows all SQL statements that will be executed to convert the database, including statements to handle auxiliary objects, such as indexes, constraints, and triggers. You can customize the plan as desired. There are various database-level and table-level options that can be used to influence the way DMU converts the database. For example, you can set the conversion method for a given table or the number of processes participating in the conversion step. The conversion plan is generated and displayed in the Conversion Progress tab. You can then modify the available options by clicking **Edit Table Conversion Plan** or **Edit Database Conversion Parameters**.

2.7.2 Data Conversion: Converting

This step converts the data into Unicode.

You accept the conversion plan and initiate the conversion by clicking the **Convert** button on the Conversion Progress tab. The DMU repeats the conversion feasibility test to ensure that no issues have been introduced while you worked on the conversion plan. It verifies also that no other sessions are connected to the database and that the database is mounted in exclusive mode. Then, the DMU begins to migrate the data, executing the statements from the conversion plan.

You can monitor the conversion progress on the Conversion Progress tab.

As an overview, the process the DMU uses to convert data is:

1. Put the database into restricted mode.

2. Disable various job queue processes.

3. Drop or disable selected indexes.

4. Disable selected triggers and constraints.

5. For tables assigned the "Convert using Data Pump" conversion method, export the tables to a database directory using Oracle Data Pump.

6. Convert the data in user tables (except blockchain and immutable tables) and in selected data dictionary tables to Unicode.

7. Convert `CLOB` columns in the data dictionary.

8. Issue the `ALTER DATABASE CHARACTER SET` statement, and convert the data in blockchain and immutable tables to Unicode.

9. For tables assigned the "Convert using Data Pump" conversion method, import the data into the tables using Oracle Data Pump.

10. Enable triggers and constraints; and re-create indexes and constraints.

11. Restore the database instance parameters.

The conversion of a table is performed by updating its columns with an `UPDATE` statement, by converting the columns while re-creating the table using the `CREATE TABLE AS SELECT` statement, or by exporting and importing the table with Oracle Data Pump. The re-creation of a table is faster than an update if most of the table rows must be converted. Oracle Data Pump export and import may be faster than the other conversion methods if the table is large and contains `CLOB` data.

After the conversion has finished, the DMU will re-create or re-enable any objects that were dropped or disabled earlier. You can check that no errors have been generated by looking at the Log pane.

> **Note:**
>
> After the database is successfully converted to Unicode, you must restart the database instance to synchronize any in-memory character set metadata information with the new Unicode database character set.

> **See Also:**
>
> "Converting the Database" for more information about the conversion GUI

3

Viewing and Setting Object Properties in the DMU

This chapter provides a reference for elements of the user interface of the Database Migration Assistant for Unicode (DMU) and describes how to use them to view and set various object properties.

You can view various properties of a database object, such as database, schema, or table, by right-clicking the object's node in the Navigator pane, and selecting **Properties** from the context menu. A Properties tab tailored for the type of the object will be shown in the client pane of the DMU main window. Each Properties tab can have up to four subtabs: General, Scanning, Readiness, and Converting. To show a subtab, click its name in the left sidebar of the tab.

The subtabs show properties and information pertaining to particular steps in the migration process. The General subtab includes properties relevant to all phases of the migration. The Scanning subtab shows parameters controlling the scanning process and the scan results. The Readiness subtab shows if data contained in the object is ready for the actual conversion step. If the data is not ready, the subtab shows the reason for this. The Converting subtab includes parameters that control the conversion step. The following sections describe properties available on all object property tabs.

If you have changed any property on a Properties tab, click **Apply** to save the change.

3.1 Viewing and Setting Database Properties
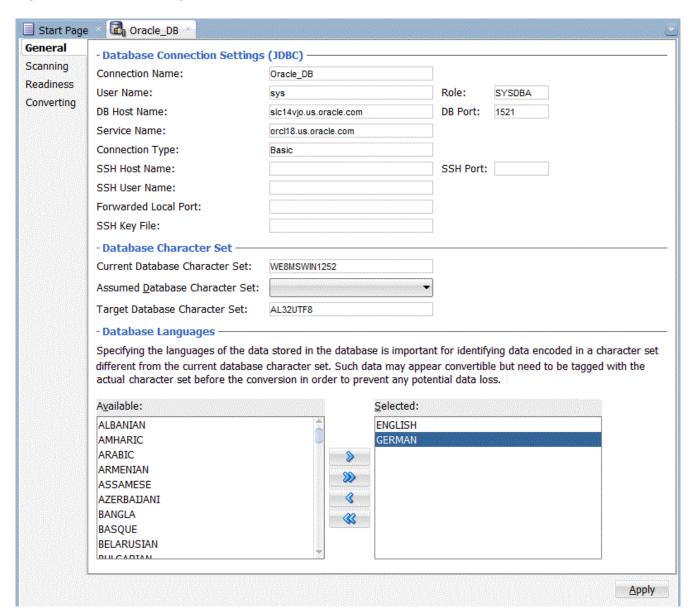
The Database Properties tab has four subtabs: General, Scanning, Readiness, and Converting.

3.1.1 Database Properties: General

On the General subtab of the Database Properties tab, you can specify the connection settings, the character set of the current and target database, and the languages in which the data is stored.

The following screen shot shows the Database Properties: General subtab.

Figure 3-1   Database Properties: General

The properties on the General subtab are as follows:

- Database Connection Settings (JDBC)

  Database connection settings to connect to the database using Java Database Connectivity (JDBC) API. The connection details are described in "Creating a Database Connection". The connection details can be changed only as described in that section.

- Database Character Set

  The database character set information is composed of three properties:

  – Current Database Character Set

    This is the current declared character set of the database that is used to interpret data stored in columns of the data types `VARCHAR2`, `CHAR`, `LONG`, and `CLOB`.

  – Assumed Database Character Set

The DMU uses the assumed database character set as the default source character set when scanning and converting character data during the migration. If the database has been used in a correct character set configuration, then the assumed character set should be the same as the database character set. However, in a pass-through configuration, as described in "Invalid Binary Storage Representation of Data", all or almost all character columns might store data in the character set of client workstations, which might be different from the declared database character set. In such a case, after you identify the real character set of the database contents, set it as the value of the Assumed Database Character Set property to let the DMU know how to correctly interpret the data.

– Target Database Character Set

This is the target database character set of the migration, either AL32UTF8 or UTF8. You can change this property only by uninstalling the DMU repository and installing it again.

> **✎ Note:**
>
> Oracle recommends using AL32UTF8 as the target database character set. Use UTF8 only if you need to have compatibility with certain applications, such as Oracle Applications 11*i*. AL32UTF8 and UTF8 character sets are not compatible with each other as they have different maximum character widths. AL32UTF8 has a maximum character width of 4 bytes, whereas UTF8 has a maximum character width of 3 bytes.

• Database Languages

Select all the possible languages in which data is stored in the database. This information is used to check whether you may have language data stored in the database that is not supported by the database character set.

If the selected language is not supported by the database character set or the assumed character set, then a warning message is displayed while saving the selected language. For example, if the database character set is `WE8MSWIN1252` and you selected Japanese, Traditional Chinese, or Simplified Chinese language, then a warning message is displayed while saving this information.

Click **Apply** to save any changes made on this subtab.
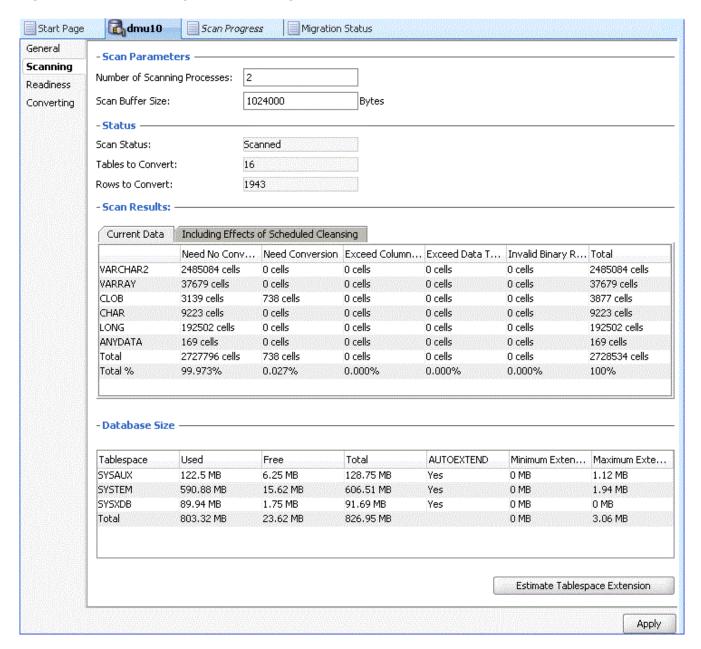
3.1.2 Database Properties: Scanning

On the Scanning subtab of the Database Properties tab, you can set general parameters that control scanning in this database. You can also view the aggregated scan results for the database.

See "Figure 3-2" for an illustration of the scanning subtabs.

Figure 3-2　Database Properties: Scanning

The properties on the scanning subtab are:

- Number of Scanning Processes

  This specifies the number of concurrent processes used to scan the database. Each scanning process consists of a parallel thread in the DMU and a database session on the server created by this thread. The default value of the property is derived from the number of CPUs on the database server. You can tune this value by changing it and measuring the time required to complete one database scan. If you ask the DMU to scan only a small number of small tables, the number of processes used for this scan might be lower than the value of this property.

  You can also change the Number of Scanning Processes in the Scan Wizard. See "Overview of the Database Scan Report" for more information.

- Scan Buffer Size

  This property controls the size in bytes of a buffer that the DMU allocates in each database server session to scan a table. The default value is 1000 kilobytes. The total buffer space used in a single scan is the number of scanning processes times the value of this property. Increasing the value of the property might speed up scanning but only as long as the allocated buffer memory fits into the available RAM on the database server.

  You can also change the Scan Buffer Size in the Scan Wizard. See "Scanning the Database with the Scan Wizard" for more information.

- Scan Status

  This property shows the aggregated scan status of the database. The following values are possible:

  – Never scanned

    No scan has been performed since the most recent installation of the DMU repository.

  – In progress

    A scan is currently in progress.

  – Scanned

    All tables in the database have been scanned and have valid scan results.

  – Partially scanned

    Some tables in the database have been scanned and have valid scan results, but the remaining tables either have never been scanned or their scan results have been invalidated by a cleansing action or by modification of their structure (metadata) outside of the DMU.

  – Issues found

    One or more tables in the database contain data with convertibility issues. These may be length limit (expansion) issues, invalid binary representation issues, or convertible data in the data dictionary.

  – Failed

    One or more tables in the database could not be scanned due to an unexpected error.

- Tables to Convert

  This is the number of tables in the database that are already identified as requiring conversion during the conversion step.

- Rows to Convert

  This is the sum of Rows to Convert property values for all tables requiring conversion in the database.

- Scan Results

  The scan results illustrate the classification of all scanned data cells in all tables in the database into categories of convertibility. A cell is a value of a given column in a given row. The categories are:

  – Invalid binary representation

    The binary representation of the cell data is invalid under the current database character set (the converted value would contain replacement characters). This means that either the assumed character set of the column is incorrect, or the data is binary (for example, a JPEG image, an encryption result, or a text document in binary format is stored in the cell), or there are some corrupted character codes in the value.

- Exceed data type limit

  The cell data will be too long for its data type after conversion.

- Exceed column limit

  The cell data will not fit into a column after conversion.

- Need conversion

  The cell data must be converted, because its binary representation in the target character set is different than the representation in the current character set, but neither length limit issues nor invalid representation issues have been found.

- Need no conversion

  The data is fine, because the binary representation of the data does not change in the conversion.

The cells are assigned twice into the preceding categories. The first classification, displayed under the "Current data" heading, is based on the current column definitions in the database. The second classification, displayed under the "Including effects of scheduled cleansing" heading, is based on column definitions that will result from application of the scheduled cleansing actions. Because only the second classification corresponds to errors that the data would really cause in the conversion step, this classification is used to determine if the database contents are ready for conversion. You can use the first classification as a reference to tell the convertibility status of the database contents in case all scheduled cleansing actions are deleted.

- Database Size

  The scan result includes details about the different tablespaces:

  - Tablespace

    The name of the tablespace that was scanned.

  - Used

    The size of space in megabytes that is used.

  - Free

    The size of space in megabytes that is free.

  - Total

    The total size in megabytes of the tablespace.

  - AUTOEXTEND

    Whether the tablespace is set to automatically extend itself by a specified amount when it reaches its size limit.

  - Minimum Extension

    The minimum size in megabytes that the tablespace must be extended in order to accommodate the conversion space requirements, which assumes the best case scenario where the tables are converted sequentially one by one. It is calculated by taking into account the post-conversion data size expansion and the temporary space requirement of the largest table converted using the CTAS method.

    An empty value indicates the tablespace is not affected by the database conversion. A value zero indicates that the tablespace has enough unused space to meet the minimum conversion space requirement. A value greater than zero indicates the tablespace must be extended by that amount to meet the minimum conversion space requirement. Note that if the tablespace is auto-extensible and the device has enough

free space to accommodate the extension size, then it does not need to be extended before starting the database conversion.

– Maximum Extension

The maximum size in megabytes that the tablespace must be extended in order to accommodate the conversion space requirements, which assumes the worst case scenario where the largest tables using the CTAS conversion method are converted at the same time. It is calculated by taking into account the post-conversion data size expansion and the temporary space requirements of the first N largest tables converted using the CTAS method (N is the number of conversion worker threads).

An empty value indicates the tablespace is not affected by the database conversion. A value zero indicates that the tablespace has enough unused space to meet the maximum conversion space requirement. A value greater than zero indicates the tablespace must be extended by that amount to meet the maximum conversion space requirement. Note that if the tablespace is auto-extensible and the device has enough free space to accommodate the extension size, then it does not need to be extended before starting the database conversion.

• Estimate Tablespace Extension

Click this button to have the DMU calculate the minimum and maximum tablespace extensions for all tablespaces. Use the results as a guideline to size the tablespaces accordingly. If enough storage is available, extending the tablespaces close to the maximum extension sizes will reduce the chance of space allocation errors during the conversion phase.

Note that the DMU does not have information of inline `CLOB` data that remain inline when they are converted to Unicode. For the sake of simplicity, the DMU assumes that all `CLOB` data are stored in `LOB` segments upon estimating the size of post-conversion segments.
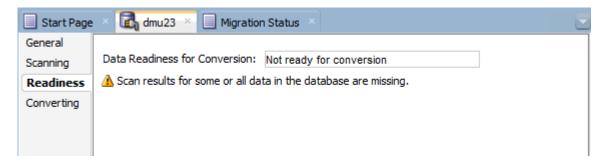
Click **Apply** to save any changes made on this subtab.

3.1.3 Database Properties: Readiness

The Readiness subtab of the Database Properties tab shows the readiness of data for conversion.

The following screen shot illustrates the Database Properties: Readiness subtab.

Figure 3-3   Database Properties: Readiness

The Data Readiness for Conversion property may have one of the following values:

• Does not need conversion

All data in the database is classified as needing no conversion, and the database passed all conversion feasibility tests described in "Preparing the Conversion".

- Ready for conversion

  All data in the database is classified as either needing no conversion, or needing conversion, and the database passed all conversion feasibility tests.

- Not ready for conversion

  Some data in the database is classified as having invalid binary representation or exceeding length limits, or the database has not passed some conversion feasibility tests.
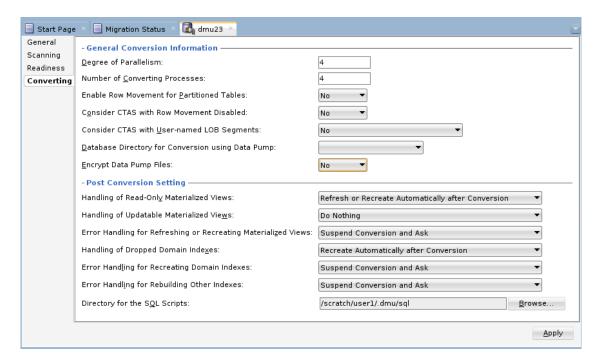
If the readiness status is "Not ready for conversion", additional information is displayed to explain the problem.

3.1.4 Database Properties: Converting

The Converting subtab of the Database Properties tab enables you to control aspects of the conversion.

The following screen shot illustrates the Database Properties: Converting subtab.

Figure 3-4   Database Properties: Converting

The following properties can be set in the Converting subtab:

- Degree of Parallelism

  If the DMU converts a table using a `CREATE TABLE AS SELECT` (CTAS) statement, this property sets the degree of parallelism that the parallel execution feature of the database should use for the operation.

- Number of Converting Processes

  This specifies the number of concurrent processes used to convert the database. Each converting process consists of a parallel thread in the DMU and a database session on the

server created by this thread. The default value of the property is derived from the number of CPUs on the database server.

*   Enable Row Movement for Partitioned Tables

    Conversion of a partitioning key column value in a row of a range or hash partitioned table could cause the converted key to point to a partition other than the one currently containing the row. The database must move the row from the old partition to the new one, if the update of the key value succeeds. For reasons described for the Consider CTAS with Row Movement Disabled property, row movement is allowed only for tables for which it was enabled explicitly with the statement `ALTER TABLE ENABLE ROW MOVEMENT`. You can set the property Enable Row Movement for Partitioned Tables to **Yes** to allow the DMU to temporarily enable row movement for partitioned tables that have partitioning key columns that require conversion.

*   Consider CTAS with Row Movement Disabled

    Because converting a table using a CTAS statement changes the physical addresses (rowids) of rows in the table, applications that store those addresses permanently could fail to locate the rows after the conversion. Therefore, by default, the DMU does not assign the CTAS conversion method to a table unless movement of rows in the table has been explicitly allowed with the statement `ALTER TABLE ENABLE ROW MOVEMENT`. On the other hand, the row movement is disabled by default for a new table if it is not enabled explicitly in the `CREATE TABLE` statement. Therefore, most tables in most databases do not allow rows to be moved even though rowids are seldom stored by applications (primary keys are the recommended way to reference rows). If you know that applications connecting to the database do not store rowids, you can set the Consider CTAS with Row Movement Disabled property to **Yes** to allow the DMU to also assign the CTAS conversion method to tables that do not have the row movement enabled.

*   Consider CTAS with User-named LOB Segments

    By default, the DMU does not assign the CTAS conversion method for tables containing user-named LOB segments because the LOB segment names cannot be preserved with the CTAS conversion method. If there is no need to preserve the user-specified LOB segment names, you can set this property to **Yes**, which will allow the CTAS conversion method to be considered and the segments to be renamed after the conversion by appending a "$DMU" suffix.

*   Database Directory for Conversion using Data Pump

    This property must be set if you select the "Convert using Data Pump" conversion method for any tables. It specifies the directory in which to store Oracle Data Pump export files during conversion. The drop-down list displays all directory objects defined in the database. If a suitable directory is not displayed, you can disconnect the DMU from the database, log in to the database and create a directory object using the `CREATE DIRECTORY` command, reconnect the DMU to the database, and choose the new directory object in the drop-down list. Because all tables designated for the "Convert using Data Pump" method must be exported to the directory before the database character set is changed, the corresponding operating system directory must be large enough to hold the sum of the sizes of all such tables.

    If a directory named `DATA_PUMP_DIR` exists in the database and is writable by the connected user, then the DMU selects this directory as the default value for this property. Otherwise, the default selection is empty.

*   Encrypt Data Pump Files

    This property applies only to tables that use the "Convert using Data Pump" conversion method. It specifies whether to encrypt the Oracle Data Pump export files. If you set this property to **Yes**, then the export files for all tables that use the "Convert using Data Pump"

method will be transparently encrypted using TDE encryption. The DMU chooses the encryption algorithm. If a table contains encrypted data, the DMU ensures that its export file is encrypted using at least the highest encryption algorithm that is used in the table. If you set the property **No**, then the export files for all tables that use the "Convert using Data Pump" conversion method will be unencrypted.

The default value for this property is calculated during repository installation. If there is at least one encrypted object present in the database, then the default value is **Yes**. Otherwise, the default value is **No**.

*   Handling of Read-Only Materialized Views

    This property determines how to handle read-only materialized views after their master tables have been converted. The choices are:

    –   Refresh Automatically After Conversion

        Refresh the read-only materialized views automatically in the post-conversion phase. This is the default option.

    –   Generate SQL Script

        Instead of refreshing the materialized views automatically, generate a SQL script containing the statements to refresh the read-only materialized views which can be executed subsequently outside of the DMU.

    –   Do Nothing

        Take no action on read-only materialized views.

*   Handling of Updatable Materialized Views

    This property determines how to handle updatable materialized views after their master tables have been converted. The choices are:

    –   Refresh Automatically After Conversion

        Refresh the updatable materialized view automatically in the post-conversion phase.

    –   Generate SQL Script

        Instead of refreshing the materialized views automatically, generate a SQL script containing the statements to refresh the updatable materialized views which can be executed subsequently outside of the DMU.

    –   Do Nothing

        Take no action on updatable materialized views. This is the default.

*   Error Handling for Refreshing Materialized Views

    This property determines how errors for refreshing materialized views are handled. The choices are:

    –   Suspend Conversion and Ask

        Suspend the failing SQL statement and let the user choose the appropriate action. This is the default option. You will be able to choose from retrying the failing statement, exporting the statement to an external script and continue, or skip the statement without exporting.

    –   Skip Failing SQL and Continue

        Automatically skip any failing SQL statements and continue with the conversion.

    –   Export Failing SQL to Script and Continue

        Automatically export any failing SQL statements to an external script that can be executed outside of the DMU and continue with the conversion.

- Handling of Dropped Domain Indexes

  This property determines how to handle dropped domain indexes. The choices are:

  – Recreate Automatically After Conversion

    Recreate the dropped domain index automatically in the post-conversion phase. This is the default option.

  – Generate SQL Script

    Instead of recreating the dropped domain index, generate a SQL script containing the statements to recreate the dropped domain index that can be executed subsequently outside of the DMU.

  – Do Nothing

    Take no action on dropped domain indexes.

- Error Handling for Recreating Domain Indexes

  This property determines how errors for recreating domain indexes are handled. The choices are:

  – Suspend Conversion and Ask

    Suspend the failing SQL statement and let the user choose the appropriate action. This is the default option. You will be able to choose from retrying the failing statement, exporting the statement to an external script and continue, or skip the statement without exporting.

  – Skip Failing SQL and Continue

    Automatically skip any failing SQL statements and continue with the conversion.

  – Export Failing SQL to Script and Continue

    Automatically export any failing SQL statements to an external script that can be executed outside of the DMU and continue with the conversion.

- Error Handling for Rebuilding Other Indexes

  This property determines how errors for rebuilding other indexes are handled. The choices are:

  – Suspend Conversion and Ask

    Suspend the failing SQL statement and let the user choose the appropriate action. This is the default option. You will be able to choose from retrying the failing statement, exporting the statement to an external script and continue, or skip the statement without exporting.

  – Skip Failing SQL and Continue

    Automatically skip any failing SQL statements and continue with the conversion.

  – Export Failing SQL to Script and Continue

    Automatically export any failing SQL statements to an external script that can be executed outside of the DMU and continue with the conversion.

- Directory for the SQL Scripts and its location (Browse button)

  Displays the location and a Browse button to navigate where you want to put the output SQL script.

  The default name for a read-only materialized view refresh script resembles:

  – RefreshMV_*connectionName_timestamp*.sql

The default name for an updatable materialized view refresh script resembles:

– RefreshUpdMV_*connectionName_timestamp*.sql

The default name for a domain index recreation script resembles:

– DomainIndexDDL_*connectionName_timestamp*.sql

The default name for a script with failed materialized view refresh SQL statements resembles:

– FailedMVRefresh_*connectionName_timestamp*.sql

The default name for a script with failed domain index recreation SQL statements resembles:

– FailedDomainIndex_*connectionName_timestamp*.sql

The default name for a script with failed other index rebuild SQL statements resembles:

– FailedOtherIndex_*connectionName_timestamp*.sql

Note that changing the script directory location during the conversion will take effect for the next SQL script to be created.

Click **Apply** to save any changes made on this subtab.

3.2 Viewing and Setting Schema Properties

The Schema Properties tab has three subtabs: General, Scanning, and Readiness. There are no schema-level properties related to conversion.

3.2.1 Schema Properties: General

The General subtab of the Schema Properties tab shows the name of the schema and the default tablespace for storage objects, such as tables, created in this schema.

Note that these properties are read-only.

3.2.2 Schema Properties: Scanning

On the Scanning subtab of the Schema Properties tab, you can view the aggregated scan results for objects in this schema.

The properties on the Scanning subtab are as follows:

• Scan Status

This property shows the aggregated scan status of the schema. The following values are possible:

– Never scanned

No table has been scanned in this schema since the most recent installation of the DMU repository.

– In progress

A table in the schema is currently being scanned.

– Scanned

All tables in the schema have been scanned and have valid scan results.

– Partially scanned

Some tables in the schema have been scanned and have valid scan results, but the remaining tables either have never been scanned or their scan results have been invalidated by a cleansing action or by modification of their structure (metadata) outside of the DMU.

– Results invalidated

Scan results of all tables in the schema have been invalidated by a cleansing action or by modification of the table structure (metadata) outside of the DMU.

– Issues found

One or more tables in the database contain data with convertibility issues. These could be length limit (expansion) issues, invalid binary representation issues, or, in the case of data dictionary schemas, presence of convertible data.

– Failed

One or more tables in the schema could not be scanned due to an unexpected error.

• Tables to Convert

This is the number of tables in the schema that are already identified as requiring conversion in the conversion step.

• Rows to Convert

This is the sum of Rows to Convert property values for all tables requiring conversion in the schema.

• Scan Results

The scan results illustrate the classification of all scanned data cells in tables of the schema into conversion categories. A cell is a value of a given column in a given row. The classification is the same as that for the database except that the results are summed up for all tables in the schema, not for all tables in the database – see "Scanning the Database".

All the preceding properties are read-only.

3.2.3 Schema Properties: Readiness

This subtab shows the readiness of data for conversion.

The Data Readiness for Conversion property might have one of the following values:

• Does not need conversion

All data in the schema is classified as needing no conversion.

• Ready for conversion

All data in the schema is classified as either needing no conversion, or needing conversion. The data has no conversion issues.

• Not ready for conversion

Some data in the schema is classified as having invalid binary representation or exceeding length limits, or some scan results are missing.

If the readiness status is "Not ready for conversion", additional information is displayed to explain the problem.

3.3 Viewing and Setting Table Properties

The Table Properties tab has four subtabs: General, Scanning, Readiness, and Converting. The Converting subtab may be hidden if the table is not ready for conversion or it requires no conversion.

3.3.1 Table Properties: General

The General subtab of the Table Properties tab shows properties such as table and schema name.

The subtab shows the following properties of a table:

- Table Name

  This is the name of the table.

- Schema Name

  This is the name of the schema to which the table belongs.

- Tablespace

  This is the name of the tablespace that contains the table. For partitioned tables, the property shows the default tablespace for new partitions.

- Table Size

  This is the size of the table as determined by its highwater mark.

- Columns That May Contain Text

  This is the list of columns in the table that might contain character data in the database character set and therefore might require conversion. The list shows column names, column data types, length constraints, the presence of a `NOT NULL` constraint, and the information if a column belongs to the primary key of the table.

These properties are read-only.

3.3.2 Table Properties: Scanning

On the Scanning subtab of the Table Properties tab, you can control the scanning of the table, and view the aggregated scan results for columns of this table.

The properties on the Scanning subtab are as follows:

- Available Rowids

  This property tells if the last scan of the table collected rowids to identify rows containing cell data of a specific type. The possible values are:

  – None

    No rowids have been collected for this table.

  – All to convert

    Rowids of all rows containing at least one column value that requires conversion have been collected. These rowids are required for the conversion method "Update only convertible rows".

  – With issues

Only rowids of rows with at least one column value with conversion issues have been collected. These rowids might improve effectiveness of working with the Cleansing Editor. See Using the DMU to Cleanse Data.

• Rowids to Collect

You can set this property to tell the DMU which rowids to collect during the next scan of the table. The possible values of this property are the same as for the Available Rowids property.

You can override the value of this property in the Scan Wizard. See "Scanning the Database".

• Split over Threshold

Set this property to **Yes** to let the DMU divide the table into multiple chunks and then scan the chunks in parallel by multiple scanning processes. The DMU will split the table only if it is larger than an internally calculated threshold.

• Done split

The value of this property tells if the last scan of the table was performed on multiple chunks in parallel.

• Scan Status

This property shows the aggregated scan status of the table. The following values are possible:

– Never scanned

No column of this table has been scanned since the most recent installation of the DMU repository.

– In progress

The table is currently being scanned.

– Scanned

All columns in the table have been scanned and have valid scan results.

– Partially scanned

Some columns in the table have been scanned and have valid scan results, but the remaining columns either have never been scanned or their scan results have been invalidated by a cleansing action or by modifications of their structure (metadata) outside of the DMU.

– Results invalidated

Scan results of all columns of the table have been invalidated by a cleansing action or by modification of the table structure (metadata) outside of the DMU.

– Issues found

One or more columns in the table contain data with conversion issues. These could be length limit (expansion) issues, invalid binary representation issues, or, in the case of data dictionary tables, presence of convertible data.

– Failed

The last scan of columns of this table failed due to an unexpected error.

• Rows to Convert

This is the number of rows in the table containing at least one column value that requires conversion.

- Scan Results

  The scan results illustrate the classification of all scanned data cells in the table into conversion categories. A cell is a value of a given column in a given row. The classification is the same as that for the database except that the results are summed up for all columns of the table, not for all tables in the database. See "Scanning the Database".

  The scan results of columns whose data type is nested table are not added to the results of the table containing the column. The DMU presents the storage tables of nested table columns as separate tables.

Click **Apply** to save any changes made on this subtab.

3.3.3 Table Properties: Readiness

This subtab shows the readiness of table data for conversion.

The Data Readiness for Conversion property might have one of the following values:

- Does not need conversion

  All data in the table is classified as needing no conversion.

- Ready for conversion

  All data in the table is classified as either needing no conversion, or needing conversion. The data has no conversion issues.

- Not ready for conversion

  Some data in the table is classified as having invalid binary representation or exceeding length limits, or some scan results are missing.

If the readiness status is "Not ready for conversion", additional information is displayed to explain the problem.

3.3.4 Table Properties: Converting

Properties on this subtab allow you to control certain aspects of the conversion of the table.

You can set the following properties:

- Conversion Method

  This property decides how the DMU will update data in the table to convert it to the target character set. The possible values are:

  – Exclude from conversion

    The table will not be converted. Only the scheduled cleansing actions might be applied.

  – Copy data using `CREATE TABLE AS SELECT`

    A copy of the table will be created by the CTAS statement and the original table will be dropped. Column values will be converted by the query contained in this statement.

    See "Notes on "Copy data using CREATE TABLE AS SELECT"" for restrictions.

  – Update all rows

    An `UPDATE` statement without a `WHERE` clause will be used to update all rows of the table.

  – Update only convertible rows

An `UPDATE` statement will update only those rows of the table whose rowids have been collected during the last scan of the table.

– Scan and update only convertible rows

An `UPDATE` statement will update only those rows of the table that contain a convertible column value as determined by an internal scanning function included in the `WHERE` clause of the statement.

– Convert using Data Pump

The table will be exported using Oracle Data Pump, then imported back into the database after the database character set has been changed.

See "Notes on "Convert using Data Pump"" for more details.

The DMU automatically assigns one of the preceding conversion methods to each scanned table. The exception is the "Convert using Data Pump" method, which must be assigned by the user. Various features of a given table and the convertibility of its data determine which of the preceding methods are valid for the table. If more than one method is valid and your tests show that one of the alternative methods will be more effective, you can change the automatic assignment by changing the value of this property.

- User-preferred Conversion Method

  This property can be used to override the DMU recommended conversion method if necessary. If it is explicitly set by the user to a specific conversion method, the setting will be honored unless the selected conversion method is not applicable to the current table. The default value is "No preference", which means the DMU recommended conversion method will be used.

- Target Tablespace

  If the conversion method for the table is "Copy data using `CREATE TABLE AS SELECT`", you can select a tablespace from this drop-down list to specify the tablespace in which the converted copy of the table will be created. The default value of this property is the tablespace containing the table.

- Preserve Position of `LONG` Column

  Due to restrictions of the CTAS statement, if the conversion method for the table is "Copy data using `CREATE TABLE AS SELECT`", any `LONG` column of the table must be converted and copied separately from the rest of the table. The default processing applied by the DMU in such a case could change the position of the `LONG` column in the table. This might break applications that select columns from the table using the asterisk syntax (`SELECT * FROM`). If you know about such an application, you can set this property to **Yes** to let the DMU apply a method that is less effective, but that will preserve the position of the `LONG` column.

Click **Apply** to save any changes made on this subtab.

Notes on "Copy data using CREATE TABLE AS SELECT"

DMU does not support the "Copy data using `CREATE TABLE AS SELECT`" conversion method for the following types of tables:

- Tables that are involved in a Change Data Capture process. Neither source tables nor change tables are supported.

- Tables containing ADT or Nested Table columns, and when these columns contain both character semantics and `CLOB` attributes.

- Advanced Queue tables

- Blockchain tables

- Immutable tables

- Updatable or writable materialized view container tables

- Materialized view master tables on which materialized view log is created

- Tables with system partitioning, that is, tables created with `PARTITION BY SYSTEM` clause

- Tables containing a virtual column that is used as a partitioning key column

- Object type tables with primary key OID

- Index-organized tables for which `OVERFLOW` is not specified.

- Tables with `XMLType` columns

- Tables containing user-defined virtual columns

- Tables containing encrypted columns.

- Tables with associated statistics

- Tables with Auto CDR configured

- Auto CDR tombstone tables

- Tables with internal triggers

- Tables with a Data Redaction policy

- Tables involved in a Streams process

- Tables with Row-Level Security or Fine-Grained Auditing policies

- Tables enabled for flashback archiving

- Oracle-supplied tables in a pluggable database in a multitenant container database

- Tables with ADT columns containing character length semantics attributes

- XML token tables whose names begin with `X$NM`, `X$PT`, or `X$QN`

- Tables with an object attribute to be lengthened to target length measured in characters during scheduled cleansing

Notes on "Convert using Data Pump"

Starting with DMU 23.1, you can select the "Convert using Data Pump" conversion method. To improve performance, Oracle recommends this method for tables that contain large amounts of `CLOB` data.

The "Convert using Data Pump" method is never automatically assigned by the DMU; it must be assigned by the user. When you specify this conversion method, tables are exported to a specified database directory using Oracle Data Pump, the database character set is changed, and the tables are imported back into the database using Oracle Data Pump. See Data Conversion: Converting for a complete list of conversion steps.

Prerequisites for "Convert using Data Pump"

The following prerequisites apply to the "Convert using Data Pump" conversion method:

- The Oracle Database version must be 11.2.0.4 or later.

- You must specify an operating system directory in which to store the Oracle Data Pump export files during the conversion:

  1. Choose or create the operating system directory.

The directory must have enough space to simultaneously hold the data for all tables to be converted using Oracle Data Pump.

2. Ensure that the database contains a corresponding directory object. For example:

```
CREATE DIRECTORY dpfiles AS '/disk1/oracle/dpfiles';
```

3. On the Converting subtab of the Database Properties tab, set the "Database Directory for Conversion using Data Pump" property by choosing the directory in the property drop-down list. See Database Properties: Converting.

- You must specify the encryption property for Oracle Data Pump export files. This controls whether Oracle Data Pump export files are transparently encrypted using TDE encryption.

  This property is assigned at the database level and cannot be assigned on a per-table basis. On the Converting subtab of the Database Properties tab, set the "Encrypt Data Pump Files" property to **Yes** or **No**. See Database Properties: Converting.

  When you set "Encrypt Data Pump Files" property to **Yes**:

  – The export files for all tables that use the "Convert using Data Pump" method will be transparently encrypted using TDE encryption. The DMU chooses the encryption algorithm. If a table contains encrypted data, the DMU ensures that its export file is encrypted using at least the highest encryption algorithm that is used in the table.

  – The database software keystore (encryption wallet) must be open during the entire conversion phase. If a database restart occurs during the conversion, be sure to open the software keystore immediately after the restart.

  – For Oracle Database version 11.2.0.4, the `COMPATIBLE` initialization parameter must be set to `11.0.0` or greater. There is no requirement for the `COMPATIBLE` setting for other supported Oracle Database versions.

  When you set "Encrypt Data Pump Files" property to **No**, the Oracle Data Pump export files for all tables marked "Convert using Oracle Data Pump" will be stored unencrypted on the database server.

- You must set the System Global Area (SGA) size for the database to a value that effectively utilizes the "Convert Using Data Pump" conversion method. SGA configuration will vary based on factors such as the size of the database, workload characteristics, and available hardware resources. In scenarios where the buffer cache is configured with only the minimum required SGA, Data Pump operations may encounter issues, such as `ORA-04031` and `ORA-39014`, which could potentially lead to failures in the DMU conversion process and subsequent data loss. Therefore, it is crucial to ensure that the SGA configuration adequately supports Data Pump operations. See "Performance Guidelines for Oracle Data Pump Parameters" in *Oracle Database Utilities* to understand more about Data Pump performance.

Restrictions for "Convert using Data Pump"

The following restrictions apply to tables that use the "Convert using Data Pump" conversion method:

- The table cannot be a blockchain or immutable table.

- The table cannot be a clustered table.

- The table cannot have columns or attributes of the `CHAR` data type that contain convertible data. If the table has a collection data type column that contains convertible data, then the element data type must be recursively checked through to its innermost constituent type to ensure that there is no `CHAR` data type present.

- Because the character conversion is controlled by Oracle Data Pump and not the DMU, some scheduled cleansing actions are not supported. For example, migrating a column to character semantics is supported, but lengthening a column is not supported.

- Setting the assumed column character set to anything other than the database character set is not supported for any columns in the table.

- If invalid binary representation data is present in any column of the table, then the "Allow Conversion of Data with Issues" property of the column must be set to **Yes** and the post-conversion length of the column must be less than or equal to the scheduled column length.

- The "Exclude from Conversion" property must be set to **No** for all columns in the table.

- For Oracle Database releases 11.2 and 12.1, the table size must be smaller than 1.5 TB and the table must not be a system-partitioned table. This restriction is lifted in 12.2 and later releases.

- For Oracle Database release 12.1.0.1, the table cannot reside in a pluggable database (PDB). This restriction is lifted in 12.1.0.2 and later releases.

3.4 Viewing and Setting Column Properties

The Column Properties tab has four subtabs: General, Scanning, Readiness, and Converting. The Converting subtab might be hidden if the column is not ready for conversion or it requires no conversion.

3.4.1 Column Properties: General

The General subtab of the Column Properties tab shows various read-only properties columns, including column names and types.

The subtab shows the following properties:

- Column Name

  This is the name of the column.

- Column Type

  This is the data type of the column.

- Column Length

  This is the length constraint of the column. Only `VARCHAR2` and `CHAR` columns might have a length constraint.

- May Contain `NULL`

  If the value of this property is Yes, the column might contain `NULL` values. Otherwise, there is a `NOT NULL` or `PRIMARY KEY` constraint disallowing `NULL` values in the column.

- Belongs to Primary Key

  If the value of this property is Yes, the column belongs to a `PRIMARY KEY` constraint.

3.4.2 Column Properties: Scanning

On the Scanning subtab of the Column Properties tab, you can view the scan results for the column.

The properties on the Scanning subtab are:

- Assumed Column Character Set

  The value of this property shows the assumed character set of the column set in the Cleansing Editor. See Using the DMU to Cleanse Data.

- Scan Status

  This property shows the scan status of the column. The following values are possible:

  – Never scanned

    No column of this table has been scanned since the most recent installation of the DMU repository.

  – In progress

    The table is currently being scanned.

  – Scanned

    All columns in the table have been scanned and have valid scan results.

  – Results invalidated

    The scan results of the column have been invalidated by a cleansing action or by modification of the table structure (metadata) outside of the DMU.

  – Issues found

    The column contains data with conversion issues. These could be length limit (expansion) issues, invalid binary representation issues, or, in the case of data dictionary tables, presence of convertible data.

  – Failed

    The last scan of the column failed due to an unexpected error.

- Maximum Pre-Conversion Length

  This property shows the length in bytes of the longest current value in the column.

- Maximum Post-Conversion Length

  This property shows the length in bytes of the longest value in the column after it will be converted to the target character set.

- Scan Results

  The scan results illustrate the classification of all scanned data cells (row values) in the column into conversion categories. The classification is the same as that for the database except that the results provided are for a single column only, not for all columns in the database. See "Viewing and Setting Database Properties".

The properties on this subtab are read-only.

3.4.3 Column Properties: Readiness

The Readiness subtab of the Column Properties tab indicates which actions are scheduled and whether the data is ready for conversion.

The subtab contains the following properties:

- Scheduled Cleansing Action

  This property shows the scheduled cleansing action defined for the column in the Cleansing Editor. See Using the DMU to Cleanse Data .

- Data Readiness for Conversion

This property indicates whether data in the column is ready for conversion. It might have one of the following values:

– Does not need conversion

All data in the column is classified as needing no conversion.

– Ready for conversion

All data in the table is classified as either needing no conversion, or needing conversion. The data has no conversion issues.

– Has exceptional data

Some data in the column is classified as having invalid binary representation or exceeding length limits.

If the readiness status is "Not ready for conversion," then additional information is displayed to explain the problem.

3.4.4 Column Properties: Converting

Properties on this subtab enable you to control certain aspects of the conversion of the column.

The following property can be set:

• Exclude from Conversion

If you set this property to **Yes**, the DMU does not update the data in this column.

• Allow Conversion of Data with Issues

If the scan results of the column show that some values have conversion issues, that is, the converted values will contain replacement characters or they will be truncated, you can still let the DMU convert the data by setting this property to **Yes**. This might be useful if you want to automatically truncate data that exceeds the column or data type limit or you are not concerned about a few corrupted values that happen to exist in the column but are of no real significance for applications that use this data. Be careful not to set the property to **Yes** just to avoid the process of cleansing the data.

Click **Apply** to save any changes made on this subtab.

4

Performing Basic DMU Tasks

This chapter shows the step-by-step use of the Database Migration Assistant for Unicode (DMU) in a few typical scenarios of the character set migration process.

If you follow the examples presented here, you can then use the tool to migrate your database. Most of the steps in this chapter are common throughout all scenarios.

4.1 Initializing the Database

Initializing a database for use with the DMU requires several installation and creation tasks.

4.1.1 Installing Required Patches

For your database release to work with the DMU, you may need to install database patches.

Patches are available at `https://support.oracle.com`.

To identify required patches and install them:

1. Refer to the latest DMU release notes to determine whether database patches are required for your database release to work with the DMU.

2. Use the `opatch` utility to check whether the required patches are already installed.

3. Set the `ORACLE_HOME` environment variable to the Oracle home directory of the database, change the current directory to the `OPatch` subdirectory of the Oracle home, and then issue the following statement:

   ```
   opatch lsinventory -patch
   ```

4. If the `opatch` utility identifies the lack of a valid `oraInst.loc` file, then locate the file in the Oracle home of the database and pass its location in the `-invPtrLoc` parameter, as shown in the following example:

   ```
   opatch lsinventory -patch -invPtrLoc ../oraInst.loc
   ```

   The output from the `opatch` utility will resemble the following:

   ```
   Invoking OPatch 10.2.0.5.0


   Oracle interim Patch Installer version 10.2.0.5.0
   Copyright (c) 2005, Oracle Corporation.  All rights reserved..


   Oracle Home       : C:\oracle\product\10.2.0\db_1
   Central Inventory : C:\Program Files\Oracle\Inventory
      from           : n/a
   OPatch version    : 10.2.0.5.0
   ```

```
OUI version        : 10.2.0.5.0
OUI location       : C:\oracle\product\10.2.0\db_1\oui
Log file location :
C:\oracle\product\10.2.0\db_1\cfgtoollogs\opatch\opatch2010-05-17_22-50-28P
M.log

Lsinventory Output file location :
C:\oracle\product\10.2.0\db_1\cfgtoollogs\opatch\lsinv\lsinventory2010-05-1
7_22-50-28PM.txt


--------------------------------------------------------------------

Interim patches (2) :

Patch  5556081     : applied on Mon Feb 08 15:50:52 CET 2010
   Created on 9 Nov 2006, 22:20:50 hrs PST8PDT
   Bugs fixed:
     5556081

Patch  5557962     : applied on Mon Feb 08 15:50:45 CET 2010
   Created on 9 Nov 2006, 23:23:06 hrs PST8PDT
   Bugs fixed:
     4269423, 5557962, 5528974


--------------------------------------------------------------------

OPatch succeeded.
```

5. Check whether the required database patches are listed among the interim patches.

6. If you discover missing patches, then download them from the My Oracle Support site and install them.

7. Review the documentation that is part of the patches for installation instructions.

4.1.2 Installing Supporting Packages

If your database has not been initialized for the DMU, then it will not contain the PL/SQL supporting packages needed to access dedicated database kernel functions.

Perform the following steps if installation documentation from any installed database patches requires them.

To install supporting packages:

1. Connect to the database server host as an operating system user who can run the SQL*Plus utility from the Oracle home directory of the database.

2. Start the SQL*Plus utility and connect as a user with the SYSDBA privilege. For example, on Linux or Microsoft Windows, enter the following command:

```
sqlplus / as sysdba
```

3. After you have logged in, execute the following command:

```
SQL> @?/rdbms/admin/prvtdumi.plb
```

The output from the script should resemble the following:

```
Library created.

Package created.

No errors.

Package body created.

No errors.
```

4.1.3 Creating a Tablespace for the DMU Repository

For ease of maintenance and to avoid fragmentation of other production tablespaces, Oracle recommends that you use a separate tablespace for the database objects forming the DMU repository.

To create a tablespace:

1. Log in to the database as a user with the `SYSDBA` privilege.

2. Ensure that the amount of free space available in the tablespace is sufficient to hold the repository contents.

   To determine the initial size of the tablespace, run the following SQL query:

```
SELECT  CEIL((t.cnt*300+c.cnt*1000)/1048576)||' MB' "Initial Size"
FROM    (SELECT COUNT(*) cnt FROM SYS.TAB$) t,
        (SELECT COUNT(*) cnt
         FROM   SYS.COL$
         WHERE  OBJ# IN (SELECT OBJ# FROM SYS.TAB$)
         AND    BITAND(property,65536)=0
         AND    TYPE# IN (1,8,58,96,112)
         AND    CHARSETFORM=1) c ;
```

   The space required by the repository is difficult to estimate. The size of the tablespace could grow significantly if you collect a lot of rowids for use by the Cleansing Editor or if you apply the "Convert only updatable rows" conversion method to many tables with large numbers of convertible cells. Therefore, Oracle recommends that the tablespace be locally managed with the default extent allocation policy and with the autoextension feature enabled. This will allow the tablespace to grow, if necessary. Define the maximum size constraints for the data files to avoid exhausting the entire disk space.

   Note that the tablespace for installing the DMU repository must have a block size of 8K or above.

3. Create a tablespace for the DMU repository, as explained in *Oracle Database 2 Day DBA*.

4.1.4 Creating a Database Connection

You must create a connection to the database that you want to analyze or migrate.

See "Creating a Database Connection".

4.2 Refreshing, Reinstalling, Upgrading, and Uninstalling the DMU Repository

After the DMU repository is installed, it is automatically refreshed at predefined points in the workflow to account for any objects containing character data that may have been added to, altered in, or removed from the database. You can also refresh, reinstall, upgrade, or uninstall the repository by selecting the appropriate option from the Migration menu.

4.2.1 Refreshing the DMU Repository

You can manually refresh the DMU repository to immediately include any recent changes to the database objects.

To refresh the DMU repository:

- In the Migration menu or the context menu of the database node in the Navigator pane, select **Refresh DMU Repository**.

  The message about the successful refresh of the DMU repository is displayed once the refresh operation is completed.

4.2.2 Reinstalling the DMU Repository

You can explicitly reinstall the DMU repository.
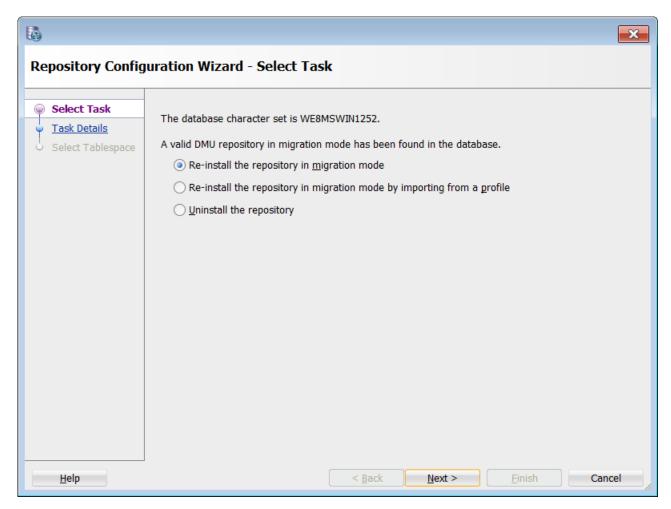
To reinstall the DMU repository:

1. In the Migration or Validation menu, select **Configure DMU Repository**.

   Figure 4-1 appears.
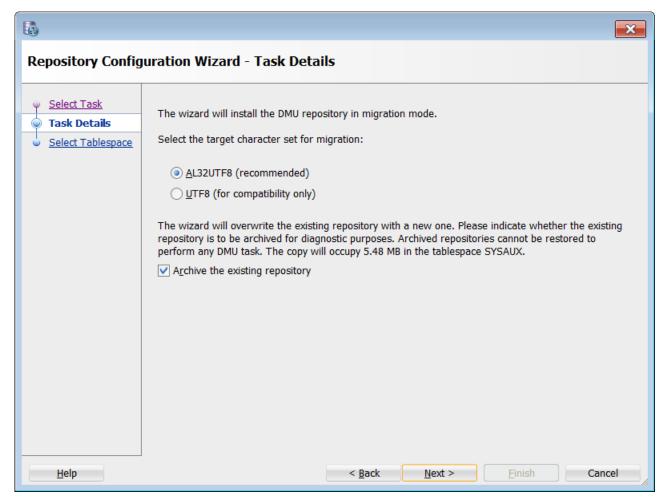
Figure 4-1   Repository Configuration Wizard - Reinstallation

2. Choose either migration mode or validation mode, and then click **Next**.

    Figure 4-2 appears.
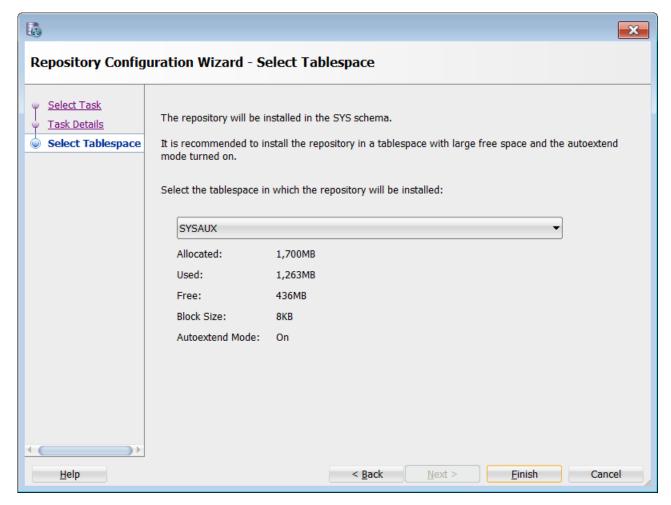
Figure 4-2   Repository Configuration Wizard - Reinstallation/Archiving

3. Select the target character set for migration. Ensure that "Archive the existing repository" is selected, and then click **Next**.

    Figure 4-3 appears.

Figure 4-3   Repository Configuration Wizard - Select Tablespace

4. Click **Finish**.

    The DMU repository is reinstalled.

4.2.3 Upgrading the DMU Repository to 23.1

When connecting DMU 23.1 to a database whose repository was installed by the DMU 19.1 or 2.1 software, the Repository Configuration Wizard enables you to upgrade the repository to DMU 23.1.

You must have a valid DMU 23.1 repository installed before you can perform migration tasks with the DMU 23.1 software.

To upgrade the repository from DMU version 19.1 or 2.1 to version 23.1:

1. From the Migration or Validation menu, select **Configure DMU Repository**.

    The Select Task page of the Repository Configuration Wizard is displayed.

2. On the Select Task page, select the following options:

    - For the Migration mode, select **Upgrade an older version of the repository in migration mode**.

    - For the Validation mode, select **Upgrade an older version of the repository in validation mode**.

3. Click **Next**.

A progress bar showing the progress of the DMU repository upgrade process is displayed.

After successful completion of the DMU repository upgrade process, a dialog box containing the following message is displayed:

```
Repository has been upgraded successfully.
```

4.2.4 Uninstalling the DMU Repository

You can uninstall the DMU repository.

To remove the DMU repository:

1. From the Migration menu or the context menu of the database node in the Navigator pane, select **Configure DMU Repository**.

A dialog box appears providing you the option to uninstall the repository.

2. Select **Uninstall the repository**, and then click **Next**.

The DMU repository is uninstalled.

4.3 Scanning the Database

In the scanning step, you analyze problem causes and choose a cleansing strategy to resolve data issues.

Scanning performs the following steps:

1. Reading character values from the database

2. Converting these values to the target character set

3. Counting how many values meet the following criteria:

    - Change in conversion

    - Do not fit into their columns

    - Do not fit into their data types

    - Contain valid character codes

The scan phase also calculates additional statistics, such as the maximum post-conversion length of values in a column. The DMU stores the calculated counts and statistics in the DMU repository as scan results. The character values resulting from the test conversion are themselves discarded. They are not stored permanently in the database.

Before a database can be converted to Unicode, the DMU must analyze the character data in the VARCHAR2, CHAR, LONG, and CLOB table columns to assess if any issues could prevent the conversion from finishing successfully without causing data corruption. The DMU analyzes the data by converting character column values in the database from their declared character set to the target Unicode character set, and checking each value to determine if any of the following conditions is true:

- The conversion result differs from the original value.

- The conversion result fits into the length limit of its column.

- The conversion result fits into its data type.

- The conversion result does not contain any replacement characters, that is, each converted source character code is valid in the declared character set of the column.

The scanner also determines if the key length of any index defined on character columns exceeds the maximum allowed key size because of scheduled cleansing actions or the increased post-converted data size of the column.

4.3.1 Setting Database Properties
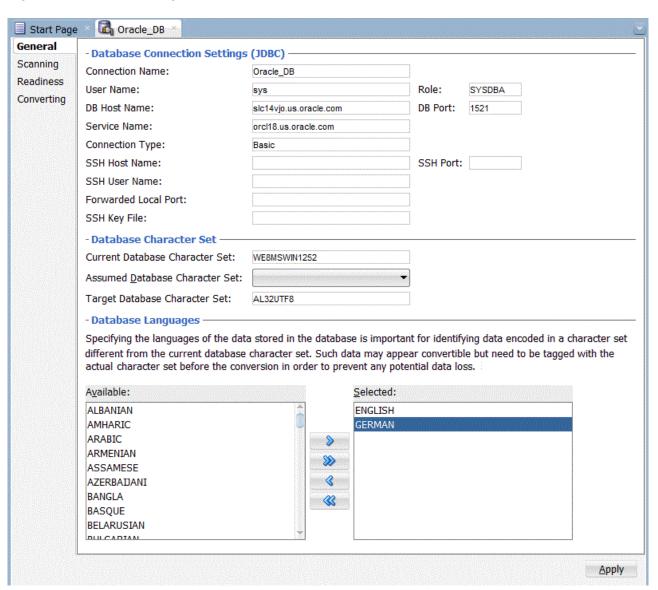
Before scanning the database, provide the DMU with the language and character set information that you collected when preparing for migration.

To set database properties when connecting to the database:

1. Connect to a database.

    The Database Properties tab is automatically opened in the client pane of the DMU window. The title of the tab is the name of the connection.

Figure 4-4   Database Properties Tab - General

To set database properties in the Navigator pane:

1. In the Navigator pane, right-click the database node and select **Properties** from the context menu.

2. From the left sidebar of the tab, choose the General subtab.

3. On the subtab, set the Assumed Database Character Set property to the identified real character set of data in the database.

   If your database is used in a correct character set configuration, as opposed to the pass-through configuration, this property should remain equal to the Current Database Character Set property.

> ✎ **See Also:**
>
> - "Review Your Preparations for Migration"
> - "Viewing and Setting Object Properties in the DMU" for more details about database and other properties

4.3.2 Scanning the Database with the Scan Wizard

Use the Scan Wizard to initiate the scan.

The scan results indicate whether data needs cleansing before it can be converted permanently without data loss. Also, the DMU uses the scan results to select the most effective conversion method for each table.

The following elements of the DMU user interface enable you to control the scanning process and view the scan results:

- Use the Scan Wizard to start the scanning process.

- Use the Scan Progress tab to monitor this process (see "Monitoring the Progress of a Scan").

- Use the Database Scan Report tab to view the scan results (see "Overview of the Database Scan Report")

The DMU might collect during scanning the rowid addresses of column cells that contain convertible values or values with expansion or invalid binary representation issues.

`CLOB` columns are not scanned or even listed in the Navigator pane in a multibyte database. Any multibyte database, including databases in AL32UTF8 and UTF8 character sets, store `CLOB` values in the same internal storage character set (Unicode UTF-16), and so `CLOB` values need no conversion when the database character set is migrated from one multibyte character set to another.

The source character set used for the test conversion of column values is the assumed character set of the column. See "Cleansing the Data".
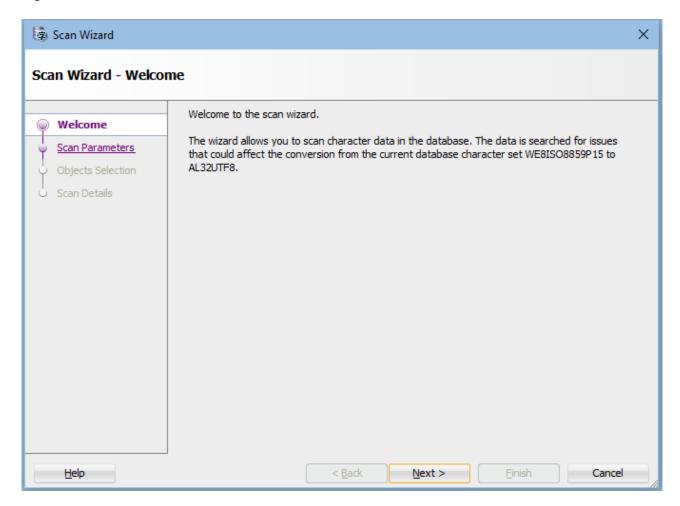
To scan the database using the Scan Wizard:

1. To open the Scan Wizard, select **Scan Database** from the Migration menu or **Scan Database**, **Scan Schema**, **Scan Table**, or **Scan Column** from the context menu of the corresponding object node in the Navigator pane.

The Scan Wizard is displayed, as in Figure 4-5.
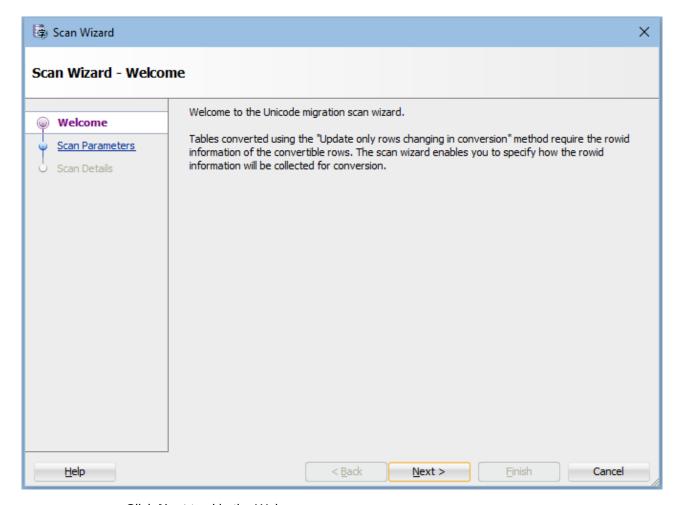
Figure 4-5   Scan Wizard - Welcome

The DMU might also open the Scan Wizard after you start the conversion process from the Conversion Details tab. The DMU opens the wizard if one or more tables in the database are to be converted using the conversion method "Update only convertible rows". The additional scan collects rowid addresses that this conversion method requires. The Scan Wizard opened during the conversion phase has a different first page, as shown in Figure 4-6. It also lacks the object selection page. See "Converting the Database" for more information about the conversion phase.
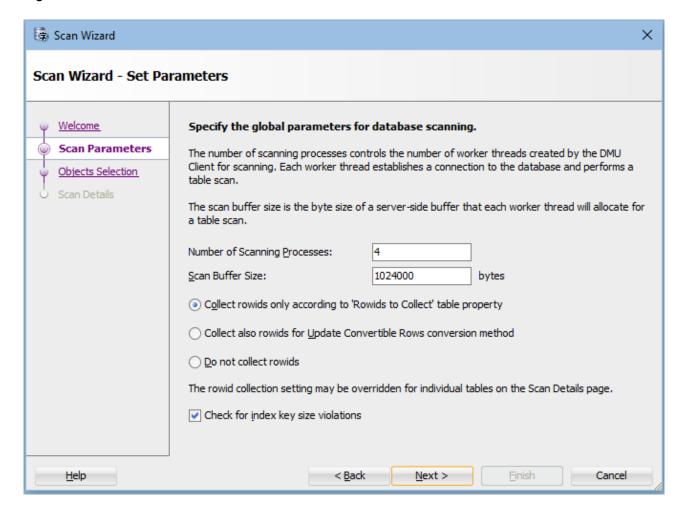
Figure 4-6   Scan Wizard - Welcome: Rowid Collection

2. Click **Next** to skip the Welcome page.

3. The second page of the wizard lets you set general parameters for the scanning process, as shown in Figure 4-7.

   The Number of Scanning Processes is the number of threads that the DMU spawns to perform the scanning process. Each thread opens a separate database connection. The DMU assigns a set of tables to each thread to scan independently. Larger tables might be split into chunks and scanned by multiple threads. Smaller tables are scanned entirely by one thread. The default value for this parameter is the value of the `CPU_COUNT` initialization parameter of the database. You can increase the parameter to see if the scanning time decreases, leveraging the parallel processing of CPUs and disks of the database server, or you can decrease the parameter to limit the negative impact that the scanning process might have on the performance of a busy production database.

Figure 4-7   Scan Wizard - Set Parameters

The parameter Scan Buffer Size controls the size in bytes of a buffer that the DMU allocates in each database server session to scan a table. The default value is 1000 kilobytes. The total buffer space used in a single scan is the number of scanning processes times the value of this property. Increasing the value of the property could speed up scanning, but only for as long as the allocated buffer memory fits into the available RAM on the database server.

The three radio buttons let you define the initial rowid collection level for tables selected for this scan. This initial level can be subsequently changed on the Scan Details page of the wizard, as described in point 5 below. The option "Collect rowids only according to 'Rowids to Collect' table property" tells the DMU to use the value of the table property 'Rowids to Collect' as the initial rowid collection level for each table. See "Table Properties: Scanning" for more details about this property.

The option "Collect also rowids for Update Convertible Rows conversion method" can be selected to let the DMU pre-collect rowid addresses for the "Update only convertible rows" conversion method. The option causes the initial rowid collection level to be set to "All to Convert" for all selected tables that have been assigned this conversion method and to the value of the property Rowids to Collect for all other selected tables. Usually, the rowids for this conversion method are collected by the already mentioned additional scan that starts automatically at the beginning of the conversion phase.

Pre-collection of rowids removes the need for this additional scan, shortening the required downtime window and enabling the database to be closed for business for a shorter time. However, if any additional convertible rows are added to a table after the last scan that pre-collected rowids for the table has been performed, these rows will not be converted during the conversion phase and will become incorrectly encoded data after the migration. Therefore, use the pre-collection feature sparingly, only on large tables confirmed to have static contents. The option "Collect also rowids for Update Convertible Rows conversion method" has no effect during the first scan of a table as no conversion method has been assigned yet to the table.
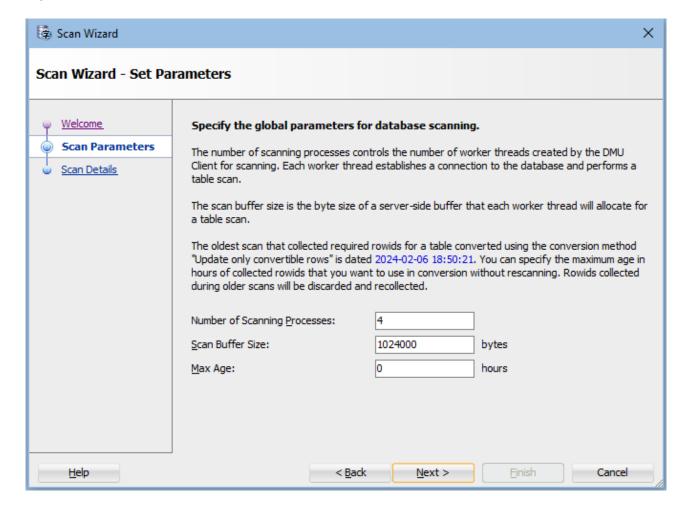
The last option, "Do not collect rowids", enables you to switch off rowid collection during the current scan. This may be useful if you just want to count the number of convertible cells and cells with convertibility issues, and you do not plan to use the filtering option of the Cleansing Editor on any of the scanned tables. The rowid collection cannot be switched off for a few special data dictionary tables in the SYS schema, which have their property "Rowids to Collect" fixed to "All to Convert".

The check box "Check for index key size violations" is automatically selected. This instructs the DMU to perform an index check and generate an index check report while scanning the database. The report lists all indexes whose key sizes will exceed the maximum supported size as a result of column modifications required for the conversion, or data conversion when the COMPATIBLE initialization parameter value is 20.0.0 or higher. This enables you to fix the problematic indexes before proceeding with the database conversion. See "Database Scan Report: Index Check Report" for more details. If you do not want to generate an index check report, you can deselect this check box.

In the Scan Wizard that the DMU opens in the conversion phase for collection of rowids, the Scan Parameters page changes as shown in Figure 4-8. The check box "Collect rowids for Update Convertible Rows conversion method" is automatically selected and the additional parameter Maximum Age is present on the page. This parameter enables you to skip rescanning of tables for which rowids have already been pre-collected in the recent Maximum Age hours. It also enables you to discard pre-collected rowids that you consider too old.

Figure 4-8   Scan Wizard - Set Parameters: Rowid Collection

See "Converting the Database" for more information about the conversion process.

Click **Next** to accept the scan parameters and open the Object Selection page.
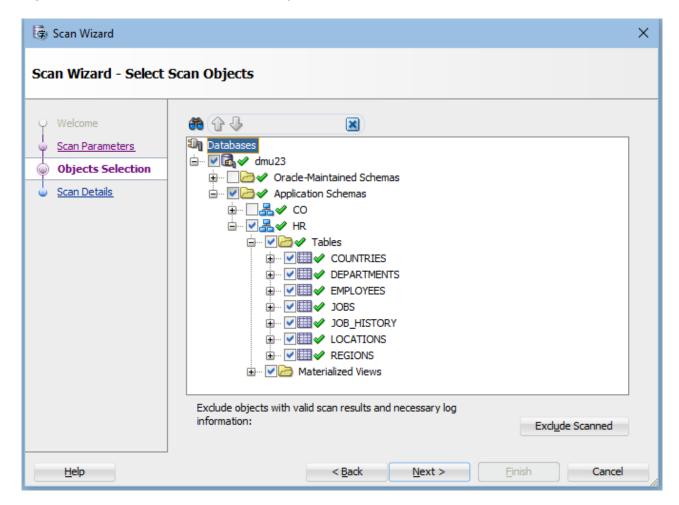
4. For the first scan, you usually select all tables in the database by opening the Scan Wizard from the Migration menu. For subsequent scans, you can select a subset of database objects for which you want the scan results to be refreshed, for example, because they have been invalidated by cleansing actions, or because you want recent DML activity on the objects to be analyzed in the context of possibly introduced conversion issues.

    If you want only invalidated scan results to be recalculated, you can select all columns in the database, by selecting the database node, and then click **Exclude Scanned**. When this option is selected, the DMU automatically deselects all columns that already have valid scan results. Now, you can manually select additional tables and columns for which you want existing scan results to be refreshed.

    If the "Collect rowids for Update Convertible Rows conversion method" option is selected, the Exclude Scanned option does not exclude tables that do not have required rowids collected by previous scans.

    Oracle strongly recommends that you refresh all scan results in the database before you start the conversion, preferably in the downtime window, when no new data can be added to the database. This ensures that all data requiring conversion is accounted for.

Figure 4-9   Scan Wizard - Select Scan Objects

Click **Next** to accept the object selection and open the Scan Details page. It might take a while to display the page, because, first, the wizard has to prepare a scanning plan for all the selected objects.
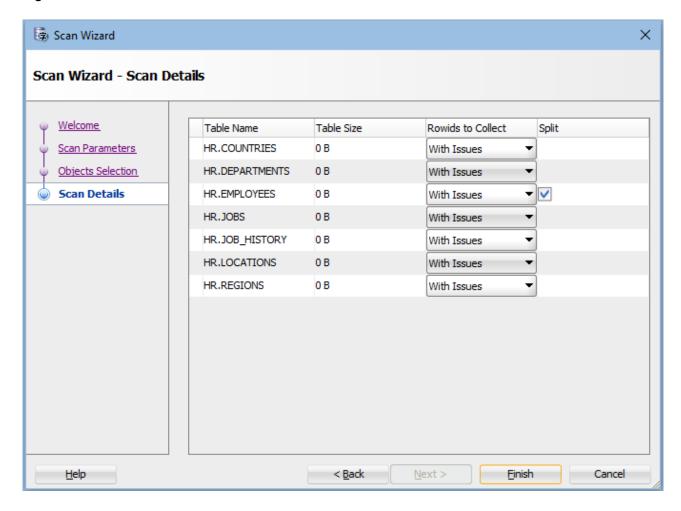
5. On the Scan Details page, shown in Figure 4-10, you can verify that all objects you wanted to be scanned are included. You can also set the rowid collection level in the drop-down lists that can be opened by selecting fields in the Rowids to Collect column. The choices for the rowid collection level are None (no collection is performed), All to Convert (collects rowids for all data that is not classified as needing no conversion), and With Issues (collects rowids for data with conversion issues only). Finding data that requires cleansing using the filtering and search features of the Cleansing Editor will be faster if rowids for problematic data cells are collected during scanning. See Using the DMU to Cleanse Data for more information about the Cleansing Editor.

The rowid collection level set in the Scan Wizard is valid only for the current scan. It does not modify the table property "Rowids to Collect" and it does not persist across invocations of the Scan Wizard.

The DMU automatically handles splitting of large tables into chunks for parallel scanning by multiple scanning processes. However, if necessary, you can prevent a table from being split by deselecting the corresponding check box in the Split column.

Clicking **Finish** will start the scan.

Figure 4-10   Scan Wizard - Scan Details
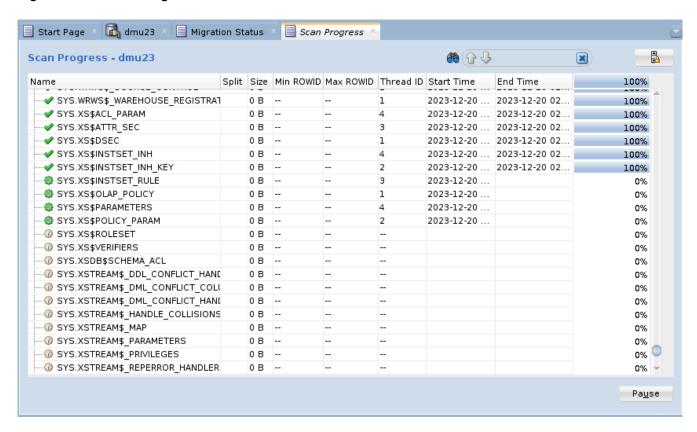
4.3.3 Monitoring the Progress of a Scan

You can use the Scan Progress tab to monitor the progress of the scanning task.

When the scanning starts, the DMU automatically opens the Scan Progress tab. The tab is shown in Figure 4-11.

Figure 4-11   Scan Progress Tab

The Scan Progress tab contains a grid that displays scan information for all tables in the scanned set. The columns of the tab grid are as follows:

- Name

  The column shows the name of the table that is described by the corresponding row of the grid. The icon to the left of the name shows the scan status. A green check mark shows that the table has already been scanned. A green sprocket-like icon marks tables that are currently being scanned. A clock marks tables that are waiting to be scanned.

  If a table is split into multiple chunks, you can click the plus sign to the left of the status icon to show information for all chunks to which the table has been split.

  > ✎ **Note:**
  >
  > The meaning of the green check mark icon differs between the Scan Progress tab and the Navigator pane/Scan Report tab (see "Overview of the Database Scan Report"). On the Scan Progress tab, the icon shows that a table has been successfully scanned, but without indicating if it contains any problematic data. In the Navigator pane or on a Scan Report tab, the icon means that a table, a schema or database has been scanned and also that its contents does not have any issues that might prevent successful conversion to Unicode.

- Split

  The number shown is the number of chunks to which the corresponding table has been split. The column is empty in grid rows showing chunk information.

- Size

  The column shows the size of the corresponding table or table chunk. The topmost row of the grid shows the added size of all tables in the scanned set.

- Min `ROWID`

  For chunks, the column shows the rowid of the first row in the chunk. It is empty for tables.

- Max `ROWID`

  For chunks, the column shows the rowid of the last row in the chunk. It is empty for tables.

- Thread ID

  The number shown in this column is the ID of a particular thread that is scanning or that has scanned the corresponding table or table chunk.

- Start Time

  The column shows the time when scanning of the corresponding table or table chunk started. If a table has been split, its row shows the earliest (minimum) start time for all its chunks. The topmost row of the grid shows the start time of the whole scanning task.

- End Time

  The column shows the time when the scanning of the corresponding table or table chunk finished. If a table has been split, its row shows the latest (maximum) end time for all its chunks. The topmost row of the grid shows the end time of the whole scanning task.

- Progress

  The column displays a progress bar and a percentage that indicate how much of the corresponding table or table chunk has already been scanned. The topmost row of the grid shows the overall progress of the scanning task.

  If a database error is encountered during the scanning of a table, the grid row corresponding to the table displays a prefix of the reported error message in this column. You can click the prefix to open a dialog box with the full message text.

The DMU scrolls down the tab grid automatically to bring new grid rows into view when the corresponding tables are assigned to scanning threads. You can suspend the automatic scrolling by clicking the **Scroll Lock** icon in the top right-hand corner of the Scan Progress tab. This enables you to monitor the overall progress of the scanning by looking at the progress indicators in the topmost row of the grid. To resume automatic scrolling, click the **Scroll Lock** icon again.

After the scan finishes, the DMU shows an information dialog box. You can now analyze the scan results on the Database Scan Report tab.

ORACLE

4.3.4 Stopping a Scan

Clicking **Pause** generates a warning dialog. Select **Abort** to stop the scan immediately or **Back** to resume scanning.

Figure 4-12   Stopping a Scan

4.3.5 Viewing the Database Scan Report

After the scan completes, view the most current scan results on the Scanning subtabs of the Properties tabs, or more conveniently on the Database Scan Report tab.

Depending on what the scan results are, you should follow one of the migration scenarios described in "Cleansing Scenario 1: A Database with No Issues", "Cleansing Scenario 2: Cleansing Expansion Issues", or "Cleansing Scenario 3: Cleansing Invalid Representation Issues".

To view the Database Scan Report:
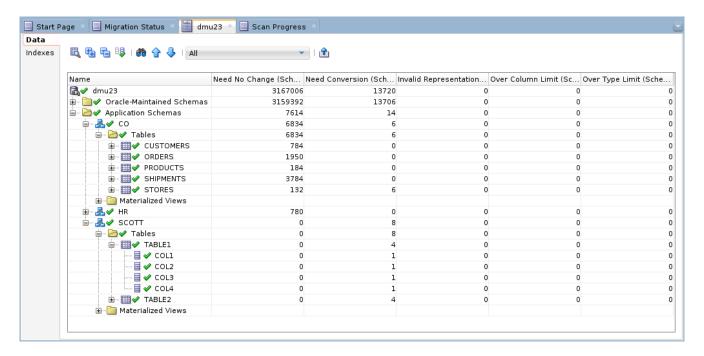
- Open the Database Scan Report tab with scan results for the whole database by selecting **Database Scan Report** from the Migration menu. A Database Scan Report tab for the whole database is shown in Figure 4-13.

- Alternatively, you can open a Database Scan Report tab for tables in a single schema, for a single table, or for a single column, by selecting **Scan Report** from the context menu of the corresponding schema, table, or column node in the Navigator pane.

4.3.6 Overview of the Database Scan Report

The Database Scan Report tab contains a toolbar on the top and a result grid underneath. You can use three methods to locate results of interest in the Report, and also export the Report.

Figure 4-13   Database Scan Report

4.3.6.1 Database Scan Report: Result Grid

The result grid is a user interface item, in form of a tree table, that displays scan results and, optionally, other properties of database objects.

The first column of the grid contains a database object tree that has the same appearance as the tree on the Navigator pane (see "Introduction to the DMU User Interface"). The root of the tree is the database, schema, table, or column for which the Database Scan Report has been opened. Each node of the tree is associated with a row of the grid. This row shows various properties of the database object corresponding to the node.

The status icon to the left of the name of the node shows the convertibility status for the database object described by the node. The meaning of icons is defined in Table 2-1.

A node can be expanded and collapsed to show or hide results of its child nodes by clicking the plus icon to the left of the node. You can expand all descendants of a node at once by selecting the row containing the node and clicking **Expand All**, the second icon on the toolbar (see Table 2-1). You can collapse all descendants by selecting the node and clicking **Collapse All**, the third icon on the toolbar.

When the Database Scan Report tab is opened, the columns displayed in the grid by default are: Need No Change (Scheduled), Need Conversion (Scheduled), Invalid Binary Representation (Scheduled), Over Column Limit (Scheduled), and Over Type Limit (Scheduled). The values shown in these columns are equal to scan results shown under the heading "Including Effects of Scheduled Cleansing" on the Scanning subtab of the Properties tab for the corresponding node. Those scan results are described in Viewing and Setting Object Properties in the DMU.

By clicking on **Customize Report**, the fourth icon on the toolbar, you can open the Customize Scan Report dialog box. In this dialog box, you can select further property columns to display along the default ones. You can also hide the default columns. The available columns correspond to properties described in Viewing and Setting Object Properties in the DMU. The columns Need No Change, Need Conversion, Invalid Binary Representation, Over Column

Limit, and Over Type Limit, without the suffix "(Scheduled)", correspond to scan results presented on Scanning subtabs under the heading "Current Data".

If a property displayed in the Database Scan Report grid column is valid only for certain types of nodes, the grid column displays no text in rows corresponding to other types of nodes. For example, the Conversion Method property is valid only for tables. Grid rows for the database, schemas, and columns do not show anything in the Conversion Method column, if it is added to the Database Scan Report tab.

If you right-click a row in the Database Scan Report grid, you can select **Scan**, **Cleansing Editor**, or **Properties** from the context menu. The Scan menu item opens the Scan Wizard in which only the object described by the current grid row is selected for scanning. This way you can quickly instruct the DMU to refresh scan results for this object. Cleansing Editor opens the Cleansing Editor tab for the selected table or table column. See Using the DMU to Cleanse Data for the description of the Cleansing Editor tab. Clicking **Properties** opens the Properties tab of the object described by the selected grid row (see Viewing and Setting Object Properties in the DMU).

The context menu for tables under the Oracle-Maintained Schemas group node contains the Data Viewer instead of the Cleansing Editor. The Data Viewer tab, opened by clicking **Data Viewer**, is a read-only version of the Cleansing Editor.

4.3.6.2 Database Scan Report: Navigating by Status Icons

If you expand Database Scan Report nodes marked with a yellow triangle, you can locate columns that contain data with convertibility issues.

The triangle icon signals that columns among node descendants have convertibility issues.

1. Expand a node marked with a triangle icon.

2. Search its children to locate child nodes marked with the same icon.

3. Repeat the preceding steps until you reach the column nodes.

Column nodes marked with a yellow triangle describe columns that must be cleansed. Nonzero counts in the grid columns Invalid Binary Representation (Scheduled), Over Column Limit (Scheduled), and Over Type Limit (Scheduled) show how many issues of each type have been found in the given database column. Counters are displayed as links that you can click to directly open the Cleansing Editor or the Data Viewer tab to take a closer look at the issues.
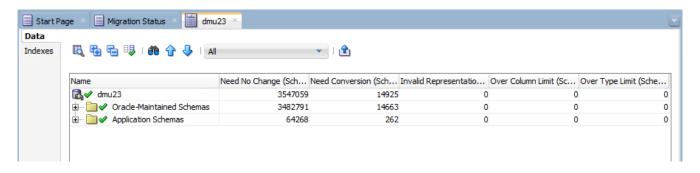
4.3.6.3 Database Scan Report: Filtering

The filtering feature of the Database Scan Report tab performs useful actions such as displaying only objects with valid scan results and displaying objects without valid scan results.

You can open the drop-down filter list on the toolbar of the tab, as shown in Figure 4-14, and select the type of results to include in the report.

Figure 4-14   Database Scan Report: Filtering

The available filters are:

- All

  Switches the filtering off. All objects are displayed.

- Scanned

  Displays only objects with valid scan results.

- Scan Failed

  Displays objects that could not be scanned, because a database error was reported.

- Not Scanned

  Displays objects without valid scan results. An object might have no valid scan results because it has never been scanned, its scan results have been invalidated by a cleansing action or a DDL, or the last scan of the object failed.

- Requiring No Conversion

  Displays objects that have valid scan results and contain only changeless data that requires no conversion.

- Requiring Conversion, Without Issues

  Displays objects that have valid scan results and contain a mixture of changeless data that requires no conversion and convertible data that requires conversion. No issues are expected during conversion of these objects.

- Blocking Conversion

  Displays objects with unresolved convertibility issues that must be acted upon before the database can be successfully converted.

- With Invalid Representation

  Displays objects that have valid scan results and contain some data with invalid binary representation, that is, with character codes that are not valid in the declared column character set.

- Exceed Dataytpe Limit

  Displays objects that have valid scan results and contain some data that exceeds its data type limit after conversion.

- Exceed Column Limit

  Displays objects that have valid scan results and contain some data that exceeds its column limit after conversion.

- With Length Issues

  Displays objects that have valid scan results and contain some data that exceeds its column limit or its data type limit after conversion.

- Not Changeless

  Displays objects that have valid scan results and contain some data that will require conversion. The data may or may not have convertibility issues. That is, it may belong to any of the categories: "Need Conversion", "Invalid Representation", "Over Column Limit", or "Over Type Limit".

- With Some Issues

  Displays objects that have valid scan results and contain data with any of the possible problems for which the DMU searches. For data dictionary schemas, this includes convertible data in columns that the DMU cannot convert.

After a filter is selected, all nodes and associated grid rows for database columns whose scan results do not fulfill the filtering condition are hidden. If all children of a node are hidden, the node is hidden as well. The exception is the root node of the report, which is never hidden.

Filtering is useless and thus disabled in Database Scan Report tabs opened for a single database column.

You can also rescan only the filtered objects in the scan report by selecting the first icon of the toolbar, which is the Rescan Filtered Objects button. As an example, if the the "Exceed Column Limit" filter is set in the scan report, then the Rescan Filtered Objects button will select all tables with "Over Column Limit" issues for the rescan.

4.3.6.4 Database Scan Report: Searching

The search feature in the Database Scan Report tab makes analyzing scan results easier in large scan reports.

Initiate a search by clicking one of the following toolbar buttons:

- Find

  The fifth button on the toolbar opens a dialog box that enables you specify the search criteria for objects.

- Find Previous
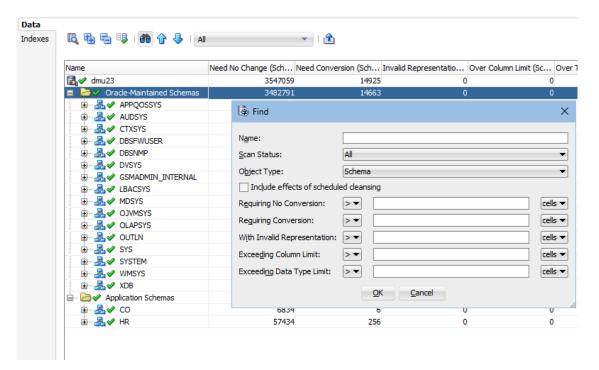
  The sixth button on the toolbar highlights the previous instance of the originally searched item.

- Find Next

  The seventh button on the toolbar highlights the next instance of the originally searched item.

Selecting **Find** displays a dialog box such as that shown in Figure 4-15.

Figure 4-15   Database Scan Report: Searching

You can specify the following search criteria in the dialog box to describe objects that you want located:

- Name

  Specify a string that should be contained in the name of the object (case-insensitive).

- Scan Status

  Specify the scan status of the object.

- Object Type

  Specify if you want to search among schemas, tables, or columns. Only one type can be selected.

- Include effects of scheduled cleansing

  Select this option if the results for which to search should include effects of the scheduled cleansing action. If the option is selected, the compared results are those displayed in grid columns with the phrase "(Scheduled)" in heading.

- Requiring No Conversion

  Select the comparison operator and the value to compare with the Need No Change results.

- Requiring Conversion

  Select the comparison operator and the value to compare with the Need Conversion results.

- With Invalid Representation

  Select the comparison operator and the value to compare with the Invalid Binary Representation results.

- Exceeding Column Limit

Select the comparison operator and the value to compare with the Over Column Limit results.

- Exceeding Data Type Limit

  Select the comparison operator and the value to compare with the Over Type Limit results.
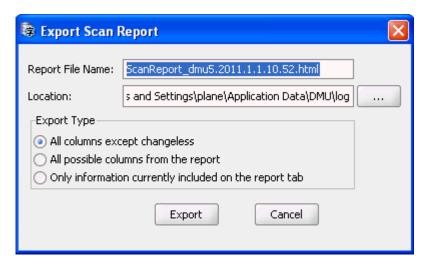
After you select **OK**, the grid row for the first schema, table, or column, as selected in Object Type, that fulfills all the specified criteria is highlighted in the scan report. You can select the **Find Previous** or the **Find Next** toolbar button to highlight the previous or the next row of the same type that fulfills the criteria.

4.3.6.5 Database Scan Report: Exporting to HTML

To save the scan results for future reference, export them to an HTML file.

Select **Export as HTML**, the last button on the toolbar. This will open the Export Scan Report dialog box, as shown in Figure 4-16.

Figure 4-16   Export Scan Report

Specify the name and directory (folder) for the HTML file. By choosing one of the options in the Export Type group, specify if you want to export, correspondingly, any one of the following:

- Only grid rows for columns and their parent objects that require conversion or have data with convertibility issues
- The entire content of the scan report
- Only those rows and columns of the result grid that are visible on the report tab (not hidden or collapsed)

Click **Export** to create the HTML report.

4.3.6.6 Database Scan Report: Index Check Report

The index check report lists all indexes whose key sizes will exceed the maximum supported size as a result of column modifications required for the conversion.

The index check report is generated during a database scan when the "Check for index key size violations" check box is selected in the Scan Wizard. The report appears on the Indexes subtab of the Database Scan Report.

You can use the index check report while preparing for database conversion. For example, after you have scanned, cleansed, and rescanned the data until there are no convertibility issues, you can examine the index check report to view any indexes whose key sizes will exceed the maximum supported size as a result of data cleansing or increased post-converted data size. If the report displays a problematic index:

1. Address the problem with the index using one of the methods described in "Maximum Index Key Size".

2. Rescan the table or schema for the index by right-clicking the table or schema node in the index check report grid and selecting **Scan**. Alternatively, you can rescan the entire database. In the Scan Wizard, be sure that the "Check for index key size violations" check box is selected.

3. View the index check report. If the report displays a green OK icon to the left of the database node, then all index key size violations have been resolved and you can continue with the database conversion. If violations still exist, repeat the preceding steps.

The index check report contains a toolbar on the top and a result grid underneath. The result grid is a user interface item, in the form of a tree table. The toolbar allows you to expand or collapse nodes in the tree table, select columns to display in the grid, and export the grid to an HTML file.
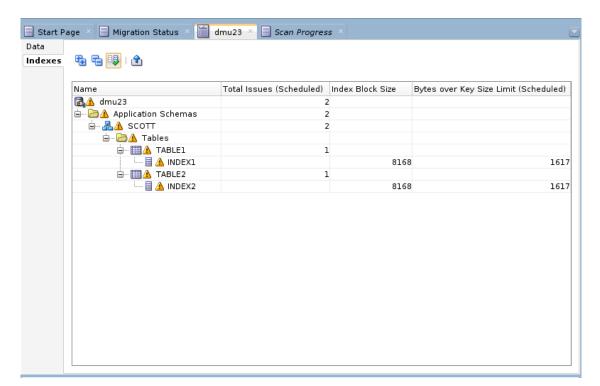
Figure 4-17   Index Check Report

The first column of the grid contains a database object tree that has the same appearance as the tree on the Navigator pane (see "Introduction to the DMU User Interface"). The root of the tree is the database, schema, or table for which the Database Scan Report has been opened. Each node of the tree is associated with a row of the grid. This row shows various properties of the database object corresponding to the node.

The object tree displays only indexes that will cause errors during the database conversion. Therefore, the status icon to the left of each node in the tree is a yellow warning triangle. A

yellow triangle to the left of a database or schema node indicates that the database or schema contains a table on which a problematic index is defined. A yellow triangle to the left of a table node indicates that a problematic index is defined on the table. A yellow triangle next to an index node indicates that the index is problematic.

A node can be expanded or collapsed to show or hide results of its child nodes by clicking the plus or minus icon to the left of the node. You can expand all descendants of a node at once by selecting the row containing the node and clicking **Expand All**, the first icon on the toolbar (see Table 2-1). You can collapse all descendants by selecting the node and clicking **Collapse All**, the second icon on the toolbar.

By clicking on **Customize Report**, the third icon on the toolbar, you can open the Customize Index Check Report dialog box. In this dialog box, you can select the columns to display in the grid. The available columns are:

- Total Issues - Total number of index key size issues

- Total Issues (Scheduled) - Total number of index key size issues, including issues caused by the the scheduled lengthening of columns. This column is displayed in the grid by default.

- Index Block Size - Block size of the tablespace that contains the index (in bytes). This column is displayed in the grid by default.

- Bytes over Key Size Limit - Estimated number of bytes by which the index key will exceed the size limit due to the migration of character length semantics columns

- Bytes over Key Size Limit (Scheduled) - Estimated number of bytes by which the index key will exceed the size limit due to the effects of scheduled lengthening of columns. This column is displayed in the grid by default.

- Estimation Method - Method used for estimating the post-conversion index key size. Possible values:

  – Column length - The estimate is based on column length

  – Post-conversion data length - The estimate is based on the maximum post-conversion data length of the column
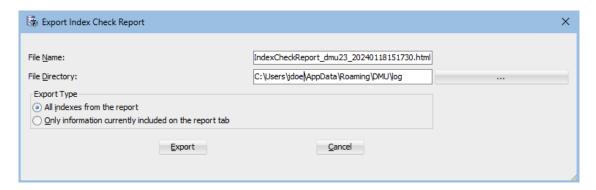
If a property displayed in a grid column is valid only for certain types of nodes, the grid column displays no text in rows corresponding to other types of nodes. For example, the Estimation Method property is valid only for indexes. Grid rows for the database, schemas, and tables do not show anything in the Estimation Method column.

If you right-click a schema or table row in the grid, you can select **Scan** or **Properties** from the context menu. The **Scan** menu item opens the Scan Wizard, in which only the object described by the current grid row is selected for scanning. This allows you to quickly instruct the DMU to refresh scan results for the object. Clicking **Properties** opens the Properties tab of the object described by the selected grid row (see Viewing and Setting Object Properties in the DMU).

You can export the grid by selecting **Export as HTML**, the last button on the toolbar. This opens the Export Index Check Report dialog box, as shown in Figure 4-18.

Figure 4-18   Export Index Check Report

Specify the name and directory (folder) for the HTML file.

By choosing one of the options in the Export Type group, specify if you want to export, correspondingly, either of the following:

• All rows of the result grid, regardless of whether they are expanded or collapsed

• Only those rows of the result grid that are visible (expanded) on the report tab

Both options export only the columns that are visible on the report tab.

Click **Export** to create the HTML report.

4.3.7 Generating the Problem Data Report

A Problem Data report shows details of objects that contain data with convertibility issues. DMU can generate a Problem Data Report as an `.xls` spreadsheet.

To generate the Problem Data report:

1. From the **Migration** menu, select **Export Problem Data**.

   The Welcome page of Export Problem Data Wizard is displayed.

2. Click **Next**.

   The Select Objects page is displayed showing all the database objects that have convertibility issues.

3. Select the database objects whose details you want to see in the Problem Data report.

4. Click **Next**. The Generate Report page is displayed.

5. Enter the values for the following fields on the Generate Report page:

   • **Preview Data Length**: Number of bytes of data that you want to see in the Problem Data report for the data cells having convertibility issues.

   • **Report File Name**: Name of the Problem Data report file. The file has the `.xls` extension.

   • **Report File Location**: Directory in which to store the Problem Data report file.
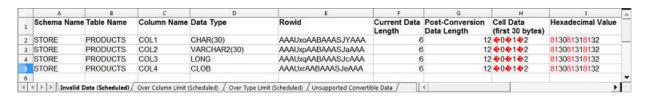
> **Note:**
>
> The default Problem Data report file name is in the format
> `ProblemDataReport_ProjectName_yyyyMMddHHmmss.xls`. The default directory
> path is the value specified for the **Log Directory** field in the **Preference** panel.
> You can either use the default values provided by DMU or specify any other
> values for Problem Data report file name and its directory path.

6. Click **Finish** to generate the Problem Data report.

The following figure shows a sample Problem Data report.

Figure 4-19   Sample Problem Data Report

> **Note:**
>
> • For the invalid representation and over size limit data, the problem characters are
>   highlighted in the Cell Data and Hexadecimal Value columns in the Problem Data
>   report.
>
> • DMU does not support converting `ANYDATA` or `ANYDATASET` to Unicode format and
>   considers it as *problem* data. This type of data is shown in the **Unsupported
>   Convertible Data** tab.

4.3.8 Scanning the PL/SQL Code

After scanning the database using the Scan Wizard, you can use the PL/SQL Code Scan
Wizard to scan the PL/SQL code in your application.

4.3.8.1 About PL/SQL Code Scans

The PL/SQL Code Scan Wizard can identify several conversion-related issues.

Specifically, scanning can identify any of the following types of Unicode conversion related
issues or *coding patterns*:

• Variable length needs expansion due to planned column length expansion

• Variable length shorter than the column length

• Variable length shorter than the column data

• Variable length semantics different from the column length semantics

• SQL function operating on character data not in the database character set

- Non-ASCII identifier name exceeding maximum length

After the PL/SQL code has been scanned, you can view the scan summary by generating the following two reports:

- PL/SQL Code Scan Report
- PL/SQL Problem Code Report

> **Note:**
>
> The PL/SQL Code Scan Wizard can be used only in the Migration mode before doing database conversion. The PL/SQL Code Scan Wizard is disabled in the Validation mode.

4.3.8.2 Prerequisites for Scanning the PL/SQL Code

Before scanning application code using the PL/SQL Code Scan Wizard, you must meet several prerequisites, including the database version, repository availability, and so on.

The prerequisites are as follows:

- Oracle database version is 12.2 or later.
- The DMU repository is installed in the database in the Migration mode.
- The DMU is connected to the database.
- The PL/Scope identifier data for the following PL/SQL objects is already collected, in the following order:

> **Note:**
>
> PL/Scope is a compiler-driven tool that collects data about identifiers in a PL/SQL source code at program-unit compilation time and makes it available in static data dictionary views. The collected data includes information about identifier types, usages (declaration, definition, reference, call, assignment), and the location of each usage in the source code. See *Oracle Database Development Guide* for more information about PL/Scope.

  – `STANDARD` and `DBMS_STANDARD` packages

    The `STANDARD` and `DBMS_STANDARD` packages declare and define Oracle Database base types, such as `VARCHAR2` and `NUMBER`. The PL/Scope identifier data for these packages is always available in Oracle Database 12.2 and later. So, you do not have to explicitly collect this data.

  – Oracle supplied packages containing Unicode migration unsafe procedures and functions

    The following Oracle supplied packages contain Unicode migration unsafe procedures and functions – `UTL_RAW`, `UTL_I18N`, and `UTL_FILE`.

    The DMU automatically collects this data when you use PL/SQL Code Scan. So, you do not have to explicitly collect this data.

– Your application's PL/SQL code

You must explicitly collect the PL/Scope identifier data for your application's PL/SQL code. To achieve this, set the PL/SQL compilation parameter `PLSCOPE_SETTINGS` to `IDENTIFIERS:ALL` by issuing the following statement:

```
SQL> ALTER SYSTEM SET PLSCOPE_SETTINGS='IDENTIFIERS:ALL';
```

Then recompile your PL/SQL code by using either of the following methods:

* Issue the `ALTER PACKAGE ... COMPILE` statement. See *Oracle Database PL/SQL Language Reference* for more information.

* Run the `DBMS_DDL.ALTER_COMPILE()` procedure. See *Oracle Database PL/SQL Packages and Types Reference* for more information.

> ✎ **Note:**
>
> Do not use the `CREATE OR REPLACE PACKAGE` statement to recompile your PL/SQL code. This method is not supported for this use case.

4.3.8.3 Scanning the PL/SQL Code with the PL/SQL Code Scan Wizard

You can scan the PL/SQL code in your application using the PL/SQL Code Scan Wizard.

To scan the PL/SQL code using the PL/SQL Code Scan Wizard:
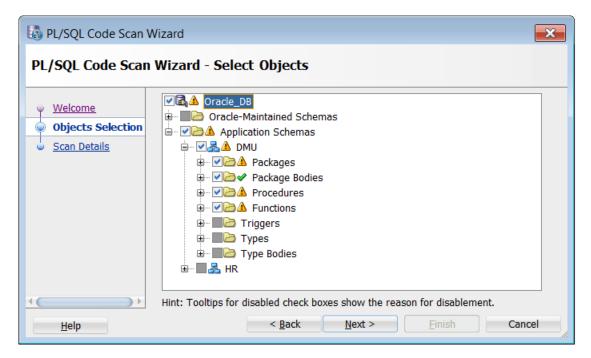
1. In the Tools menu, click **Scan PL/SQL Code...**. The Welcome page is displayed.

Figure 4-20   PL/SQL Code Scan Wizard - Welcome Page
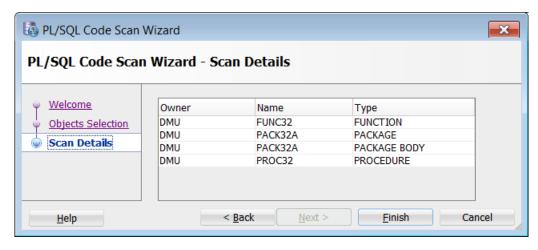
2. Click **Next**. The Select Objects page is displayed.

Figure 4-21   PL/SQL Code Scan Wizard - Select Objects Page
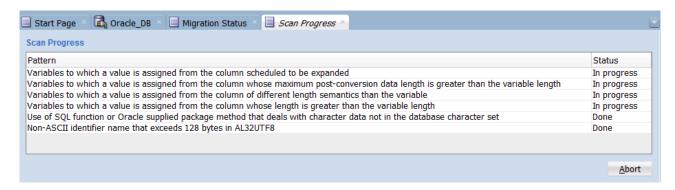
3. On the Select Objects page, select the PL/SQL objects that you want to scan.

4. Click **Next**. The Scan Details page is displayed, showing the list of PL/SQL objects that will be scanned.

Figure 4-22   PL/SQL Code Scan Wizard - Scan Details Page

5. Click **Finish**. The Scan Progress panel is displayed, showing the progress of the PL/SQL code scan operation.
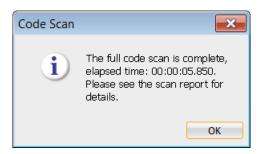
Figure 4-23   PL/SQL Code Scan Progress Panel

If you want to cancel the scan operation, click **Abort**.

6. After the PL/SQL code scan is complete, the Code Scan completion dialog is displayed.

Figure 4-24   Code Scan Completion Dialog
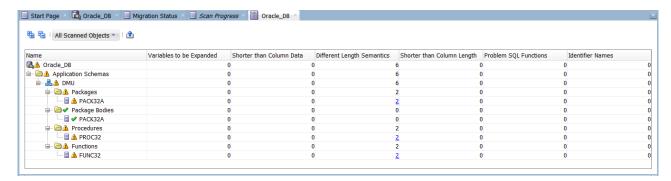
4.3.8.4 Viewing the PL/SQL Code Scan Report

After the PL/SQL code scan has completed, you can view the summary of the scan using the PL/SQL code scan report.

To view the PL/SQL code scan report:

1. In the Tools menu, click **PL/SQL Code Scan Report…**. The PL/SQL code scan report is displayed.
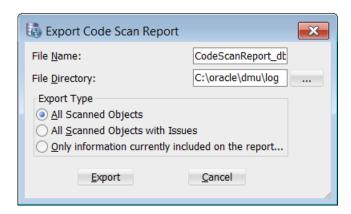
Figure 4-25   PL/SQL Code Scan Report

The PL/SQL code scan report shows the number of errors in PL/SQL code in a specific schema, package, procedure, and function.

You can also export this report to an HTML file by clicking the export icon ![icon] present in the top section of the report. When you click the export icon, the Export Code Scan Report dialog is displayed as shown below.

Figure 4-26   Export Code Scan Report
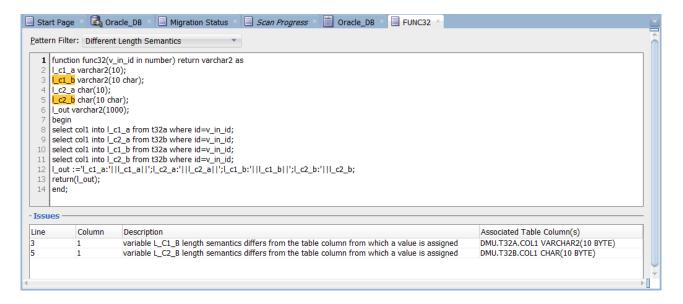
Here, you can specify the file name and directory path to store the report. You can also specify the type of information that you would like to to include in the exported file, that is, to include all the scanned objects, or all the scanned objects having issues, or only the information that is currently shown in the report.

Click **Export** to export the report to an HTML file.

2. In the PL/SQL code scan report, when you click the error count number link for a specific PL/SQL object, the Code Viewer panel for the PL/SQL object is displayed.
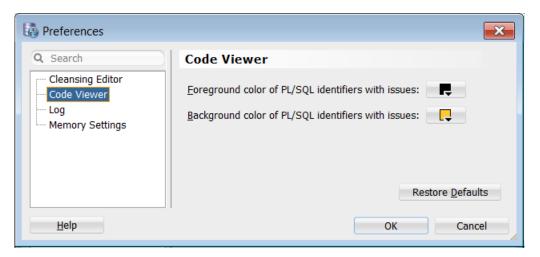
Figure 4-27   Code Viewer Panel

The Code Viewer panel highlights the code having issues in the PL/SQL object, and displays the descriptions of the issues below the code. By default, all types of issues are highlighted in the PL/SQL code.
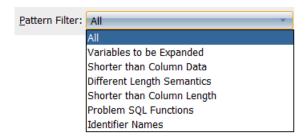
The foreground and background colors used for highlighting the PL/SQL code having issues can be configured using the Code Viewer page of the Preferences dialog box, as shown in the following figure. You can open the Preferences dialog box by selecting **Preferences** from the Tools menu.

Figure 4-28   Code Viewer Preferences Page

3. If you want a specific issue type to be highlighted in the PL/SQL code, select the required issue type from the Pattern Filter list that is present in the top section of the Code Viewer panel.

Figure 4-29   Pattern Filter List

4.3.8.5 Viewing the PL/SQL Problem Code Report

After the code scan completes, use the Export PL/SQL Problem Code Wizard to generate an Excel report describing problems in the code.

To view the PL/SQL problem code report:

1. In the Tools menu, click **PL/SQL Problem Code Report…**.

    The Welcome page appears.
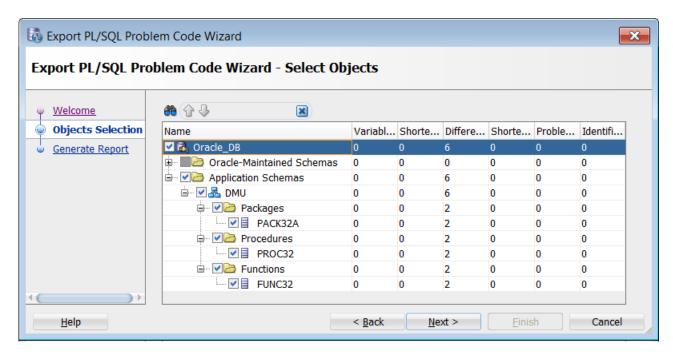
Figure 4-30   Export PL/SQL Problem Code Wizard - Welcome Page

2. Click **Next**.

   The Select Objects page is displayed, showing the PL/SQL objects having issues.
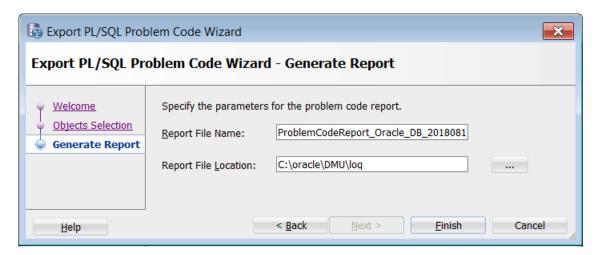
Figure 4-31   Export PL/SQL Problem Code Wizard - Select Objects Page

3. On the Select Objects page, select the PL/SQL objects that you want to include in the problem report.

4. Click **Next**.

   The Generate Report page is displayed.

Figure 4-32   Export PL/SQL Problem Code Wizard - Generate Report Page

5. On the Generate Report page, you can specify file name and directory path for storing the report as an Excel file.

   The default PL/SQL problem code report file name is in the format `ProblemCodeReport_ProjectName_yyyyMMddHHmmss.xls`, where *ProjectName* is the database connection name that you have set in the DMU GUI, and *yyyyMMddHHmmss* is the current timestamp generated by combining the values of year, month, day, hour, minutes, and seconds. For example, `ProblemCodeReport_Oracle_DB_20180818150947.xls`. Here, `Oracle_DB` is the project name and `20180818150947` is the timestamp.
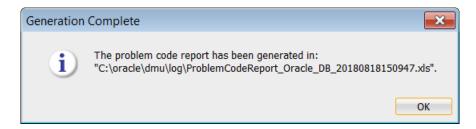
   The default directory for storing the report is the value specified for the Log Directory field on the Log page of the Preferences dialog box. The Preferences dialog box can be opened by selecting Preferences from the Tools menu.

   You can either use the default values or specify any other values for the PL/SQL problem code report file name and its directory path.

6. Click **Finish**.
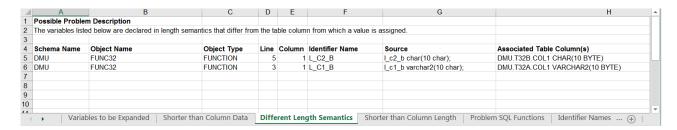
   A progress bar showing the progress of generating the PL/SQL problem code report is displayed. After the report has been generated successfully, a dialog box showing its name along with its directory path is displayed.

Figure 4-33   Report Generation Complete Dialog

A PL/SQL problem code report file is an Excel file containing multiple worksheets, where each worksheet contains PL/SQL code issues related to a specific type of issue or *coding pattern* as shown in the following sample report.

Figure 4-34   Sample PL/SQL Problem Code Report Excel File

	A	B	C	D	E	F	G	H
1	Possible Problem Description							
2	The variables listed below are declared in length semantics that differ from the table column from which a value is assigned.							
3								
4	Schema Name	Object Name	Object Type	Line	Column	Identifier Name	Source	Associated Table Column(s)
5	DMU	FUNC32	FUNCTION	5	1	L_C2_B	l_c2_b char(10 char);	DMU.T32B.COL1 CHAR(10 BYTE)
6	DMU	FUNC32	FUNCTION	3	1	L_C1_B	l_c1_b varchar2(10 char);	DMU.T32A.COL1 VARCHAR2(10 BYTE)
7								
8								
9								
10								

Variables to be Expanded | Shorter than Column Data | **Different Length Semantics** | Shorter than Column Length | Problem SQL Functions | Identifier Names ...

> **Note:**
>
> A problem code report file has a limit of 65536 rows. If a problem code report file crosses 65536 rows, then one more problem code report file is created with the additional rows added to it. When more than one problem code report files are created, incremental integer numbers are suffixed to their file names.
>
> For example:
>
> ```
> ProblemCodeReport_ProjectName_yyyyMMddHHmmss_1.xls
> ProblemCodeReport_ProjectName_yyyyMMddHHmmss_2.xls
> ...
> ProblemCodeReport_ProjectName_yyyyMMddHHmmss_n.xls
> ```

4.3.9 Scanning the Database with the DMU Command-Line Scanner (DMU-CLS)

The DMU Command-line Scanner (DMU-CLS) is a command-line tool for scanning the database. It provides the same functionality as the GUI scan.

The DMU-CLS scans a database to check its character data and provides information about the data readiness for its migration to a Unicode database or about any problem data that is currently present in the existing Unicode database. After scanning a database, the DMU-CLS can provide its analysis results in the form of a scan report and a problem data report.

> **See Also:**
>
> "Scanning the Database with the Scan Wizard"

4.3.9.1 Prerequisites for Using the DMU-CLS

Before you scan an Oracle database using DMU-CLS, you must meet prerequisites such as the database version, installation of required packages, and so on.

The following are the prerequisites:

- The Oracle database version must be 11.2.0.4 or later.

- The PL/SQL package `SYS.DBMS_DUMA_INTERNAL` must be installed in the Oracle database.

- All the other DMU connections must be closed.

- The database user using the DMU-CLS must have the `SYSDBA` role.

> ✎ **See Also:**
>
> "Managing a DMU Repository Using the DMU-CLS" to learn more about installing the DMU repository

4.3.9.2 Starting the DMU-CLS

Start the DMU-CLS using the `dmucls` command, specifying parameters on the command line or using a command file.

The command syntax depends on your platform::

- On a Windows system:

```
c:\> dmucls.bat parameters
```

- On a Linux or UNIX system:

```
$ sh dmucls.sh parameters
```

In the preceding examples, *parameters* are optional command-line parameters. You can also store the `dmucls` command parameters in a text file called the *parameter file*. You can then specify the parameter file name as the only parameter to the `dmucls` command instead of specifying all the individual parameters on a command-line.

The examples in this document are provided for Linux and UNIX systems.

> ✎ **Note:**
>
> - You can see the list of all the `dmucls` command parameters and their descriptions using the following command:
>
>   ```
>   $ sh dmucls.sh -h
>   ```
>
> - The `dmucls` command parameters and their values must be specified as key-value pairs on the command line. For example:
>
>   ```
>   $ sh dmucls.sh param1=value1 param2=value2
>   ```
>
> - For the parameters having multiple values, the values must be separated by a comma (,). For example:
>
>   ```
>   $ sh dmucls.sh param1=value1,value2
>   ```
>
>   Any additional spaces between two comma-separated values are ignored.
>
> - The `dmucls` command parameters and their values are case-insensitive, except the values for the parameters `CONNECTION`, `PARFILE`, `REPORT_NAME`, and `REPORT_DIR`.

The following examples show some sample `dmucls` commands:

Example 1:

```
$ sh dmucls.sh CONNECTION=db_connection_1 OPERATION=scan SCHEMAS=hr,oe
REPORT_NAME=scan_report
```

Here, the DMU-CLS connects to the database having the database connection name `db_connection_1` that is configured using the DMU GUI and scans the schemas `hr` and `oe`, and stores the scan result in a report file having the name `scan_report`.

Example 2:

```
$ sh dmucls.sh CONNECTION=db_connection_1 OPERATION=INSTALL_MIGRATION_REPO
REPO_TABLESPACE=sysaux REPO_TARGET_CHARSET=UTF8
```

Here, the DMU-CLS connects to the database having the connection name `db_connection_name` and installs the DMU repository having the target database character set of `UTF8` in the tablespace `sysaux`.

> **✎ Note:**
>
> If the value for `REPO_TARGET_CHARSET` is not specified, then the default value of `AL32UTF8` is used as the target database character set for the DMU repository. Oracle recommends using `AL32UTF8` as the target database character set.

> **✎ See Also:**
>
> - "DMU-CLS Parameters" for the list of all the `dmucls` command parameters and their descriptions.
> - "DMU-CLS Parameter File" for more information about the DMU-CLS parameter file.

4.3.9.3 DMU-CLS Parameters

You can specify several parameters using `dmucls`, such as `CONNECTION`, `OPERATION`, and `SCHEMAS`.

The following table lists all the `dmucls` command parameters and their descriptions.

> **✎ Note:**
>
> The `dmucls` command parameters and their values are case-insensitive, except the values for the parameters `CONNECTION`, `PARFILE`, `REPORT_NAME`, and `REPORT_DIR`.

Table 4-1   List of `dmucls` Command Parameters

Parameter	Description
CONNECTION	Oracle database connection name already configured in the DMU using the DMU GUI. This is one of the three ways in which a database connection information can be specified to the `dmucls` command. The other two ways to specify a database connection information to the `dmucls` command are the *Easy Connect* format and the *Connect Descriptor* format, which can be specified only using the CONNECT_TO parameter.

Example:

```
CONNECTION=db_connection_1
```

In the preceding example, `db_connection_1` is the Oracle database connection name that is already configured in the DMU using the DMU GUI.

Note:

- If the `Basic` type of database connection is configured, the DMU-CLS will prompt you to enter the database administrator password, if it is not already stored in the DMU.
- If the `SSH` type of database connection is configured and if key file is not used, then the DMU-CLS will prompt you to enter password for the SSH server user. If key file is used and a passphrase is set for the key, then the DMU-CLS will prompt you to enter the passphrase.
- The Easy Connect format and the Connect Descriptor format cannot be used to create a database connection of `SSH` type. To create a database connection of `SSH` type, you must first configure it in the DMU GUI and then reference it in the CONNECTION parameter.

Table 4-1   (Cont.) List of `dmucls` Command Parameters

Parameter	Description
CONNECT_TO	Oracle database connection string in one of the following formats – *Easy Connect* format or *Connect Descriptor* format. Use one of these database connection string formats to connect to a database when the database connection name is not already configured in the DMU using the DMU GUI.

* Easy Connect format:

    username@[//]*host*:*port*/*service_name*[as sysdba]
* Connect Descriptor format:

    username@(description=(address=(protocol=tcp)(host=*host*)(port=*port*))(connect_data=(service_name=*service_name*)))[as sysdba]

Here,

username is the Oracle database user name having the SYSDBA role.

host is the Oracle database server DNS name or IP address.

port is the Oracle database server port number.

service_name is the service name registered with the database listener for the Oracle database server.

[//] is optional and is used for URL-style host name values.

[as sysdba] is optional as DMU-CLS implicitly assumes that the database user connecting to the Oracle database server already has the SYSDBA role.

Examples:

* Easy Connect format examples:

    ```
    CONNECT_TO=sys@server1.us.example.com:1521/orcl
    ```

    ```
    CONNECT_TO=sys@192.168.40.2:1521/orcl
    ```

* Connect Descriptor format examples:

    ```
    CONNECT_TO=sys@(description=(address=(protocol=tcp)
    (host=server1.us.example.com)(port=1521))
    (connect_data=(service_name=sales.us.example.com)))
    ```

    ```
    CONNECT_TO=sys@(description=(address=(protocol=tcp)
    (host=192.168.40.2)(port=1521))
    (connect_data=(service_name=sales.us.example.com)))
    ```

Note:

* The DMU-CLS will prompt you to enter the database administrator password each time you run the `dmucls` command with the CONNECT_TO parameter.
* The Easy Connect format and the Connect Descriptor format cannot be used to create a database connection of SSH type. To create a database connection of SSH type, you must first configure it in the DMU GUI and then reference it in the CONNECTION parameter.

Parameter	Description
BUFFER	Memory size in bytes to allocate for each database server session for scanning a table.

You can specify any value from 1 byte to 9999999999 bytes (10 GB). The default value is the value stored in the DMU repository, if available; otherwise it is set to 1024000 bytes (1 MB).

Example:

`BUFFER=5120000`

Table 4-1   (Cont.) List of `dmucls` Command Parameters

Parameter	Description
PROCESSES	Number of concurrent scanning process threads to use for scanning the database. Note that each thread opens a separate database connection. You can specify any value from 1 to 9999. The default value is the value stored in the DMU repository, if available; otherwise it is set to the number of CPUs present on the database server. Example: `PROCESSES=4`
OPERATION	The DMU-CLS operation to perform. You can specify one of the following operations: • `SCAN`: Scan the database and generate a scan report by default. If you do not want to generate a scan report by default after completion of the database scan, then specify the additional parameter `SCAN_REPORT_TYPE=NONE`. • `INSTALL_MIGRATION_REPO`: Install the DMU repository in the Migration mode. • `INSTALL_VALIDATION_REPO`: Install the DMU repository in the Validation mode. • `UPGRADE_REPO`: Upgrade the DMU repository. • `UNINSTALL_REPO`: Uninstall the DMU repository. • `SCAN_REPORT`: Generate a scan report (containing information about scanned columns) without scanning the database and by using the earlier database scan results that are already stored in the DMU repository. • `INDEX_CHECK_REPORT`: Generate an index check report (containing information about indexes defined on scanned columns) without scanning the database and by using the earlier index check results that are already stored in the DMU repository. This report identifies index keys that will exceed the maximum supported size due to column modifications that are necessary for conversion. • `PROBLEM_DATA_REPORT`: Generate a problem data report. The default operation is `SCAN`. Example: `OPERATION=INSTALL_MIGRATION_REPO`
SCHEMAS	Schemas to include in the: • database scan or • scan report or • index check report or • problem data report Example: `SCHEMAS=hr,oe`
TABLES	Tables to include in the: • database scan or • scan report or • index check report or • problem data report Note that the tables must be specified in the form of a full identifier, that is, *schema_name*.*table_name*. Example: `TABLES=hr.employees,hr.departments`

Table 4-1   (Cont.) List of `dmucls` Command Parameters

Parameter	Description
COLUMNS	Columns to include in the: • database scan or • scan report or • problem data report This parameter cannot be used for the index check report. Note that the columns must be specified in the form of a full identifier, that is, *schema_name*.*table_name*.*column_name*. Example: `COLUMNS=hr.employees.first_name,hr.employees.last_name`
EXCLUDE	Tables to exclude from the: • database scan or • scan report or • index check report This parameter cannot be used for the problem data report. Note that the tables must be specified in the form of a full identifier, that is, *schema_name*.*table_name*. Example: `EXCLUDE=hr.locations,hr.countries`
EXCLUDE_SCANNED	Option to exclude database objects with valid scan results and necessary log information. You can specify one of the following values: • `TRUE`: Exclude already scanned database objects for the current database scan. • `FALSE`: Include already scanned database objects for the current database scan. The default value is `FALSE`. Example: `EXCLUDE_SCANNED=TRUE`

ROWIDS_COLLECTION	Rowid collection method for scanning tables. You can specify one of the following values:
	
	• `TABLE_SETTING`: Collect rowids only according to `Rowids to Collect` table property.
	• `FOR_CONVERSION`: Collect also rowids for `Update only Convertible Rows` conversion method.
	
	  Note that you can specify this option only in the Migration scan mode and not in the Validation scan mode.
	• `NONE`: Do not collect rowids.
	
	  Note that rowid collection cannot be disabled for a few special data dictionary tables in the `SYS` schema for which the `Rowids to Collect` property is always set to `All to Convert`.
	
	The default value is `TABLE_SETTING`.
	
	Example:
	
	`ROWIDS_COLLECTION=FOR_CONVERSION`
	
	See Also:
	
	The information provided in step 3 of the section "Scanning the Database with the Scan Wizard" for additional details about the rowid collection methods.

Table 4-1   (Cont.) List of `dmucls` Command Parameters

Parameter	Description
CHECK_INDEX	Option to check whether the key length of any index defined on a character column exceeds the maximum allowed key size because of a scheduled column modification or the increased post-converted data size of the column. You can specify one of the following values:
	• TRUE: Perform this check in the current database scan and generate an index check report.
	• FALSE: Omit this check in the current database scan.
	The default value is TRUE.
	Example:
	CHECK_INDEX=FALSE
	Note: The REPORT_NAME parameter cannot be used for index check reports. Index check reports always use the following naming conventions:
	When the CONNECTION parameter is used to connect to the database, the index check report file name format is:
	IndexCheckReport_*<db_connection_name>_<datetime>*.html
	When the CONNECT_TO parameter is used to connect to the database, the index check report file name format is:
	IndexCheckReport_*<db_name_identifier_in_db_service_name>_<datetime>*.html
REPORT_NAME	File name to use for:
	• a scan report (OPERATION=SCAN or OPERATION=SCAN_REPORT) or
	• a problem data report (OPERATION=PROBLEM_DATA_REPORT)
	You need not specify any file extension for the report file name as DMU-CLS always uses the file extension of .html for a scan report file and .xls for a problem data report file by default.
	If you do not specify this parameter, then the default report file name used is:
	• For a scan report:
	When the CONNECTION parameter is used to connect to the database, the default scan report file name format is:
	ScanReport_*<db_connection_name>_<datetime>*.html
	When the CONNECT_TO parameter is used to connect to the database, the default scan report file name format is:
	ScanReport_*<db_name_identifier_in_db_service_name>_<datetime>*.html
	• For a problem data report:
	When the CONNECTION parameter is used to connect to the database, the default problem data report file name format is:
	ProblemDataReport_*<db_connection_name>_<datetime>*.xls
	When the CONNECT_TO parameter is used to connect to the database, the default problem data report file name format is:
	ProblemDataReport_*<db_name_identifier_in_db_service_name>_<datetime>*.xls
	Example:
	REPORT_NAME=scan_report
	See Also:
	• "DMU-CLS Scan Report"
	• "DMU-CLS Problem Data Report"

Table 4-1   (Cont.) List of `dmucls` Command Parameters

Parameter	Description
REPORT_DIR	Directory in which a scan report, index check report, or problem data report file needs to be stored. You can specify either an absolute directory path or a directory path that is relative to the directory from which the `dmucls` command is run.
	If you do not specify this parameter, then the default directory used for storing a report file is the DMU log directory:
	• On a Windows system:
	`%APPDATA%\DMU\log`
	• On a Linux and a UNIX system:
	`$HOME/.dmu/log`
	Example:
	`REPORT_DIR=/usr/dmu/reports`
SCAN_REPORT_TYPE	Scan report type to use for generating a scan report. You can specify one of the following report types:
	• `NONE`: Do not generate a scan report by default after the database scan is completed.
	• `ALL`: Include all the scanned columns in the scan report, including the columns that do not need conversion.
	• `NON_CHANGELESS`: Include only those scanned columns in the scan report that need conversion.
	The default scan report type is `NON_CHANGELESS`.
	Example:
	`SCAN_REPORT_TYPE=ALL`
PROBLEM_DATA_LENGTH	Number of bytes of data that you want to see in a problem data report for the data having convertibility issues.
	You can specify any value from 1 byte to 16383 bytes (16 KB). The default value is 30 bytes.
	Example:
	`PROBLEM_DATA_LENGTH=100`
REPO_TABLESPACE	Tablespace in which the DMU repository needs to be installed in either the Migration mode or the Validation mode.
	Example:
	`REPO_TABLESPACE=SYSAUX`
REPO_TARGET_CHARSET	Unicode character set to use for the DMU repository, that is, for the target database. You can specify the value of either `AL32UTF8` or `UTF8`. The default value is `AL32UTF8`.
	Example:
	`REPO_TARGET_CHARSET=AL32UTF8`

Table 4-1   (Cont.) List of `dmucls` Command Parameters

Parameter	Description
ASSUMED_CHARSET	Assumed character set of the database to be scanned, that is, character set of the source database.
	If this parameter is not specified, then the assumed character set value stored in the DMU repository is used. If the DMU repository also does not contain the assumed character set value, then the value of the NLS_CHARACTERSET parameter present in the NLS_DATABASE_PARAMETERS table of the source database is used.
	If the value specified by you for the ASSUMED_CHARSET parameter is different from the one that is already stored in the DMU repository, the DMU-CLS replaces the existing assumed character set value stored in the DMU repository with the new value specified by you and invalidates the existing scan results present in the DMU repository.
	Note: • If the database to be scanned is configured to use the correct character set, then the assumed character set should be the same as the database character set. However, in a pass-through configuration, as described in "Invalid Binary Storage Representation of Data", all or almost all character columns might store data in the character set of client workstations, which might be different from the declared database character set. In such a case, after you identify the real character set of the database contents, specify it as the value for the ASSUMED_CHARSET parameter to let the DMU know how to correctly interpret the data. • You should specify this parameter only when you exactly know the difference between the assumed database character set and the actual character set of the data stored in the database. • You can reset the assumed character set value by setting it to NONE. Example: `ASSUMED_CHARSET=WE8ISO8859P1` **See Also:** "Setting the Assumed Character Set"
PARFILE	Parameter file to use. Specify this parameter when you want to use the parameters and their values stored in a parameter file instead of entering them on a command-line. You can specify either an absolute file path or a file path that is relative to the directory from which the `dmucls` command is run.
	Example: `PARFILE=/usr/dmu/params/scan_params.txt` **See Also:** "DMU-CLS Parameter File"

4.3.9.3.1 Specifying Special Characters in the dmucls Command on the Command-line

Most operating systems and command-line interpreters or shells do not interpret special characters. Multiple methods exist for specifying a special character in a parameter value of a `dmucls` command.

For example, in C shell and Bourne shell of UNIX, the character `?` is used for matching any single character and the character `i` is used for matching any number of characters. A whitespace is considered as a special character in most of the operating systems.

If you specify a special character in a parameter value of a `dmucls` command as it is, then DMU-CLS displays an error. There are multiple methods of specifying a special character in a parameter value of a `dmucls` command, so that the DMU-CLS interprets it literally by ignoring the special meaning given to that character in the operating system. In general, the parameter

value containing a special character is enclosed in double quotes or single quotes, or the special character is prefixed with an *escape character*, such as a backslash (\) in a UNIX or a Linux system, or a double quote in a Windows system.

The following are various methods of specifying special characters in parameter values in a `dmucls` command on the command-line.

- If a database object name (a schema name, a table name, or a column name) contains a special character, such as a whitespace, a dot (.), or a comma (,), then one of the following methods can be used to specify it on the command-line:

  - the database object name must be enclosed in escaped double quotes and an appropriate escape character must be prefixed to the special character.

    For example:

    * On a UNIX or a Linux system

      ```
      TABLES=hr.\"temp\ table\"
      ```

    * On a Windows system

      ```
      TABLES=hr.\"temp" table\"
      ```

  - (only for a UNIX or a Linux system) the whole parameter value must be enclosed in single quotes with the database object name enclosed in double quotes.

    For example:

    ```
    TABLES='hr."temp table"'
    ```

  - (only for a Windows system) the database object name must be enclosed in escaped double quotes, which additionally must be enclosed in double quotes.

    For example:

    ```
    TABLES=hr."""temp table"""
    TABLES=hr."\"temp table\""
    ```

  - (only for a Windows system) the database object name must be enclosed in escaped double quotes, and additionally the whole parameter value must be enclosed in double quotes.

    For example:

    ```
    TABLES="hr.""temp table"""
    TABLES="hr.\"temp table\""
    ```

- If a parameter value contains a special character for the parameters `CONNECTION`, `PARFILE`, `REPORT_NAME`, `REPORT_DIR`, or `CONNECT_TO`, then one of the following methods can be used to specify it on the command-line:

  - the parameter value must be enclosed in double quotes.

    For example:

    ```
    CONNECTION="db 122"
    CONNECT_TO="user name@host:port/service"
    REPORT_NAME="scan report"
    ```

- – an escape character must be prefixed to the special character.

  For example:

  * On a UNIX or a Linux system

  ```
  CONNECTION=db\ 122
  CONNECT_TO=user\ name@host:port/service
  REPORT_NAME=scan\ report
  ```

  * On a Windows system

  ```
  CONNECTION=db" 122
  CONNECT_TO=user" name@host:port/service
  REPORT_NAME=scan" report
  ```

- – (only for a UNIX or a Linux system) the whole parameter value must be enclosed in single quotes.

  For example:

  ```
  CONNECTION='db 122'
  CONNECT_TO='user name@host:port/service'
  REPORT_NAME='scan report'
  ```

- • In the parameter value for CONNECT_TO, if the user name contains the at symbol (@), then you can use one of the following methods to specify it on the command-line:

  - – The user name must be enclosed in escaped double quotes.

    For example:

    ```
    CONNECT_TO=\"user@name\"@host:port/service
    ```

  - – The user name must be enclosed in double quotes and the whole parameter value must be enclosed in single quotes (only for a UNIX or a Linux system).

    For example:

    ```
    CONNECT_TO='"user@name"@host:port/service'
    ```

> **See Also:**
>
> "Specifying Special Characters in Parameter Values in a Parameter File" for the methods of specifying special characters in parameter values in a *parameter file*

4.3.9.4 DMU-CLS Scopes for Database Scan and Report Generation

The DMU-CLS supports different scopes for scanning a database as well as for generating reports, such as a scan report and a problem data report.

Table 4-2   DMU-CLS Scopes for Database Scan and Report Generation

Scope	The dmucls Command Parameter
Database-level scope • For database scan: Scan a whole database. • For report generation: Generate a report containing data for the whole database. This is the default scope.	None of these parameters is specified: SCHEMAS or TABLES or COLUMNS. **Note:** Additionally, you can specify the EXCLUDE parameter to exclude certain tables from a database scan or from a scan report. The EXCLUDE parameter cannot be used for a problem data report. `EXCLUDE=schema1.table1, schema2.table2, ...`

Schema-level scope
• For database scan: Scan the specified schemas only.
• For report generation: Generate a report containing data for the specified schemas only.
	`SCHEMAS=schema1, schema2, ...`
	
	Note:
	Additionally, you can specify the EXCLUDE parameter to exclude certain tables from a database scan or from a scan report. The EXCLUDE parameter cannot be used for a problem data report.
	
	`EXCLUDE=schema1.table1, schema2.table2, ...`

Table-level scope • For database scan: Scan the specified tables only. • For report generation: Generate a report containing data for the specified tables only.	`TABLES=schema1.table1, schema1.table2, ...`
Column-level scope • For database scan: Scan the specified columns only. • For report generation: Generate a report containing data for the specified columns only.	`COLUMNS=schema1.table1.column1, schema2.table2.column2, ...`

> **Note:**
>
> You must specify only one scope in the `dmucls` command, otherwise the DMU-CLS reports an error.

4.3.9.5 DMU-CLS Scan Modes

The DMU-CLS automatically picks a scan mode based on the mode of installation of the DMU repository.

Table 4-3   DMU-CLS Scan Modes

Scan Mode	Description
Migration scan mode	The Migration scan mode is always used for scanning a database when the DMU repository is installed in the Migration mode.
Validation scan mode	The Validation scan mode is always used for scanning a database when the DMU repository is installed in the Validation mode.

4.3.9.6 DMU-CLS Execution Statuses

The DMU-CLS returns a status code after execution.

Table 4-4   DMU-CLS Execution Statuses

Status Code	Description
0	Operation successfully completed.
1	Operation failed.
2	In the Validation mode: Scan successfully completed. Invalid data found. In the Migration mode: Scan successfully completed. Invalid, over column limit, or over datatype limit data found.
3	New DMU connection to the database is not allowed. Another DMU GUI process or DMU-CLS process is already connected to the database.
4	Failed to connect to the database.

4.3.9.7 DMU-CLS Operations Examples

This topic demonstrates typical use cases for DMU-CLS.

The following are the examples of various operations that can be performed using the DMU-CLS.

- The following example shows how to scan a schema for collecting rowids using the `Update Convertible Rows` conversion method.

  ```
  $ sh dmucls.sh CONNECTION=db_connection_1 SCHEMAS=hr
  ROWIDS_COLLECTION=FOR_CONVERSION
  ```

- The following example shows how to scan multiple tables present in different schemas.

  ```
  $ sh dmucls.sh CONNECTION=db_connection_1 TABLES=hr.employees,oe.orders
  ```

- The following example shows how to scan all the tables in the `HR` schema, except tables `employees` and `departments`.

  ```
  $ sh dmucls.sh CONNECTION=db_connection_1 SCHEMAS=hr
  EXCLUDE=hr.employees,hr.departments
  ```

- The following example shows how to scan multiple columns present in different tables and in different schemas.

  ```
  $ sh dmucls.sh CONNECTION=db_connection_1
  COLUMNS=hr.employees.lastname,oe.orders.order_mode
  ```

4.3.9.8 DMU-CLS Parameter File

You can store all the command-line parameters of the `dmucls` command in a text file called the **parameter file**.

You can specify any name and extension for a parameter file. You can then specify this parameter file name as the only parameter to the `dmucls` command as shown below instead of specifying all the individual parameters on a command-line.

```
$ sh dmucls.sh PARFILE=scan_params.txt
```

In the preceding example, `scan_params.txt` is the DMU-CLS parameter file containing command-line parameters. You can specify either the absolute file path or the relative file path for a parameter file. When no file path is specified, the parameter file is assumed to be stored in the same directory where the `dmucls` command is being executed.

> **✎ Note:**
>
> If a parameter file name or path contains a whitespace character, then the whole file path should be enclosed in double quotes. For example:
>
> ```
> $ sh dmucls.sh PARFILE="/usr/dmu/params/scan params.txt"
> ```

The following example shows the contents of a sample parameter file:

```
CONNECTION=db_connection_1
SCHEMAS=hr,sales
OPERATION=scan
BUFFER=4096000
PROCESSES=2
```

Specifying this parameter file as a parameter to the `dmucls` command is equivalent to:

```
$ sh dmucls.sh CONNECTION=db_connection_1 SCHEMAS=hr,sales OPERATION=scan
BUFFER=4096000 PROCESSES=2
```

In a parameter file, each parameter and its value must be specified as a key-value pair in a single separate line. For multivalue parameters, the multiple values must be separated by a comma (,). Additional whitespaces between two comma-separated values are ignored. All parameter names and their values are case-insensitive, except the values for the parameters `CONNECTION`, `PARFILE`, `REPORT_NAME`, and `REPORT_DIR`.

You can add a comment anywhere in a parameter file as a single separate line with the pound character (#) in the beginning of the line. For example:

```
# Scan HR schema
SCHEMAS=hr
```

4.3.9.8.1 Specifying Special Characters in Parameter Values in a Parameter File

Specifying **special characters** in parameter values in a parameter file is similar to specifying special characters in parameter values in the `dmucls` command on the command line.

One difference is that in a parameter file, you need not use *escape characters* as prefixes for the double quotes and special characters. Also, there is no need to use the single quotes enclosing the parameter values on UNIX and Linux systems.

For example, in a parameter file, you can provide the following parameter value:

```
TABLES=hr."temp table"
```

Without using a parameter file, however, you must specify this parameter value on a command line as follows:

• On a UNIX or Linux system:

```
TABLES=hr.\"temp\ table\"
```

or

```
TABLES='hr."temp table"'
```

• On a Windows system:

```
tables="hr.""temp table"""
```

or

```
tables=hr."""temp table"""
```

or

```
tables="hr.\"temp table\""
```

or

```
tables=hr."\"temp table\""
```

or

```
tables=hr.\"temp" table\"
```

The following are some examples of specifying certain special characters in parameter values in a parameter file.

• Specifying at symbol (@) in the user name for the CONNECT_TO parameter value:

```
CONNECT_TO="user@name"@host:port/service
```

• Specifying a whitespace in a parameter value:

```
CONNECTION=db 122
CONNECT_TO=user name@host:port/service
REPORT_NAME=scan report
```

> **Note:**
>
> Oracle recommends that you use a parameter file for specifying DMU-CLS parameters rather than specifying them on a command-line, so as to reduce the complexity of handling special characters in parameter values.

> **See Also:**
>
> "Specifying Special Characters in the dmucls Command on the Command-line" for the methods of specifying special characters in parameter values in the `dmucls` command on the command-line

4.3.9.9 Managing a DMU Repository Using the DMU-CLS

Before using the DMU-CLS, you must install a DMU repository using either the DMU-CLS or the DMU GUI. This section describes the DMU-CLS technique.

The following are the `dmucls` command parameters for managing a DMU repository.

Table 4-5   The `dmucls` Command Parameters for Managing a DMU Repository

Parameter	Description
OPERATION	The following are the `OPERATION` parameter values for managing a DMU repository: • `INSTALL_MIGRATION_REPO`: Install the DMU repository in the Migration mode. • `INSTALL_VALIDATION_REPO`: Install the DMU repository in the Validation mode. • `UPGRADE_REPO`: Upgrade the DMU repository. • `UNINSTALL_REPO`: Uninstall the DMU repository.
REPO_TABLESPACE	The tablespace in which the DMU repository needs to be installed in either the Migration mode or the Validation mode. You must specify this parameter when installing the DMU repository.
REPO_TARGET_CHARSET	The Unicode character set to use for the DMU repository, that is, for the target database. You must specify this parameter when installing the DMU repository. The value for this parameter can be either `AL32UTF8` or `UTF8`. The default value for this parameter is `AL32UTF8`.

> **Note:**
>
> Unlike the DMU GUI, the DMU-CLS does not support the functionality of recovering the DMU repository. If you use the DMU-CLS and connect to a DMU repository that indicates the need for recovery, the DMU-CLS does not perform the recovery task, but continues execution of the specified DMU-CLS operation.

Example 4-1   Installing DMU Repository Using the DMU-CLS

The following example shows how to install a DMU repository in the Migration mode having the Unicode character set of `AL32UTF8`.

```
$ sh dmucls.sh CONNECTION=db_connection_1 OPERATION=INSTALL_MIGRATION_REPO
REPO_TABLESPACE=sysaux REPO_TARGET_CHARSET=AL32UTF8
```

The following example shows how to upgrade a DMU repository.

```
$ sh dmucls.sh CONNECTION=db_connection_1 OPERATION=UPGRADE_REPO
```

The following example shows how to uninstall a DMU repository.

```
$ sh dmucls.sh CONNECTION=db_connection_1 OPERATION=UNINSTALL_REPO
```

> ✎ **See Also:**
>
> "Installing the DMU Repository" for information about installing a DMU repository using the DMU GUI

4.3.9.10 DMU-CLS Scan Report

The DMU-CLS generates a scan report by default after completing a database scan (`OPERATION=SCAN`). You can also generate a report without scanning a database by using the earlier scan result stored in the DMU repository (`OPERATION=SCAN_REPORT`).

The following are the `dmucls` command parameters for generating a scan report.

Table 4-6   The `dmucls` Command Parameters for Generating a Scan Report

Parameters	Description
OPERATION	Specify one of the following operation parameter values. • SCAN: Generate a scan report by default after scanning a database. If you do not want to generate a scan report by default after completion of the database scan, then specify the additional parameter SCAN_REPORT_TYPE=NONE. • SCAN_REPORT: Generate a scan report (containing information about scanned columns) without scanning the database and by using the earlier database scan results that are already stored in the DMU repository. • INDEX_CHECK_REPORT: Generate an index check report (containing information about indexes defined on scanned columns) without performing a scan and index check on the database and by using the earlier index check results that are already stored in the DMU repository. This report identifies index keys that will exceed the maximum supported size due to column modifications that are necessary for conversion.

Table 4-6   (Cont.) The `dmucls` Command Parameters for Generating a Scan Report

Parameters	Description
CHECK_INDEX	Option to check whether the key length of any index defined on a character column exceeds the maximum allowed key size because of a scheduled column modification or the increased post-converted data size of the column. You can specify one of the following values:
	• TRUE: Perform this check in the current database scan and generate an index check report.
	• FALSE: Omit this check in the current database scan.
	The default value is TRUE.
	Example:
	CHECK_INDEX=FALSE
	Note: The REPORT_NAME parameter cannot be used for index check reports. Index check reports always use the following naming conventions:
	When the CONNECTION parameter is used to connect to the database, the index check report file name format is:
	IndexCheckReport_*<db_connection_name>_<datetime>*.html
	When the CONNECT_TO parameter is used to connect to the database, the index check report file name format is:
	IndexCheckReport_*<db_name_identifier_in_db_service_name>_<datetime>*.html
SCHEMAS or TABLES or COLUMNS	Specify one of the following scopes to determine what scan results to include in the scan report.
	• Database-level scope (none of these parameters specified: SCHEMA, TABLES or COLUMNS): Include the scan results of the whole database in the scan report or index check report.
	• Schema-level scope (parameter SCHEMAS): Include the scan results of the specified schemas only in the scan report or index check report.
	• Table-level scope (parameter TABLES): Include the scan results of the specified tables only in the scan report or index check report.
	• Column-level scope (parameter COLUMNS): Include the scan results of the specified columns only in the scan report. The column-level scope does not apply to index check reports.
EXCLUDE	Specify this parameter if you want to exclude certain tables from the scan report or index check report.
	This parameter can be used only for the database-level scope and the schema-level scope of scan report generation.

Table 4-6   (Cont.) The `dmucls` Command Parameters for Generating a Scan Report

Parameters	Description

REPORT_NAME and REPORT_DIR	Use REPORT_NAME to specify the file name for a scan report. Use REPORT_DIR to specify the directory path for a scan report or an index check report. These parameters are optional.
	Note:
	• You need not specify any file extension for a scan report as it is always set to `.html` by default.
	• If a scan report file name or its directory path contains spaces, then it must be enclosed in double quotes.
	• If you do not specify a file name for a scan report, then the default file name used is:
	  – When the CONNECTION parameter is used to connect to the database:
	    `ScanReport_<db_connection_name>_<datetime>.html`
	    Example of default scan report file name where the Oracle database connection name is `db_connection_1`:
	    `ScanReport_db_connection_1_20170615012127.html`
	  – When the CONNECT_TO parameter is used to connect to the database:
	    `ScanReport_<db_name_identifier_in_db_service_name>_<datetime>.html`
	    Example of default scan report file name where the Oracle database service name is `sales.us.example.com`:
	    `ScanReport_sales_20170615012127.html`
	• If you do not specify a directory for a scan report or index check report, then the default directory used is the DMU log directory:
	On a Windows system:
	`%APPDATA%\DMU\log`
	On a Linux and a UNIX system:
	`$HOME/.dmu/log`

SCAN_REPORT_TYPE	Specify one of the following scan report types. • NONE: Do not generate a scan report by default after the database scan is completed. • ALL: Include all the scanned columns in the scan report, including the columns that do not need conversion. • NON_CHANGELESS: Include only those scanned columns in the scan report that need conversion. The default scan report type is NON_CHANGELESS.

The following example shows how to generate a scan report and an index check report using the `dmucls` command.

```
$ sh dmucls.sh CONNECTION=db_connection_1 OPERATION=SCAN
TABLES=hr.employees,hr.departments REPORT_NAME=scan_report
REPORT_DIR=/usr/dmu/reports SCAN_REPORT_TYPE=ALL
```

4.3.9.11 DMU-CLS Problem Data Report

The DMU-CLS generates a problem data report as an `.xls` spreadsheet. The report shows details of database objects with convertibility issues.

The following are the `dmucls` command parameters for generating a problem data report.

Table 4-7   The `dmucls` Command Parameters for Generating a Problem Data Report

Parameters	Description
OPERATION	Specify the operation parameter value of PROBLEM_DATA_REPORT.
SCHEMAS or TABLES or COLUMNS	Specify one of the following scopes to determine what data having convertibility issues to include in the problem data report. • Database-level scope (none of these three parameters specified: SCHEMA, TABLES or COLUMNS): Include the data having convertibility issues in the whole database in the problem data report. • Schema-level scope (parameter SCHEMAS): Include the data having convertibility issues in the specified schemas only in the problem data report. • Table-level scope (parameter TABLES): Include the data having convertibility issues in the specified tables only in the problem data report. • Column-level scope (parameter COLUMNS): Include the data having convertibility issues in the specified columns only in the problem data report.

REPORT_NAME and
REPORT_DIR
	Specify the file name and the directory path for the problem data report. These parameters are optional for generating a problem data report.
	
	Note:
	• You should not specify any file extension for a problem data report as it is always set to .xls by default.
	• If a problem data report file name or its directory path contains spaces, then it must be enclosed in double quotes.
	• If a problem data report size crosses a certain limit, the DMU breaks the problem data report into multiple files having incremental numbers appended to their file names.
	    For example, problem_data_report_1.xls, problem_data_report_2.xls, and so on.
	• If you do not specify a file name for a problem data report, then the default file name used is:
	    – When the CONNECTION parameter is used to connect to the database:
	        ProblemDataReport_<db_connection_name>_<datetime>.xls
	        Example of a default problem data report file name where the Oracle database connection name is db_connection_1:
	
	        ProblemDataReport_db_connection_1_20170615012127.xls
	
	    – When the CONNECT_TO parameter is used to connect to the database:
	        ProblemDataReport_<db_name_identifier_in_db_service_name>_<datetime>.xls
	        Example of a default problem data report file name where the Oracle database service name is sales.us.example.com:
	
	        ProblemDataReport_sales_20170615012127.xls
	
	• If you do not specify a directory for a problem data report, then the default directory used is the DMU GUI log directory:
	    On a Windows system:
	    %APPDATA%\DMU\log
	    On a Linux and a UNIX system:
	    $HOME/.dmu/log

Table 4-7   (Cont.) The `dmucls` Command Parameters for Generating a Problem Data Report

Parameters	Description
PROBLEM_DATA_LENGTH	Specify the number of bytes of data that you want to see in the problem data report for the data having convertibility issues.
	You can specify any value from 1 byte to 16383 bytes. The default value is 30 bytes.

The following example shows how to generate a problem data report using the `dmucls` command.

```
$ sh dmucls.sh CONNECTION=db_connection_1 OPERATION=PROBLEM_DATA_REPORT
TABLES=hr.employees,hr.departments REPORT_NAME=problem_data_report
REPORT_DIR=/usr/dmu/reports PROBLEM_DATA_LENGTH=200
```

4.3.9.12 DMU-CLS Error Messages

DMU provides error message reporting.

The following are the various error messages that can be displayed by the DMU-CLS.

DMU-60000 failed to parse parameter <parameter name>

Cause: The parameter was specified in an invalid format or value.
Action: Refer to the documentation for the correct syntax.

DMU-60001 missing parameter <parameter name>

Cause: The parameter with no default value was required at that time but not specified.
Action: Specify the required parameter either in parameter file or command line.

DMU-60002 multiple operation scopes requested: <operation scopes>

Cause: The user specified multiple operation scopes at once.
Action: Check the parameters and remove the redundant ones.

DMU-60003 unknown parameter name <parameter name>

Cause: The user specified parameter name is incorrect.
Action: Check the parameter name.

DMU-60004 failed to find parameter file

Cause: The parameter file was not found for the given path.
Action: Check the parameter PARFILE to use the correct path.

DMU-60005 failed to parse parameter file

Cause: The parameter file was written in an invalid format.
Action: Refer to the documentation for the correct syntax.

DMU-60006 multiple values not allowed for parameter <parameter name>

Cause: The parameter was specified for multiple times.
Action: Check the parameters and remove the redundant ones.

DMU-60007 failed to find DMU connection name <connection name>

Cause: The DMU connection name couldn't be found.
Action: Check the connection name.

DMU-60008 multiple database connections specified

Cause: User specified the parameters CONNECTION and CONNECT_TO at the same time.
Action: Check the parameters and remove one of the connections.

DMU-60009 couldn't perform operation <operation name> for repository status <status>

Cause: The current repository status is illegal for the specified operation.
Action: Check the repository status and parameters.

DMU-60010 invalid identifier for parameter <parameter name>

Cause: The specified database identifier is incorrect for the parameter.
Action: Check the parameters and database identifiers.

DMU-60011 couldn't find database object <object name>

Cause: The database object could not be found in the database.
Action: Check the identifier of the database object.

DMU-60012 'FOR_CONVERSION' is not a valid option for scanning in validation mode

Cause: User specified rowids_collection=for_conversion when scanning database in validation mode.
Action: Check the parameter and change its value.

DMU-60013 couldn't find data with issues within the specified scope

Cause: There is no problem data (either not scanned or scanned but no problem data found) within the specified scope when trying to export Problem Data report.
Action: Check the identifier of the database object, or try scan first.

DMU-60014 unknown character set name <character set name>

Cause: The specified character set name is not a valid Oracle character set name.
Action: Check the character set name.

DMU-60015 <parameter value> is not a valid option for operation <operation>

Cause: The specified parameter value is invalid for the operation.
Action: Check the parameter values.

4.4 Cleansing the Data

You must resolve data issues identified during scanning before you can convert the database. **Cleansing actions** are actions that can take to resolve convertibility issues.

The DMU does not allow the conversion process to start until all issues have been resolved or explicitly marked as ignorable.

> ✎ **See Also:**
>
> - "Using the DMU to Cleanse Data" for details regarding cleansing
> - My Oracle Support Note 124721.1 for data cleansing instructions for migrating an Oracle E-Business Suite database to Unicode
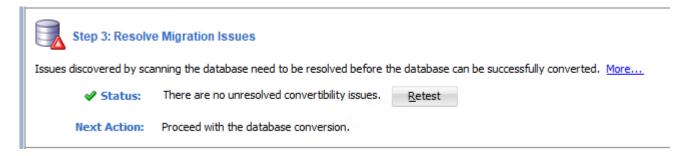
4.5 Converting the Database

After all issues in the database have been resolved, you can convert your database character data.

The Migration Status tab must resemble what is shown in Figure 4-35.

Figure 4-35   Migration Status Tab - No Unresolved Issues
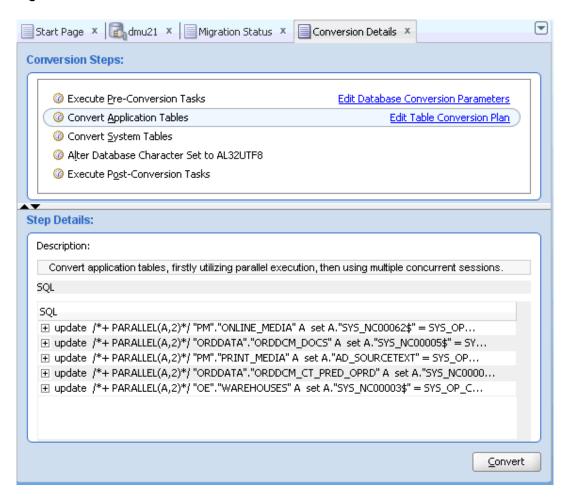
To convert the database:

1. To begin the conversion process, select **Convert Database** from the Migration menu or the context menu of the database node in the Navigator pane.

    The DMU generates a conversion plan, which is a list of SQL statements to run in the conversion phase, and shows it on the Conversion Details tab in the client pane of the DMU window, as shown in Figure 4-36. The Conversion Steps area on the tab lists the successive steps of the conversion process. When you click a step in the list, the Step Details area shows the list of SQL statements to be executed as part of the step. For a more detailed explanation of this tab's capabilities, see "Figure 4-36".

Figure 4-36   Conversion Details Tab

2. Click **Edit Database Conversion Parameters** to display the Converting subtab of the Database Properties tab.

   See "Viewing and Setting Database Properties" for a description of parameters that you can set on this subtab.

3. Click **Edit Table Conversion Plan** to show the conversion plan details for application tables.

   The upper part of the displayed dialog box contains a list of all tables that will be processed during the conversion. When you select a table from the list, the lower part shows the SQL statements that will be used to process the table and its dependent objects. You can customize this list by setting the properties Conversion Method, Target Tablespace, Preserve LONG Position, and Parallel Execution. See "Viewing and Setting Table Properties" for a description of these parameters.
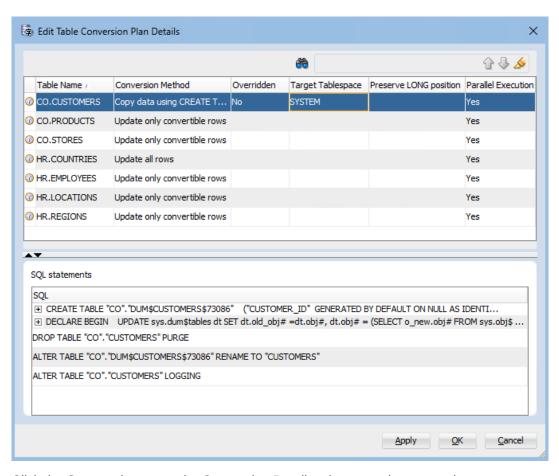
Figure 4-37　Edit Table Conversion Plan Details: Example 1

4. Click the **Convert** button on the Conversion Details tab to start the conversion process.

   The DMU might display warnings about unexpected sessions present in the database and about the age of the oldest scan results. Review the warnings as described below. Accept the warning messages to continue.

The SQL statement `ALTER DATABASE CHARACTER SET`, which the DMU uses in the conversion phase, succeeds only if the session executing the statement is the only user session logged into the database. Therefore, before starting the conversion, the DMU warns you about any user sessions logged into the database that are not opened by the DMU itself. You may use the following SQL statement in SQL*Plus or SQL Developer to find out the details of the offending sessions:

```
SELECT sid, serial#, username, status,
       osuser, machine, process, program
FROM v$session
WHERE username IS NOT NULL
AND program <> 'Database Migration Assistant for Unicode';
```

If the column `V$SESSION.PROGRAM` in a row returned by the above query contains the name of the Oracle Database executable, in an operating system dependent format, followed by "(J*nnn*)", where *nnn* are three decimal digits, for example "ORACLE.EXE (J000)", then the user session described by the row has been created by the database server itself to execute a job submitted through the PL/SQL package `DBMS_JOB` or `DBMS_SCHEDULER`. You can query the data dictionary views `DBA_JOBS_RUNNING` and `DBA_SCHEDULER_RUNNING_JOBS` to identify jobs currently

running in the database. Before continuing with conversion, ensure that all the jobs have finished their work and that their sessions have disappeared from `V$SESSION`.

To prevent further jobs from starting, issue the following SQL statement:

```
ALTER SYSTEM SET job_queue_processes=0 SCOPE=MEMORY
```

You can stop an already running `DBMS_SCHEDULER` job using the procedure `DBMS_SCHEDULER.STOP_JOB`. You cannot stop a `DBMS_JOB` job unless you shut down the database in immediate mode.

If you accept the warning message about unexpected sessions without disconnecting the sessions, the conversion will start but the step `ALTER DATABASE CHARACTER SET TO` *`target_character_set`* will fail reporting "ORA-12721: operation cannot execute when other sessions are active". At this moment, you can still disconnect the offending sessions and resume the conversion as described in the next section.

The warning message about the oldest scan result in the database lets you estimate the staleness of scan results. The DMU does not monitor DML changes to table data and it does not know if there are any data convertibility issues or any columns requiring conversion introduced into the database beyond those identified in the existing scan results. If the scan results are too old, you risk that some recently added data will remain unconverted or will be converted incorrectly. Oracle recommends that you perform a full database scan just before starting conversion, already in the migration downtime window, after all applications have been shut down and no new data comes into the database. If the database is very large and a full database scan takes long time, you may identify all large tables that are known to always contain only changeless data and deselect them in the Scan Wizard.

After you accept all warnings, if there are tables to be converted using the conversion method "Update only convertible rows" in the database, the DMU opens the Scan Wizard to rescan those tables. This is to ensure that the required rowid information for the tables is available and up-to-date. If you have just performed a full database scan in the downtime window or you have scanned all affected tables individually with the "All to Convert" rowid collection level, according to an time-optimized schedule, you may prefer to accept existing rowid information for the tables. Use the "Maximum age" parameter on the Scan Parameters page of the Scan Wizard to tell the DMU that it should accept available rowid information that is not older than the specified value in hours. After the scan has finished, the conversion process starts.

When the conversion starts, the Conversion Details tab changes its form and begins to display progress information for the conversion process. The link Edit Table Conversion Plan becomes View Table Conversion Progress. Clicking **View Table Conversion Progress** opens a table conversion progress dialog box, where you can view the detailed conversion progress of application tables.

When the conversion is completed, the DMU displays a confirmation message. If there is an error caused by any of the conversion SQL statements, it is signaled by a red icon next to the name of the step on the Conversion Steps list and next to the affected SQL statement in the Step Details list. Error details are displayed at the bottom of the Conversion Details tab, after you click the failed statement.
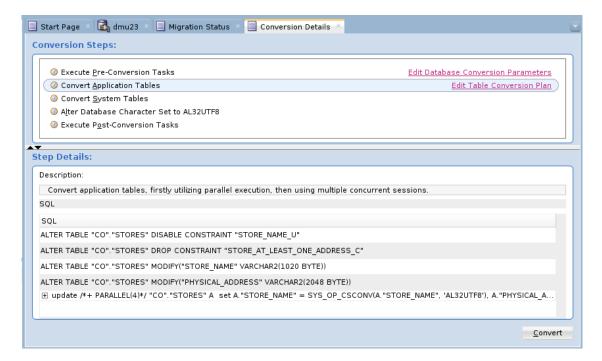
4.5.1 Conversion Details Tab

The Conversion Details tab works in two modes: planning and monitoring.

Immediately after you open the tab by selecting **Convert Database**, the tab works in the planning mode. The tab contains a list of SQL statements that the DMU plans to execute to convert the database to the target character set. The statements are grouped into steps listed

in the Conversion Steps area. When you click a conversion step, the DMU displays the SQL statements associated with the step in the Step Details area. If a plus icon is shown to the left of a SQL statement, you can click the icon to expand the area to show the full text of the statement.

Figure 4-38   Conversion Details Tab: Planning Mode
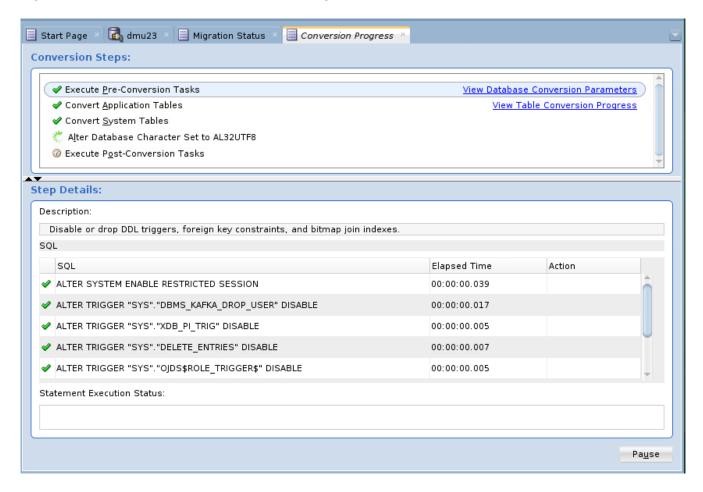
If you click the link **Edit Database Conversion Parameters**, the Converting subtab of the Database Properties tab is opened in a dialog box. See "Database Properties: Converting" for a description of available parameters.

If you click the link **Edit Table Conversion Plan**, the DMU opens the Edit Table Conversion Plan Details dialog box, described in "Edit Table Conversion Plan Details Dialog".

When you are ready to start conversion, click **Convert** at the bottom of the tab. The Conversion Details tab switches to monitoring mode, as shown in Figure 4-39.

Figure 4-39   Conversion Details Tab: Monitoring Mode
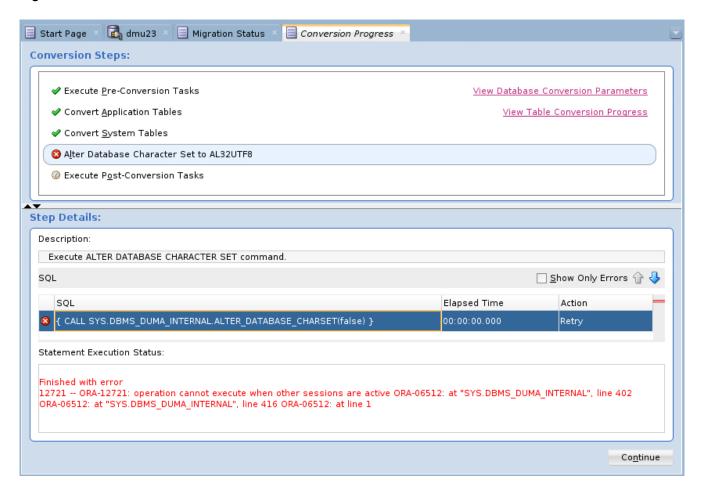
In the monitoring mode, the Convert button is replaced with a Pause button. If you click **Pause**, a warning dialog is displayed. In this dialog, you have the options of terminating the conversion process immediately (**Abort**) or returning to the conversion operation (**Back**).

In the monitoring mode, the link Edit Table Conversion Plan changes to View Table Conversion Progress. The link opens the View Table Conversion Progression dialog box in the monitoring mode.

The green check mark marks the steps and the SQL statements that have already been successfully executed. The animated circle icon marks the currently executed step, while the sprocket-like icon marks the currently executed SQL statement.

If the database reports an error during the conversion process, the currently executed step and the statement that caused the error are marked with the red X error icon, as shown in Figure 4-40. The conversion process is suspended.

Figure 4-40   Conversion Errors Tab

If you click the failing statement, the reported database error messages are displayed in the Statement Execution Status field. You can tell the DMU to retry or skip the statement by selecting the corresponding option from the drop-down list in the Action column of the SQL Statement area. The Skip option is not available for the ALTER DATABASE CHARACTER SET statement. For other statements, use the Skip option very cautiously because leaving a statement out of the conversion process could leave the database in an inconsistent state. After you select how to proceed with the failing statement, click **Continue** to resume the conversion process.

When the conversion process finishes, the DMU displays a confirmation dialog box. Status icons are removed from the Navigator pane. You can now reinstall the repository in Unicode Validation Mode by selecting **Configure DMU Repository** from the Migration menu.

4.5.2 Edit Table Conversion Plan Details Dialog

The Edit Table Conversion Plan Details dialog box enables you to modify conversion parameters for individual tables.

The dialog box is shown in Figure 4-41. To modify a parameter for a table, click the corresponding cell in the upper grid of the dialog box. The cell can then be edited, if the conversion method of the table allows this. When you click a row of a table in the upper grid of the dialog box, the lower grid displays the SQL statements associated with conversion of the table.

Figure 4-41   Edit Table Conversion Plan Details: Example 2

The conversion parameters that you can set for an individual table are:

- Conversion Method

- Target Tablespace

- Preserve LONG Position

- Parallel Execution

The first three parameters are described in "Table Properties: Converting". The Parallel Execution parameter specifies if parallel hints are added to the SQL statements for the given table to use the Parallel DDL and Parallel DML features of the database.

The Overridden column displays a value for a particular table only if a user-preferred conversion method was specified for the table. The value indicates whether the user-preferred conversion method was overridden by the DMU during the database scan (Yes) or not (No).

The search toolbar on the top of the dialog box enables you to quickly locate tables with a particular substring in their names. Enter the substring into the search box and press **ENTER**. If the substring is found in some names, the arrow icons are enabled to let you navigate between those names. If you click the marker pen icon, the matching tables are highlighted.
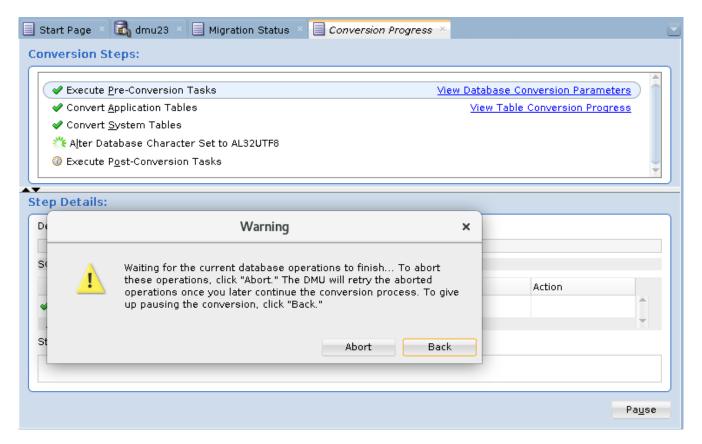
When you are ready, click **OK** to accept parameter changes and close the dialog box. Click **Apply** if you want to accept the changes but leave the dialog box open. When **Apply** or **OK** is clicked, the SQL statements for the changed tables are regenerated.

The Edit Table Conversion Plan Details dialog box can also be opened in monitoring mode during the conversion process by clicking the link **View Table Conversion Progress**. In the monitoring mode, the upper grid of the dialog box gets two additional columns: Elapsed Time and Progress, which show the time used to convert individual tables and progress bars reporting the percentage of the already processed table data. The lower grid also gets two additional columns: Elapsed Time and Action, which show the time used to execute individual SQL statements and an action to select if a statement fails, either **Retry** or **Skip**.

4.5.3 Stopping a Conversion

You can stop a conversion by clicking **Pause**. A dialog box lets you stop the conversion process immediately (**Abort**) or resume the conversion (**Back**).

Figure 4-42   Stopping a Conversion

4.6 Validating Data as Unicode

The DMU can validate the contents of an existing AL32UTF8 or UTF8 database, which might have been converted in the past or initially created in the Unicode character set. In either case, you can use the DMU to check the current data.

You can use the DMU with databases already converted to AL32UTF8 or UTF8, or with new databases in these character sets, to verify that all data is indeed in the declared database character set. Because the processes to validate and migrate data differ, the DMU user interface differs for the two usage types as well.

The type of the user interface presented by the DMU is decided at the repository installation time. If the current database character set is neither AL32UTF8 nor UTF8, then the DMU

activates the migration mode automatically. It activates the validation mode, if the current database character set is AL32UTF8. If the current database character set is UTF8, then the first page of the Repository Configuration Wizard (see "Installing the DMU Repository") gives you the choice between migration to AL32UTF8, which is the recommended Unicode character set, and activation of the validation mode. To activate the validation mode after you have migrated your database to Unicode, reinstall the DMU repository.

You can perform the validation of current character data by performing a scan, and see if any problems occur in the Database Scan report. The report shows which table columns contain sequences of bytes that do not form valid UTF-8 character codes. If any invalid data is reported, you can view the data and analyze where the problem is in the GUI. The DMU also offers you the means to correct the data in the Cleansing Editor. You can repeat the process of scanning and fixing the issues until no more invalid data is reported.

Fixing the data should usually be accompanied by fixing an application configuration problem that caused the invalid data to be stored in the database. Client configuration issues, usually a pass-through configuration one, are the most common reason for incorrectly encoded character data being stored in an AL32UTF8 or UTF8 database. You should periodically scan your AL32UTF8 and UTF8 databases in validation mode to discover any encoding issues as early as possible.

4.6.1 Introduction to the User Interface in Validation Mode
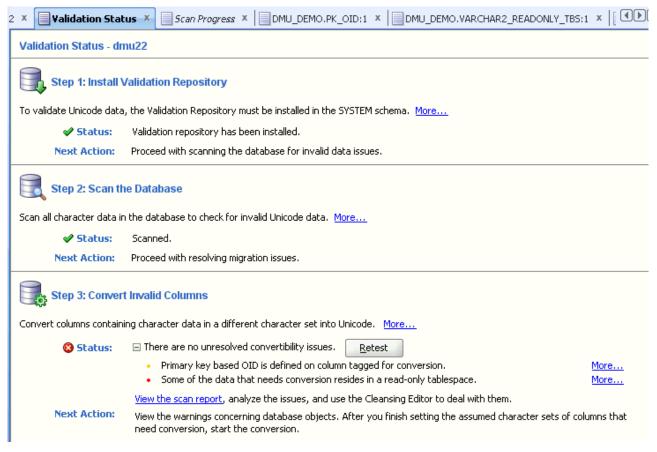
The DMU user interfaces in validation and migration modes are similar, but have some noteworthy differences.

The main differences are as follows:

• The Migration Status tab becomes the Validation Status tab, shown in Figure 4-43.

• The Migration menu becomes the Validation menu.

• The Convert Database item in the Migration menu becomes the Convert Invalid Columns item in the Validation menu – the item invokes the procedure to convert columns from their assumed character set to the database character set.

• The Database Scan Report shows only the database object name, scan results for current database contents and the Assumed Character Set property. Other report columns are not available.

Figure 4-43   Validation Status Tab

- Context menus in the Cleansing Editor tab do not have options to open Schedule Column Modification and Schedule Attribute Modification dialog boxes. Only the immediate operations Modify Column and Modify Attribute are possible.

- The button Show Impact of Scheduled Cleansing is absent from the toolbar of the Cleansing Editor.

- The Conversion Details tab for the Convert Invalid Columns process is very similar to the user interface for the conversion step of the migration mode, except that fewer conversion steps are present.

- The Table Conversion Plan dialog box is simplified compared to the migration mode. It shows the tables and columns to be converted and the associated SQL statements but it does not allow you to set any table-level conversion parameters. Also, this version of the dialog box does not show conversion progress. You can monitor conversion progress only on the Conversion Details tab.

- You can find any remaining issues preventing the conversion of the invalid columns under the "Status" field of step 3 on the Validation Status Panel. The reported issues need to be reviewed and resolved until the status indicates there are no more unresolved convertibility issues, at which point you can proceed to convert the columns tagged with assumed character sets to Unicode.

4.6.2 Validating Data

To validate data, you must scan the database, cleanse the data, and then convert incorrectly encoded data.

To validate data as Unicode:

1. Install the repository

   The DMU repository is installed in the validation mode, as described previously. After installation, the repository can be used many times in repeated application of the validation process. The repository must be reinstalled only if you upgrade the database or the DMU.

2. Scan the database

   The whole database or selected schemas, tables, or columns are scanned to look for illegal codes. Scanning of the whole database is highly recommended but you can choose to scan only a subset of objects if the database is too large to be scanned regularly without affecting production work and a preceding migration process revealed that only a small set of tables actually contains non-ASCII data.

   If the scan result shows only columns with data not requiring conversion, the contents of the database are correct and this part of the validation process is finished.

   The value of the property "Report U+FFFD as an invalid character" on the Scanning sub-tab of the Database Properties tab determines how the DMU interprets the Unicode default replacement character U+FFFD (the byte sequence 0xEF 0xBF 0xBD in AL32UTF8 and UTF8). If the property value is "Yes", the character is treated as invalid data. This is the default behavior, because the presence of this character in data usually indicates that the data is the result of character set conversion of some input that was not properly tagged with its real character set. If you use the character U+FFFD for some internal processing purposes and you do not want the DMU to report it as invalid, change the property value to "No".

   You initiate scanning and view the results as they were in the migration mode. See "Scanning the Database" for more details.

3. Cleanse the data

   If the scan results show any columns with data having invalid binary representation, you must diagnose the source of this data and fix the associated application or configuration problem. See "Cleansing Scenario 3: Cleansing Invalid Representation Issues" for information about possible reasons for data having invalid binary representation.

   You have two options to cleanse invalid representation issues: migrating a column to a binary data type, if its contents is binary, or converting it in the database character set, if it is encoded in some other character set. The first step to convert column data is to tag it with its proper character set by setting the column's Assumed Character Set property. Refer to "Cleansing Scenario 3: Cleansing Invalid Representation Issues" and "Setting the Assumed Character Set" for more information.

   If none of the columns has its Assumed Character Set property changed, the scan results never show any data length issues. However, if you change this property for a column and rescan the column, cells exceeding column limit or data type limit might be reported. You must resolve the length issues before you proceed to the next step. You can lengthen columns or migrate them to another data type in immediate mode, the same way you do this in migration mode. See Using the DMU to Cleanse Data and in particular "Modifying Columns" for more information.

All cleansing actions are immediate in validation mode. Setting the Assumed Character Set property is immediate in that the new value immediately affects scan results and data display, but the character set conversion is not performed at the time of setting. The character set conversion must be performed in a separate conversion step. After the Assumed Character Set of a column has been set properly, the column is reported in scanning as not having invalid binary representation issues. Go to the Validation Status tab to see if the conversion step is needed to convert any column from its assumed character set to the database character set. Before the conversion step is performed, the database is not yet clean.

Cleansing actions invalidate scan results of affected tables. Rescan the tables to confirm that the cleansing actions were successful. If you have set the Assumed Character Set property of a column, rescanning will tell you if column data can be converted from the assumed character set to the database character set without truncation issues.

4. Convert incorrectly encoded data

If any column had its Assumed Character Set property set during cleansing, this step should be executed to convert content of the column from the assumed character set to the database character set. In this step, all columns marked with an Assumed Character Set different from the database character set are physically converted to the database character set. The conversion is performed with SQL `UPDATE` statements. You start the process by selecting **Convert Invalid Columns** from the Validation menu or from the context menu of the database node in the Navigator pane, and then by clicking **Convert** on the displayed Conversion Details tab.

After the column content is converted, the Assumed Character Set property is reset to the database character set. You can rescan the converted columns to confirm that the conversion was successful.

Reapply the validation process regularly, but at least after each change to the configuration of your system, for example, upgrade of an application or the database, addition of a new application, or addition of new application users, especially from a new country. Let some time pass between the configuration change and the scanning so that there is high probability that some scanned data has been stored in the database already in the new configuration.

If the database character set is UTF8, the DMU repository is installed in the validation mode and if you decide to migrate the database to AL32UTF8, you must uninstall the repository and install it again, selecting the migration mode.

4.7 Creating a Diagnostic Package

To help resolve any problems that occurred while using the DMU, you can create a diagnostic package for subsequent analysis.

This package is a jar file that contains a project log file, a scan report, and an Oracle Data Pump dump file containing current and any archived DMU repository tables.

To create a diagnostic package:
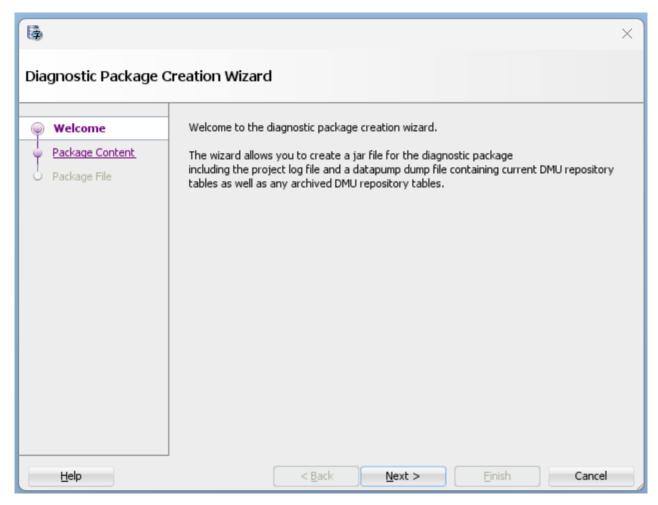
1. From the Tools menu, click **Create Diagnostic Package**.

   Figure 4-44 is displayed.
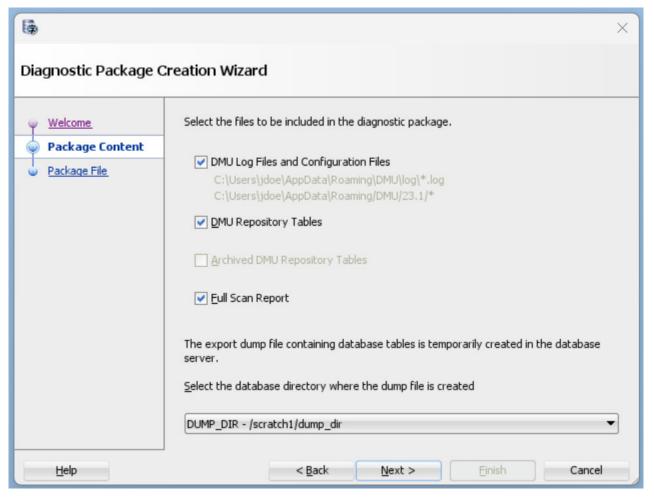
Figure 4-44   Diagnostic Package Wizard

2. Click **Next**.
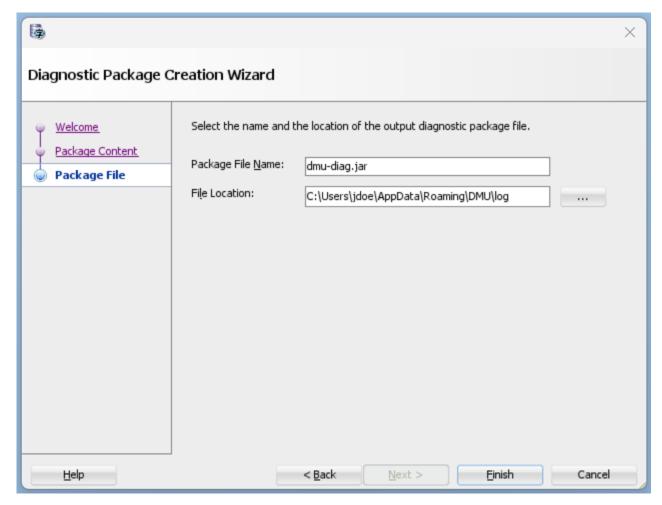
   Figure 4-45 is displayed.

Figure 4-45   Diagnostic Package Wizard - Package Content

3.   Click **Next**. Figure 4-46 is displayed.

Figure 4-46   Diagnostic Package Wizard - Package File

4. After clicking **Finish**, the diagnostic jar file is created in the path chosen.

5

Advanced Topics in the DMU

This chapter discusses advanced topics to consider when working with the Database Migration Assistant for Unicode (DMU).

5.1 Excluding Columns and Tables From Migration

In some situations, you may want to exclude selected columns or tables from scanning or conversion.

Situations include the following:

- You need to solve a problem of multiple character sets in a single column (see "Working with Multilingual Columns").

- There is a character column in the database that contains binary data. You cannot migrate the column to a binary data type (`RAW`, `LONG RAW`, or `BLOB`), because you are unable to modify applications accessing the database, for example, because they are not developed in-house. You want to avoid converting the column so that the applications may continue to access the data in a pass-through configuration.

- There is a very large table in the database, with terabytes of data, which – you are absolutely sure – contains no convertible data. Such tables:

  – Cannot contain `CLOB` columns, unless the current database character set is multibyte.

  – Can contain only character columns with internal application codes, yes/no flags, credit card numbers or other data known to contain only basic ASCII characters.

  – Must not contain data entered by end users from keyboards that support non-ASCII characters even if the users are supposed to enter English text only. Standard keyboards of computers running Microsoft Windows always allow such characters to be entered.

  You may want to avoid scanning such a large table to reduce the scanning time of the whole database. Oracle strongly recommends that you scan even very large tables at least once.

- As with the case of very large tables, you may want to avoid scanning archival data stored in read-only tablespaces on slow storage devices, such as DVD-ROM jukeboxes. Again, you should be absolutely sure that the tables contain no data requiring conversion.

- There are read-only tablespaces or a read-only table that you cannot change to read/write for some important reason. You want to be able to convert the rest of the database to Unicode, even though the tablespace or the table contains convertible data that the DMU will not be able to convert. You accept that data in the tablespace or the table will be unreadable after the conversion unless appropriate workarounds are implemented or the DMU converts the data later in validation mode.

- As with read-only tablespaces, if you have an offline tablespace or offline data file that you cannot switch back online, the DMU will not be able to scan or convert data in this tablespace or data file. You still want to convert the rest of the database and you accept that the contents of the tablespace or the data file may be unreadable if you later switch the tablespace or the data file back online. You can use the DMU in validation mode to later convert this data.

See "Handling Non-Accessible Data" for further discussion about database objects in read-only and offline mode.

To exclude a table from scanning, simply deselect it on the object selection page of the Scan Wizard.

To exclude a column from conversion, open its Column Properties tab (see "Viewing and Setting Column Properties") and set the Exclude from Conversion property on the Converting subtab to Yes. The DMU does not consider columns excluded from conversion when checking for convertibility issues that prevent you from starting the conversion step. To exclude a table from conversion, exclude all its convertible columns.

5.2 Handling Non-Accessible Data

Some database data might be inaccessible to applications. The data may be non-updatable or even unreadable.

The DMU cannot convert non-updatable data and it can neither convert nor scan unreadable data. Data with access restrictions is data contained in any of the following database objects:

5.2.1 Read-Only Tables Considerations

To protect a table contents against updates, you can alter a table to read-only mode. All Oracle Databases since release 11.1 support the read-only mode for database tables.

A read-only table can be queried, its constraints modified, and its segment moved, shrunk, or expanded. However, none of its column values can be modified in any way and no new rows can be added. You can alter the table back to read/write mode and again to read-only, as many times as required.

Read-only tables cannot be updated and so their contents cannot be converted by the DMU. Therefore, the DMU automatically alters read-only tables to read/write mode before attempting conversion. After the conversion, the mode is changed back to read-only.

A read-only table may also be re-created, if it is converted using the "Copy data using `CREATE TABLE AS SELECT`" conversion method.

5.2.2 Read-Only Tablespaces Considerations

If any segment of a table in a read-only tablespace contains data requiring conversion, then the DMU cannot successfully convert the table

You can place a tablespace into read-only mode, thereby preventing its data from being updated. Data files of read-only tablespaces may reside on read-only media or on standard read/write disk storage devices. For example, historical data in very large databases is frequently moved to read-only tablespaces. The tablespaces are backed up only once, just after putting them into read-only mode. Because read-only tablespaces cannot change, no further backups are required. This significantly reduces backup time.

If any segment of a table, including partition, `CLOB`, `VARRAY`, and IOT overflow segments, contains data that requires conversion, and if the segment belongs to a read-only tablespace, the DMU cannot successfully convert the table. The DMU reports this as a convertibility issue on the Migration Status tab and does not allow you to start the conversion step until the problem is resolved.

The approach to resolve this convertibility issue depends on the reason for which the problematic tablespaces have been put into read-only mode. If the reason was to reduce

backup time, and if the data is still on standard storage devices, then perform the following steps:

1. Put the tablespaces back into read/write mode.

2. Convert them with the rest of the database.

3. Put them into read-only mode again.

4. Refresh the backup.

If the tablespaces have been put into read-only mode and moved to read-only media, the possible solutions are:

- If enough disk storage can be arranged for, permanently or temporarily, to accommodate all read-only tablespaces with convertible data, copy the tablespaces to this storage, make them read/write, convert together with the rest of the database, make read-only again, and either leave on the disk storage, or put back on read-only media. With large number of read-only tablespaces containing convertible data, this may not be a viable solution.

- Create an auxiliary database in the same character set as the main database, that is, the database to be migrated. Move the read-only tablespaces logically to the new database using the transportable tablespace feature, leaving the data files physically in their current location. Create a database link in the main database pointing to the new database. By creating auxiliary views or through application changes, make the read-only data in the auxiliary database visible to applications connecting to the main database. The data will be converted while being transported over the database link.

5.2.3 Offline Tablespaces and Data Files Considerations

If a segment of a table in an offline tablespace or data file contains character data requiring conversion, then the DMU cannot scan or convert the table.

You can put a tablespace or selected data files in a tablespace into offline mode. Data contained in an offline tablespace or data file is inaccessible for both reading and writing. Offline mode is required by various administrative operations on data files, such as renaming or moving from one storage device to another.

If any segment of a table, including partition, `LOB`, `VARRAY`, and IOT overflow segments, contains character data that may require conversion, and the segment belongs to an offline tablespace or data file, the DMU can neither scan nor convert the table. You must put such a tablespace or data file back to online mode before scanning or converting the affected tables. Otherwise, errors, such as `ORA-00376`, will be reported in the scan and conversion steps.

The DMU will report an error on the Migration Status tab and prevent you from starting the conversion step if convertible data is found in an offline data file.

5.2.4 Working With External Tables

An external table is a table whose data resides in files outside of the database. External files pose special considerations for character sets.

The database contains definitions of table columns (metadata) but table rows are fetched from external files when a query referencing the table is issued. Oracle Database supports two access drivers that read the external files: `ORACLE_LOADER` and `ORACLE_DATAPUMP`. `ORACLE_LOADER` reads text files that are in the format supported by the SQL*Loader. `ORACLE_DATAPUMP` supports binary files that are compatible with the Data Pump Export and Import utilities (`expdp` and `impdp`).

DML statements modifying external tables are not supported. An `ORACLE_DATAPUMP` external file may be created and filled with data when an external table is created with the statement `CREATE TABLE ... ORGANIZATION EXTERNAL ... AS SELECT` but it cannot be later modified from inside the database. `ORACLE_LOADER` files must be created and modified outside of the database. The DMU does not convert external files along with the database contents.

When an external table is created, the character set of its data files is established as follows:

- The character set of an `ORACLE_DATAPUMP` file is stored in the data file itself when the file is created by the `ORACLE_DATAPUMP` driver or the Data Pump Export utility.

- You can specify the character set of an `ORACLE_LOADER` file in the `CHARACTERSET` parameter in the access parameters clause of the external table definition. If the parameter is not specified, then the database character set is used to interpret the contents of the file.

If the declared character set of an external file differs from the database character set, then the database converts the data automatically while reading it. If the database character set changes but the character set of the external file does not change, then the database adapts itself to the new configuration and converts external files to the new database character set.

> ⚠️ **Caution:**
>
> If the character set of an `ORACLE_LOADER` file is not declared explicitly in the access parameters clause of an external table definition, then a change to the database character set also changes the implicit declaration of the character set of the external file. If the file itself is not converted to the new database character set, then the external file declaration no longer corresponds to the real character set of the file and the external table is no longer correctly readable.
>
> Before migrating a database to Unicode, you must either add missing explicit character set declarations to all external table definitions or convert the file contents to the new database character set.

5.2.4.1 Cleansing External Tables

Even though the DMU does not convert external tables, it does include them in the scanning step of the migration process.

The scan results show you, if the file contents will still fit into the declared column lengths after migration, even if data expands in conversion to the new Unicode database character set. The results also warn you if any illegal character codes are present in the external files. These codes will no longer be readable if the file contents must be converted on the fly.

If the scan report shows invalid binary representation issues, you must identify the source of the invalid codes, as discussed in "Invalid Binary Storage Representation of Data" and "Cleansing Scenario 3: Cleansing Invalid Representation Issues".

The following sections describe actions that you can perform to cleanse the external tables from various reported issues.

5.2.4.2 Cleansing Length Issues

If the scan report shows length issues in external table contents, then you can alter the table to lengthen the affected columns or migrate them to character semantics.

The DMU does not support cleansing actions on external tables so you must do this in another tool, such as SQL*Plus or SQL Developer. Changing a `VARCHAR2` column to `CLOB` may be necessary, if the data expands above 4000 bytes. To change the column data type to `CLOB`, you must re-create the external table. This is a fast operation, as only metadata changes are involved, but you must remember to re-create any dependent objects, such as grants.

5.2.4.3 Correcting Character Set Declaration of ORACLE_LOADER Files

The character values of an external table may be read by the `ORACLE_LOADER` driver in a pass-through configuration.

In a pass-through, the declared character set of data files and the database character set are the same, but the real character set of the data content is different. You can repair the configuration by declaring the real character set in the access parameters clause of the external table definition.

You must not change the declaration in a production database before the database is converted, because this would break the pass-through configuration and make the table unreadable.

Before the database is converted, you should change the Assumed Column Character Set property of the affected external table columns (described in "Setting the Assumed Character Set") to the identified real character set of the data files and rescan the table to identify any additional length and invalid binary representation issues that might come up after the data file character set declaration is corrected.

Oracle recommends that you create a script to modify all affected access parameter clauses in the database and run it directly after the conversion phase of the DMU finishes successfully.

5.2.4.4 Correcting Character Set Declaration of ORACLE_DATAPUMP Files

The character values of an external table may be read by the `ORACLE_DATAPUMP` driver in a pass-through configuration.

In the character set configuration described in the previous section, ensure that the Data Pump input files are correctly re-tagged with their real character set directly after the database is converted to Unicode.

Unfortunately, the character set declaration is stored internally in Data Pump files and cannot be easily modified. A more complex procedure is needed to fix the declaration. Alternatively, you can arrange for the external table files to be provided in `ORACLE_LOADER` format, so that you have full control over their character set declaration in the external table's access parameters clause.

If a Data Pump file is used in a pass-through configuration, it means that its source (exported) database also works in this configuration, as otherwise it could not produce an incorrectly tagged file. The recommended approach to fix the character set declaration of a Data Pump file is, therefore, to fix the database character set of its source database.

If you have no control over the character set of the source database, you must:

- Create an auxiliary, empty database in the character set of the Data Pump files

- Import the files – this will happen in the pass-through configuration

- Change the database character set to the real character set of the files

- Export the files and use them for the external table in the main database (after it is migrated to Unicode)

Contact Oracle Support for information about fixing a pass-through configuration by changing the database character set with the Character Set Scanner utility (`csscan`) and the `csalter.plb` script.

The following PL/SQL code enables you to identify the character set of a Data Pump file:

```
DECLARE
    et_directory_name VARCHAR2(30) := '<directory object name>';
                                        -- for example, 'DATA_PUMP_DIR'
    et_file_name        VARCHAR2(4000) := '<file name>';
                                        -- for example, 'EXPDAT.DMP'
    et_file_info        ku$_dumpfile_info;
    et_file_type        NUMBER;
BEGIN
  dbms_datapump.get_dumpfile_info
    ( filename => et_file_name
    , directory => et_directory_name
    , info_table => et_file_info
    , filetype => et_file_type );
  FOR i IN et_file_info.FIRST..et_file_info.LAST LOOP
    IF et_file_info.EXISTS(i) THEN
      IF et_file_info(i).item_code = 11 THEN
        dbms_output.put_line( 'Character set of the file is ' ||
                                    et_file_info(i).value );
      END IF;
    END IF;
  END LOOP;
END;
```

5.2.4.5 Fixing Corrupted Character Codes

Analysis of invalid binary representation issues in an external table may show that some corrupted character codes exist in some character values.

Typically, corruptions occur because of a user error, an application defect, or a temporary configuration problem. In any case, correct the values in the source database and the affected files, and then re-export or unload them again.

With `ORACLE_LOADER` files, you can fix the invalid codes directly in the files with a text editor. However, the solution is effective only if the files are not regularly replaced with a new version produced from the same source database contents.

5.2.4.6 Handling Binary Data

Invalid binary representation issues in an external table may be caused by binary data being declared and fetched as character data by the external table driver.

To cleanse this type of issues, you must redefine the external table to use binary data types, such as `RAW` and `BLOB`, for the affected columns.

Oracle strongly discourages attempts to continue using the pass-through configuration to fetch the binary data into a database with a multibyte character set, such as UTF8 or AL32UTF8. Such configuration may cause unexpected issues now or in the future.

5.2.4.7 Performance Considerations for ORACLE_LOADER Files

The Oracle Database Utilities Guide lists several performance hints for the `ORACLE_LOADER` driver.

The following hints are especially relevant in the context of character set migration to Unicode:

- Single-byte character sets are the fastest to process.

- Fixed-width character sets are faster to process than varying-width character sets.

- Byte-length semantics for varying-width character sets are faster to process than character-length semantics.

- Having the character set in the data file match the character set of the database is faster than a character set conversion.

If you can choose between leaving an `ORACLE_LOADER` file in its current character set and arranging for the file to be provided in the new Unicode database character set, you should consider the following conclusions drawn from the above hints:

- If the current character set of the file is multibyte, using UTF8 or AL32UTF8 database character set for the file will not significantly influence the parsing time, that is, time needed to divide the file into records and fields, but it will save on conversion time. Performance of the queries referencing the external table will be better.

- If the current character set of the file is single-byte, using UTF8 or AL32UTF8 database character set for the file will slow down parsing but it will save on conversion time. You should benchmark both configurations to find out which one is more efficient.

- If you decide to convert the file from a single-byte character set to UTF8 or AL32UTF8, try to express field lengths and positions in bytes versus characters, if maximizing query performance is important.

5.3 Migrating Data Dictionary Contents

The DMU classifies tables as belonging to the data dictionary based on the schema that owns them. The DMU handles data dictionary tables differently from other tables in the database

Schemas that the DMU considers to be in the data dictionary appear under the Oracle-Maintained Schemas group node in the Navigator panel. If you create your own table in a data dictionary schema, such as `SYS` or `SYSTEM`, then the DMU treats it as other data dictionary tables. Oracle discourages creating user tables in data dictionary schemas. In this release, the DMU supports character set conversion of only a subset of metadata kept in the data dictionary tables.

> ✎ **See Also:**
>
> "Introduction to the DMU Interface and Navigation"

5.3.1 Scanning Data Dictionary Tables

Most data dictionary tables are scanned in the same way as user-defined tables.

The usual convertibility issues – data exceeding column limit, data exceeding data type limit, and data having invalid binary representation – are reported in the same way as well. The difference lies in reporting of data that needs conversion to the target Unicode character set. Because the DMU does not support converting of data dictionary contents in this release, except for a few exceptions, non-convertible columns containing data requiring conversion are marked with the yellow triangle warning icon and are considered a convertibility issue that prevents starting the database conversion step. The Database Scan Report filtering condition "With Some Issues" includes these columns as well – see "Database Scan Report: Filtering".

The View Data tab, described in "Viewing Data", which is a read-only version of the Cleansing Editor for data dictionary and other non-modifiable tables, shows convertible values in non-convertible data dictionary columns in dedicated colors configured on the Cleansing Editor tab in the Preferences dialog box, by default black on orange background.

Convertible data in the few columns that the DMU does convert, which are listed in "Data Dictionary Tables That Are Converted", is not reported as an issue.

For implementation reasons, the tables `SYS.SOURCE$`, `SYS.ARGUMENT$`, `SYS.IDL_CHAR$`, `SYS.VIEW$`, `SYS.PROCEDUREINFO$`, and `SYS.PLSCOPE_IDENTIFIER$` are always scanned with "Rowids to Collect" parameter set to "All to Convert". See "Data Dictionary Tables That Are Converted" for more details.

5.3.2 Cleansing Data Dictionary Tables

Oracle supports neither altering the structure of data dictionary tables nor updating their contents. Therefore, the DMU does not allow cleansing such tables.

You can set the assumed character set of columns of the data dictionary tables. However, the selected character set is considered only when displaying the columns on the View Data tab. The source character set used for conversion is always the assumed database character set.

5.3.2.1 Cleansing Data Length Issues

If data length issues are reported for data dictionary contents, then the only way to cleanse the issues is to replace the metadata with its shorter version.

The same length issues affect all character set migration methods, not only migration with the DMU. Therefore, you must cleanse the issues even if you plan to use alternative conversion methods, such as moving data with the Data Pump utilities.

Because the usual length limit for an identifier is only 30 bytes in Oracle databases earlier to Oracle Database 12c Release 2 (12.2), longer identifiers containing non-ASCII letters, especially those written in non-Latin scripts, may easily exceed the limit. For example, a Greek or Russian identifier longer than 15 characters will not fit into the 30 bytes limit after conversion from EL8MSWIN1253 or CL8MSWIN1251 to UTF8 or AL32UTF8. Chinese, Japanese, and Korean characters usually expand from 2 to 3 bytes, so identifiers longer than 10 characters become an issue.

The most common length issues are object names becoming longer than allowed after conversion. To shorten an identifier, rename the corresponding database object with an appropriate SQL statement or a PL/SQL package call. Although some objects, most must be dropped and re-created under the new name. When you drop an object, some dependent objects may also be dropped, in which case they must be re-created.

> **✎ Note:**
>
> Starting with Oracle Database 12c Release 2 (12.2), the maximum limit for an identifier is 128 bytes. For the earlier Oracle Database versions, the maximum limit for an identifier is 30 bytes.

For example, you can rename a table, view, sequence, or private synonym using the SQL statement `RENAME`. You can rename a table column using the SQL statement `ALTER TABLE RENAME COLUMN`. However, you cannot rename a table cluster. You must drop it and re-create under another name. But before you drop a cluster, you must drop all tables stored in the cluster. Because there are many possible auxiliary objects created for tables, such as privileges, indexes, triggers, row-level security policies, outlines, and so on, you may end up re-creating many objects. In this situation, Data Pump utilities and the Metadata API may be helpful.

After renaming a database object, you must change all application code that references this object to use the new name. This includes PL/SQL and Java code in the database, but also all affected client applications.

Another type of metadata that often expands beyond maximum allowed length is free text comments for various database objects. Similarly, you must update the comments with a shorter version to cleanse any reported length issues. Most comments can be updated with an appropriate SQL statement or PL/SQL package call.

> **✎ See Also:**
>
> - "Identifying Metadata" to learn how to identify the correct statements to cleanse data dictionary issues
>
> - *Oracle Database Utilities* to learn more about the Data Pump utilities and the Metadata API

5.3.2.2 Cleansing Invalid Binary Representation Issues

A common reason for invalid binary representation of data is the pass-through scenario.

If SQL statements or PL/SQL calls are issued in a pass-through configuration, it is possible to create and use a database object that is named using characters that are not valid or do not have the expected meaning in the database character set. For example, a seemingly senseless table name in a WE8MSWIN1252 database might be interpreted as correct on a Japanese JA16SJIS client. Alternatively, a PL/SQL module might contain comments that are not legible in the database character set but that make sense when viewed in another character set.

Invalid binary representation of database object names rarely occurs. Restrictions on characters allowed in non-quoted identifiers make it visible from the very beginning. Invalid binary representation of object comments is more probable but also easier to fix.

To cleanse the invalid binary representation issues caused by the pass-through configuration, if they are common to the application data as well, set the assumed database character set property of the database to the real character set of the database contents. Otherwise, update the affected metadata with a version that does not have convertibility issues.

To fix invalid representation issues in PL/SQL and Java source code or in view definitions, use Metadata API to retrieve the DDL statements creating the objects, correct the problematic characters in the statement text and execute the statements to re-create the objects. If object names are not affected, use the `CREATE OR REPLACE` syntax to change the code without having to re-create related objects, such as privileges.

> ✏️ **See Also:**
>
> - "Database Properties: General"
> - "Invalid Binary Storage Representation of Data"

5.3.2.3 Identifying Metadata

A difficult step in the process of resolving data dictionary convertibility issues is to map the issues shown in a database scan report to the type of metadata that is stored in the affected tables and columns.

The data dictionary tables, presented in the DMU interface, are generally not documented. The documented way to view their contents is through data dictionary views such as `DBA_TABLES`, `DBA_TAB_COLUMNS`, `DBA_RULES`, `DBA_SCHEDULER_JOBS`, and many others.

To identify the type of metadata that has convertibility issues, try one of the methods below, in the presented order:

- Look at the problematic character value on the View Data tab. The value itself may already tell you the metadata that it belongs to. For example, the value may be `This column keeps customer e-mail`, which suggests that it is a column comment, or it may be `HR_EMPLOYEE_V`, which may correspond to a common convention to name views, thus showing the value is a view name.

- Search for the name of the table in which the issues are reported in SQL script files named `cat*.sql` and `cd*.sql` and located in the `rdbms/admin/` subdirectory of your database Oracle home directory. By mapping the table and its columns to the right documented view, you can find out which metadata has the reported issues.

- Contact Oracle Support or post a question on the globalization forum on the Oracle Technology Network (OTN) website.

Once you identified the metadata that has the reported convertibility issues, refer to Oracle documentation to identify the right procedure to change this metadata.

> ✏️ **See Also:**
>
> - "Viewing Data"
> - *Oracle Database Reference* to learn about the scripts that define data dictionary views

5.3.3 Data Dictionary Tables That Are Converted

The DMU converts some data in the data dictionary.

The DMU converts only the following data in the data dictionary:

- `CLOB` columns – this is necessary only in a single-byte database

- Binary XML token manager tables, with names like `XDB.X$QN%` and `XDB.X$NM%`

- PL/SQL source code (text of `CREATE PROCEDURE`, `CREATE FUNCTION`, `CREATE PACKAGE`, `CREATE PACKAGE BODY`, `CREATE TYPE BODY`, `CREATE TRIGGER`, and `CREATE LIBRARY`); type specifications (`CREATE TYPE`) are not converted

- View definitions (text of `CREATE VIEW`)

- The following columns:
  - `SYS.SCHEDULER$_JOB.NLS_ENV` – NLS environment for Database Scheduler jobs (`DBMS_SCHEDULER`)
  - `SYS.SCHEDULER$_PROGRAM.NLS_ENV` - NLS environment for Database Scheduler job programs (`DBMS_SCHEDULER`)
  - `SYS.JOB$.NLS_ENV` – NLS environment for legacy jobs (`DBMS_JOB`)
  - `CTXSYS.DR$INDEX_VALUE.IXV_VALUE` - attribute values of Oracle Text policies
  - `CTXSYS.DR$STOPWORD.SPW_WORD` - all stopwords in all stoplists of Oracle
  - `SYS.COM$.COMMENT$` - user-specified comments
  - `SYS.ERROR$.TEXT` - error text
  - `SYS.RULE$.R_COMMENT` - rule comments
  - `SYS.TRIGGER$.WHENCLAUSE` - text of when clause for triggers
  - `SYS.TRIGGER$.DEFINITION` - definition text for triggers
  - `SYS.TRIGGER$.ACTION#` - action to fire for triggers
  - over 50 different columns in `SYS`, `SYSTEM`, and `CTXSYS` schemas that contain user comments for various database objects

The PL/SQL source code and the view source text are kept in multiple tables. The DMU checks the following columns when processing the source code and view definitions:

- `SYS.VIEW$.TEXT` – view definition text
- `SYS.SOURCE$.SOURCE` – PL/SQL and Java source code
- `SYS.ARGUMENT$.PROCEDURE$` – PL/SQL argument definitions: procedure name
- `SYS.ARGUMENT$.ARGUMENT` – PL/SQL argument definitions: argument name
- `SYS.ARGUMENT.DEFAULT$` – PL/SQL argument definitions: default value
- `SYS.PROCEDUREINFO$.PROCEDURENAME` – names of procedures and functions declared in packages
- `SYS.IDL_CHAR$.PIECE` – internal representation of PL/SQL
- `SYS.PLSCOPE_IDENTIFIER$.SYMREP` – internal representation of PL/SQL; this table did not exist before version 11.1 of Oracle Database

The DMU does not report convertible character data in the preceding tables and columns as a convertibility issue. Any convertible data in the remaining tables and columns of the data dictionary is flagged as a convertibility issue in scan reports and on the Migration Status tab. The database conversion step cannot be started before the flagged data is removed.

5.3.4 Data Dictionary Tables That Are Ignored

The DMU does not scan all data dictionary tables.

The DMU does not scan the following tables:

- `SYS.HISTGRM$`, which contains column histogram statistics

- Automatic Workload Repository object statistics history kept in tables with names such as `SYS.WRI$_OPTSTAT_OPR` and `SYS.WRI$_OPTSTAT_%_HISTORY`

- DMU repository tables in the `SYSTEM` schema

- `CSSCAN` repository tables in the `CSMIG` schema

The contents of these tables are also not considered when the DMU decides if the database conversion is allowed to start.

The `SYS.HISTGRM$` table is not scanned because its `EPVALUE` column (storing end-point values) may contain binary data. As histograms and other table statistics depend on binary representation of data in character columns, you should anyway re-gather statistics for all converted tables in the database after migration. Collection of statistics refreshes the contents of the `SYS.HISTGRM$` table and revalidates it in the new database character set. If you do not refresh the statistics, the optimizer may exhibit incorrect behavior.

Similarly, the historical object statistics kept in Automatic Workload Repository become stale after the migration because they also depend on binary representation of character data. The DMU does not migrate those statistics. You should purge them manually after migration by calling `DBMS_STATS.PURGE_STATS(SYSTIMESTAMP)`.

> ✎ **See Also:**
>
> - *Oracle Database PL/SQL Packages and Types Reference* for more information about the package `DBMS_STATS` and the procedure `PURGE_STATS`
>
> - *Oracle Database Concepts* for more information about optimizer statistics

The DMU and `CSSCAN` repository data becomes invalid after the database is migrated to a new database character set. Therefore, there is no point in migrating it along with the database. After the migration, you should drop the repositories and re-create them, if you still need them. If you re-create the DMU repository after migration, choose the validation mode – see "Validating Data as Unicode".

5.3.5 Handling Automatic Workload Repository Tables

The `SYS` schema contains a number of tables with names beginning with `WRI$`, `WRH$%`, and `WRR$_`, which comprise the Automatic Workload Repository (AWR).

In addition to historical object statistics, mentioned in "Data Dictionary Tables That Are Ignored", this repository stores snapshots of vital system statistics, such as those visible in various fixed views, for example, `V$SYSSTAT` and `V$SQLAREA`.

If non-ASCII characters are used in object names or in SQL statements, such as character literals or comments, they may get captured into the AWR tables. The DMU scan will report such characters as convertible data dictionary content, which prevents conversion of the

database. To remove this data completely, re-create the Automatic Workload Repository by logging into SQL*Plus with `SYSDBA` privileges and running:

```
SQL> @?/rdbms/admin/catnoawr.sql
SQL> @?/rdbms/admin/catawr.sql
```

As the `catawr.sql` script is not present in Oracle Database versions 10.2.0.4 and earlier, Oracle recommends that you install the Oracle Database patch set 10.2.0.5 before purging AWR contents.

5.4 Migrating Blockchain and Immutable Tables

Starting with DMU 23.1, you can migrate a database that contains blockchain and immutable tables.

Blockchain tables are insert-only tables that organize rows into a number of chains. Each row in a chain, except the first row, is chained to the previous row by using a cryptographic hash. Immutable tables are insert-only tables that provide an efficient variation on blockchain tables by eliminating the cryptographic hash capability. Blockchain and immutable table rows cannot be modified once inserted. Deleting a row from a blockchain or immutable table is either prohibited or restricted, based on the insertion time of the row. For these reasons, blockchain and immutable tables require special handing during migration to ensure that table integrity is not compromised.

The DMU performs the special handing of blockchain and immutable tables automatically and seamlessly. However, Oracle recommends that you review the information in this section before starting the migration of a database containing blockchain and immutable tables so you can be aware of the process.

Data Preparation: Scanning Blockchain and Immutable Tables

The scanning process for blockchain and immutable tables is the same as the scanning process for other types of tables. Blockchain and immutable tables that contain non-ASCII data in `CHAR`, `VARCHAR2`, `CLOB`, or `LONG` columns will be marked for conversion.

If a blockchain or immutable table is empty, or if it contains only ASCII data, it will not be marked for conversion.

Data Preparation: Cleansing Blockchain and Immutable Tables

If the scanning process reports an issue with a blockchain or immutable table, you can use the Cleansing Editor to perform immediate or scheduled cleansing, and you can also perform immediate or scheduled Bulk Cleansing.

The following table lists immediate and scheduled Cleansing Editor actions, and indicates whether they are supported for blockchain and immutable tables (Yes) or not (No).

Table 5-1   Cleansing Editor Support for Blockchain and Immutable Tables

Action	Immediate Cleansing	Scheduled Cleansing
Migrate column to character semantics	Yes	Yes Not supported for `CHAR` data type columns

Table 5-1   (Cont.) Cleansing Editor Support for Blockchain and Immutable Tables

Action	Immediate Cleansing	Scheduled Cleansing
Increase length of column	Yes	Yes
		CHAR data type columns can be lengthened by bytes only
Change column data type	No	No
Assumed character set for column	No	No
Edit Data, Copy, and Paste context menu items	No	N/A

The following table lists immediate and scheduled Bulk Cleansing actions, and indicates whether they are supported for blockchain and immutable tables (Yes) or not (No).

Table 5-2   Bulk Cleansing Support for Blockchain and Immutable Tables

Action	Immediate Bulk Cleansing	Scheduled Bulk Cleansing
Migrate to character length semantics	Yes	Yes
		Not supported for CHAR data type columns
Set the "Allow Conversion of Data with Issues" property	No	Yes, but only for invalid data
Perform pattern-based replacement cleansing	No	No

Data Conversion: Preparing Blockchain and Immutable Tables for Conversion

During this phase, the DMU executes a conversion feasibility test on all tables, including blockchain and immutable tables. As with other types of tables, you can modify the table-level conversion options for a blockchain or immutable table by clicking the link **Edit Table Conversion Plan**. In the Edit Table Conversion Plan Details dialog box, note that:

- **Scan and update only convertible rows** is the only supported conversion method for blockchain and immutable tables.

- Blockchain and immutable tables cannot be excluded from conversion.

- The "Allow Conversion of Data with Issues" property is supported for a blockchain or immutable table column only if the column contains invalid data.

- The SQL statements that will be used to process blockchain and immutable tables are not displayed. This holds true further along in the conversion process, when the **Edit Table Conversion Plan** link becomes the **View Table Conversion Progress** link; the SQL statements are also omitted from the View Table Conversion Progress dialog box.

Data Conversion: Converting Blockchain and Immutable Tables

Blockchain and immutable table data is converted to Unicode during the ALTER DATABASE CHARACTER SET operation. See Data Conversion: Converting for a complete list of conversion steps.

Data Conversion: Error Handling for Blockchain and Immutable Tables

Because blockchain and immutable tables are converted *during* the ALTER DATABASE CHARACTER SET operation, there are special considerations for error handling.

If the conversion of a blockchain or immutable table fails, the database conversion process is suspended and the DMU displays the Conversion Errors Tab, as shown in Figure 4-40.

In this tab, the `ALTER DATABASE CHARACTER SET` statement is displayed as the failing statement. If you click this statement, the reported database error messages are displayed in the Statement Execution Status field, including the names of the failed blockchain and immutable tables, along with their applicable error codes.

Note that the only possible action for the `ALTER DATABASE CHARACTER SET` statement is **Retry**; you cannot **Skip** this statement. Therefore, the recommended course of action is to resolve the issues with the problematic blockchain and immutable tables, and then retry the failed statement by clicking **Continue**.

Another course of action is to terminate the conversion process in the DMU. However, this action must be considered carefully, because once the conversion process is terminated, it cannot be subsequently resumed. If you attempt to terminate the conversion process, a warning dialog is displayed. In this dialog, you are presented with the following options:

- **Yes** - Terminate the conversion and disconnect from the database.

  The database will not be available to non-`SYS` users. If you would like to make the database available to non-`SYS` users:

  1. Ensure that the database is open in `RESTRICTED` mode.

  2. Run the `ALTER_DATABASE_CHARSET` procedure with parameter `FORCE_COMPLETE` = `TRUE`:

     ```
     EXEC SYS.DBMS_DUMA_INTERNAL.ALTER_DATABASE_CHARSET(TRUE)
     ```

     When the procedure completes, all blockchain and immutable table data has been converted to Unicode, and the database character set has been changed to Unicode.

  3. Start the DMU and connect to the database. The DMU will proceed with the Post-Conversion Tasks and finish the conversion.

- **No** - Immediately stop the conversion.

  You can later resume the conversion by clicking the **Continue** button.

- **Cancel** - Return to the conversion operation.

5.5 Working with Multilingual Columns

While analyzing contents of a column in the Cleansing Editor, you may discover that none of the assumed character sets set for the column makes all values in the column appear correctly at the same time. Instead, each value seems to be correct in one of the selected character sets.

This result indicates that the column contains a mixture of data in different character sets. You might also gather this information from analysis of data sources for your database.

Multiple character sets in a single column are possible in the pass-through scenario, if clients working in various character sets all store data in this column.

This release of the DMU does not contain any feature dedicated to cleansing this type of convertibility issue. The following procedure is recommended when you must deal with multiple character sets in a single `CHAR` or `VARCHAR2` column:

To work with multiple character sets in a single CHAR or VARCHAR2 column:

1. Find any auxiliary data that can help you identify the real character set of a single value in an affected column. Examples of such data are:

   • a country code associated with the value

   • an identifier of the operator who entered the value

   • an identifier of a subsidiary responsible for entering the value

2. Create a mapping table that maps auxiliary data to possible character sets of the values. If your company standardizes on a certain type of workstations, the source country of the analyzed value usually defines the client character set used to enter the value.

3. Mark columns that contain data in multiple character sets for exclusion from conversion – see "Column Properties: Converting".

4. Verify that there are no length issues with the columns. If required, cleanse them by making longer or by shortening problematic values. Do not migrate to `CLOB`. See Example 5-1 below for information about how to check for length issues.

5. Convert the database.

6. Using the mapping table, convert the affected columns with the SQL function `CONVERT` specifying the target Unicode character set in its second argument and the identified value character set in its third argument.

Example 5-1   Multilingual Column Considerations

Assume your database contains the table `CUSTOMERS`, with columns `CUSTOMER_NAME_ORIGINAL` and `CREATED_BY`. The column `CUSTOMER_NAME_ORIGINAL`, defined as `VARCHAR2(80 BYTE)`, contains the names of customers in their mother tongue in multiple character sets. The column `CREATED_BY` contains system IDs of employees who entered the customer data. You want to migrate the database to the character set AL32UTF8.

To solve the issue of multiple character sets in the `CUSTOMER_NAME_ORIGINAL` column start with creating a mapping table that maps employees' system IDs to client character sets of the employees' workstations. A table in your application that defines the system IDs may be helpful in locating the country in which the employee works and thus determining the character set of the client workstation that the employee uses. Further assume the created mapping table is named `CREATED_BY_TO_CHARSET_MAPPING` and has the columns `CREATED_BY` and `CHARACTER_SET`. The contents of such a table might resemble the following:

```
CREATED_BY     CHARACTER_SET
----------     -------------
...
JSMITH         WE8MSWIN1252
JKOWALSKI      EE8MSWIN1250
SKUZNETSOV     CL8MSWIN1251
WLI            ZHS16GBK
...
```

Now, set the Exclude from Conversion property of the `CUSTOMER_NAME_ORIGINAL` column to Yes to prevent data from being corrupted while the rest of the database is converted.

Check for possible length issues in `CUSTOMER_NAME_ORIGINAL` by running the following SQL:

```
SELECT c.ROWID
  FROM customers c, created_by_to_charset_mapping csm
 WHERE VSIZE(CONVERT(c.customer_name_original,
                     'AL32UTF8',
```

```
                          csm.character_set)) > 80
     AND c.created_by = csm.created_by
```

If no rows are returned, there are no length issues. Otherwise, the returned rowids will help you locate the problematic values.

When the database is ready, convert it to AL32UTF8. Change the application configuration as required for the new database character set.

After the database conversion, run the following update statement:

```
UPDATE customers c
    SET c.customer_name_original =
        (SELECT CONVERT(c.customer_name_original,
                        'AL32UTF8',
                        csm.character_set)
          FROM created_by_to_charset_mapping csm
         WHERE csm.created_by = c.created_by)
```

This will convert the individual values of the column according to their assumed character set. Let employees who entered the customer data or who speak the relevant languages verify the post-conversion values for correctness.

5.6 Advanced Convertibility Issues

Some less frequently encountered convertibility issues are not handled automatically by the DMU. These issues might require additional scanning and cleansing steps outside of the tool.

5.6.1 Convertibility Issues: Uniqueness Validation

After database contents have been converted to Unicode, it is possible that rows in a table no longer satisfy a unique or primary key constraint.

Specifically, the following situations can occur:

- A unique or primary key column has data with length issues, that is, some values expand in conversion beyond the column or data type length limit, and you set the property "Allow Conversion of Data with Issues" of this column to Yes. In such a case, the DMU will automatically truncate column values during the conversion step so that they fit into the existing length constraint. However, a truncated value may become identical with another value already in the column from which it differed only be the truncated suffix.

- In various Oracle character sets, there are multiple character codes that map to a single Unicode code point. This is usually a result of:

  – an attempt to provide compatibility mapping for historical changes to a character set definition or to its interpretation by different vendors

  – an attempt to provide simplified mapping for codes that cannot be exactly mapped to Unicode, for example, because the actual mapping consists of a sequence of Unicode codes, and this is not supported by Oracle's conversion architecture

  – Unicode Standard unification rules, which cause certain groups of Han (Chinese) characters to get a single code point assigned, even though characters in such a group may be separately encoded in legacy East Asian character sets

  If two character values in a single column differ only by characters that have the same mapping to Unicode in the assumed character set of the column, they become identical after conversion to Unicode.

The following character sets supported by the DMU have multiple codes that map to the same Unicode code point:

– BG8PC437S

– IW8MACHEBREW

– IW8MACHEBREWS

– JA16EUCTILDE

– JA16MACSJIS

– JA16SJIS

– JA16SJISTILDE

– JA16SJISYEN

– JA16VMS

– KO16KSCCS

– LA8ISO6937

– ZHS16MACCGB231280

– ZHT16BIG5

– ZHT16CCDC

– ZHT16HKSCS

– ZHT16HKSCS31

– ZHT16MSWIN950

You can use the Oracle Locale Builder utility to check which character codes are affected.

> **✎ See Also:**
>
> *Oracle Database Globalization Support Guide* for more information about the Oracle Locale Builder utility

If you suspect that a unique or primary key constraint might be affected by one of the above two issues, you can verify if you indeed have a problem by attempting to create an appropriate unique functional index. For example, if you have a unique or primary key constraint on character columns `tab1.col1` and `tab1.col2` and a numeric column `tab1.col3`, attempt to create the following index:

```
CREATE UNIQUE INDEX i_test
  ON tab1(SYS_OP_CSCONV(col1,'AL32UTF8','<assumed character set of col1>'),
          SYS_OP_CSCONV(col2,'AL32UTF8','<assumed character set of col2>'),
          col3)
  TABLESPACE ...
```

Substitute `'UTF8'` for `'AL32UTF8'`, if this is the actual target character set. The third parameter to `SYS_OP_CSCONV` may be omitted, if the assumed character set of a column is the same as the database character set (default). If the statement fails reporting "ORA-01452: cannot CREATE UNIQUE INDEX; duplicate keys found", there is a uniqueness problem in the column.

The `CREATE INDEX` statement above assumes that you do not plan any scheduled lengthening of the columns and the columns are not defined using character length

semantics. If you plan to extend `col1` to *n1* bytes and `col2` to *n2* bytes, use the following statement:

```
CREATE UNIQUE INDEX i_test
  ON tab1(SYS_OP_CSCONV(SUBSTRB(RPAD(col1,n1,' '),1,n1),
                            'AL32UTF8','<assumed character set of col1>'),
          SYS_OP_CSCONV(SUBSTRB(RPAD(col2,n2,' '),1,n2),
                            'AL32UTF8','<assumed character set of col2>'),
          col3)
  TABLESPACE ...;
```

Substitute `'UTF8'` for `'AL32UTF8'`, if the target character set is UTF8.

If `col1` and `col2` are defined using character length semantics and their character lengths are, respectively, *n1* and *n2*, use the following statement:

```
CREATE UNIQUE INDEX i_test
  ON tab1(SYS_OP_CSCONV(SUBSTRB(RPAD(col1,4*n1,' '),1,4*n1),
                            'AL32UTF8','<assumed character set of col1>'),
          SYS_OP_CSCONV(SUBSTRB(RPAD(col2,4*n2,' '),1,4*n2),
                            'AL32UTF8','<assumed character set of col2>'),
          col3)
  TABLESPACE ...;
```

Substitute `'UTF8'` for `'AL32UTF8'` and 3* for 4*, if the target character set is UTF8.

5.6.2 Convertibility Issues: Index Size

The maximum size of an index key in an Oracle database index cannot exceed the data block size in the tablespace of the index minus around 25% overhead.

The maximum is equal to the sum of maximum byte lengths of all key columns plus rowid length plus the number of length bytes. If changes are made to a key column during conversion, its index key size may increase as follows:

- For databases in which the `COMPATIBLE` initialization parameter is set to a value of `20.0.0` or higher, the index key size is calculated using the maximum lengths of column data stored in the key columns. An increase in column data size may cause the index key to exceed its allowed maximum length.

- For all other databases, the index key size is calculated using the maximum lengths of key columns. If the maximum byte length of a character column belonging to an index key increases during database conversion, either because:

  – it is being recalculated from the column character length for the new database character set

  or

  – a lengthening cleansing action is defined on the column

  The maximum byte length may cause the index key to exceed its allowed maximum length.

Starting with DMU 23.1, you can view and resolve potential problems with index key sizes prior to converting a database. When scanning the database, you can instruct the DMU to perform an index check and generate an index check report. This report lists all indexes whose key sizes will exceed the maximum supported size as a result of column modifications required for the conversion or an increased post-conversion column data size. You can then fix the problematic indexes before proceeding with the database conversion. See "Database Scan Report: Index Check Report" for more details.

5.6.3 Convertibility Issues: Partition Range Integrity

The DMU does not support converting a database in which any partition bounds require conversion. Therefore, you may want to migrate such a database using Data Pump utilities.

When preparing for the migration process, consider the following potential issues with partition integrity. Oracle Database distributes table rows among range partitions by comparing values of the partitioning key columns with partition bounds. This comparison uses binary sort order, that is, the byte representations of values, as stored on disk, are compared byte by byte. With character columns, the representation depends on the database character set and may change in conversion to Unicode. If the binary representation of partition bounds changes, the partitions might get reordered and the distribution of rows among partitions might change significantly.

You should analyze all range partitioned tables with convertible data in partitioning columns. If required, adjust their partition definitions, so that rows are still distributed according to your expectations after the database migration to Unicode.

Partition range bounds and partition list values may also suffer from the uniqueness issue described in "Convertibility Issues: Uniqueness Validation". If such an issue is encountered for a partitioned table, the table cannot be imported successfully without partition definitions being adjusted accordingly.

5.6.4 Convertibility Issues: Objects in the Recycle Bin

The DMU scans character columns in tables that have been dropped and are in the recycle bin like normal tables.

Scan results are included in the database scan report. Unlike the normal application tables, the dropped tables in recycle bin are not allowed to contain data that requires conversion. You cannot start the conversion step until tables with convertible data are removed from the recycle bin. No cleansing actions are supported on dropped tables.

5.6.5 Convertibility Issues: PL/SQL Local Identifiers Greater Than 30 Bytes

The DMU does not support converting names of PL/SQL stored modules, that is, stored procedures, stored functions and packages.

However, the DMU does automatically convert PL/SQL source code and view definitions including non-ASCII characters in:

- names of procedures, functions, and types defined in packages
- local identifiers, such as variable names and type names
- character literals
- comments

In the conversion step, the DMU fetches the relevant `CREATE OR REPLACE PACKAGE|PACKAGE BODY|PROCEDURE|FUNCTION|TYPE|TYPE BODY|VIEW` statements from the data dictionary, converts them to the target character set and, after the database character set has been changed to UTF8 or AL32UTF8, executes them.

The DMU does not check if any converted identifier exceeds its length constraint (usually 30 bytes for Oracle databases earlier to Oracle Database 12c Release 2 (12.2)). If an identifier becomes longer than allowed after conversion to Unicode, the resulting PL/SQL module will be

created but its compilation will fail, usually reporting "PLS-00114: identifier '*<identifier>*' too long". The status of the module will be "invalid".

You should verify the status of all PL/SQL stored modules after the database conversion. If any module is in the invalid state because of an identifier being too long, you must manually shorten the identifier – in its definition and in all places where it is referenced.

> **Note:**
>
> Starting with Oracle Database 12c Release 2 (12.2), the maximum limit for an identifier is 128 bytes. For the earlier Oracle Database versions, the maximum limit for an identifier is 30 bytes.

5.7 Adapting Applications for Unicode Migration

The migration of a database to Unicode always impacts applications connecting to this database.

The scale of the impact depends on multiple factors, such as the following:

- Do you want the applications to process new languages, which the database will now be able to store?

  Support for new languages usually entails the need to adapt applications to process Unicode data. Applications that have been programmed to process only single-byte character sets will need significant changes to be able to process the full Unicode character repertoire. On the other hand, applications that will work with the same limited number of characters as before may require only minimal changes, taking advantage of Oracle client/server character set conversion.

- Will languages with complex scripts require GUI support from the applications?

  Complex script-rendering capabilities are necessary to display and accept text written in complex scripts, such as Arabic or Indian. Complex rendering includes, among other requirements, combining adjacent characters, where some fragments read from right-to-left, while some are left-to-right, as well as changing character shape depending on the character's position in a word.

- What technologies are the applications built with?

  Depending on the development framework in which the applications have been developed, modifying them to accept new languages may be relatively easy or very difficult. Fortunately, most modern environments, such as Java or Microsoft Windows, offer built-in or installable support for complex script rendering and Unicode processing. Applications can take advantage of this support, which simplifies the adaptation process, but does not eliminate it.

- Are the applications developed in-house or by a third-party vendor?

  Obviously, you can adapt only those applications of which you have the source code. Applications developed by vendors must be adapted by those vendors. If this turns out to be impossible, you may have to replace your applications with another software solution.

In addition to these considerations, requirements to make migration-related changes to applications can result from:

- Table structure changes coming from cleansing actions, columns being lengthened or migrated to another data type.

- Changed characteristics of the database character set, such as higher maximum byte width of a character.

- Additional requirements of the new languages to be processed. This might include new sorting rules, new date or number formatting rules, non-Gregorian calendar support, and so on.

- Changed binary sort order of strings in Unicode compared to the old legacy character set.

Details of adapting applications for new languages exceed the scope of this guide, but the following sections describe the minimal changes that may be required to continue running existing applications with a migrated database.

5.7.1 Running Legacy Applications Unchanged

You might want to run some of your existing applications unchanged after migration of their back-end database to Unicode.

For example, you might not have access to the source code of the applications to adapt them for Unicode or modifying the applications may be economically unjustified.

The main requirement for running an application unchanged after migration of its back-end database to Unicode is that all Unicode characters that the application may encounter also exist in a legacy character set for which the application was originally written. That is, if the application was written, for example, to process only the ISO 8859-1 standard character set (WE8ISO8859P1), then only the corresponding Unicode characters U+0000 - U+007F and U+00A0 - U+00FF are allowed in the subset of database contents accessed by the application.

If this requirement is fulfilled, the client character set for the application, as specified in the NLS_LANG environment setting, shall be set to the above legacy character set (in practice, it usually means that NLS_LANG remains the sam before and after the migration) and the client/server communication protocol will take care of converting character data between the legacy encoding used by the application and the Unicode encoding used by the database.

If the only characters processed by an application are standard US7ASCII characters, the application does not require any modification at all, because UTF8 and AL32UTF8 binary codes of all ASCII characters are identical to US7ASCII. Therefore, the processed bytes do not differ before and after the database character set migration.

Binary codes of characters outside of the ASCII range - for example, accented Latin letters - remain the same on the client side but change on the database side after the database character set change. The most important difference is usually the number of bytes in the character codes. While accented Latin letters, and also Cyrillic or Greek letter, occupy one byte each in single-byte legacy character sets, such as WE8MSWIN1252 or CL8MSWIN1251, they occupy two bytes in UTF8 and AL32UTF8. Certain special characters, such as the Euro currency symbol, occupy three bytes. Therefore, any affected columns, PL/SQL variables and user-defined data type attributes on the database side whose lengths are expressed in bytes need to be adjusted to accommodate additional bytes that are added in the client/server character set conversion when data comes from the application to the database.

If an application does not rely on the database to control data lengths, controlling the length limits itself, and if all data processed by the application is entered into the database only through the application, byte lengths of affected columns, PL/SQL variables, and user-defined data types attributes may be adjusted by increasing them appropriately, usually by multiplying by three. If an application relies on the database to control data lengths, by handling the returned errors (such as ORA-12899), or if data for the application may come from external sources, the recommended way of adjusting lengths in SQL and PL/SQL is to keep the original absolute length number and to change the length semantics from bytes to characters. That is,

a `VARCHAR2(10 [BYTE])` column or PL/SQL variable should become a `VARCHAR2(10 CHAR)` column or variable.

As the above length issues obviously affect the existing database contents as well, the column and attribute lengths usually have to be increased already as part of the cleansing step of the database character set migration process. The PL/SQL variables must be adjusted independently, unless their data type is expressed using the `%TYPE` attribute.

A significant problem exists because of absolute length limits of basic character data types. A `VARCHAR2` value stored in a database cannot exceed 4000 bytes and a `CHAR` value cannot exceed 200 types. Therefore, you can expand `VARCHAR2` columns and attributes only up to 1333 (in UTF8) or 1000 (in AL32UTF8) characters to have a guarantee that they can really store that number of random character codes without hitting the data type limit. If longer values are already commonly processed by an application, it may be impossible to continue running the application without modifying it, for example, to use the `CLOB` data type instead of `VARCHAR2`.

Independently of the above length issues, some applications may rely on a specific binary sorting order of queried character values coming from the database. Characters of the US7ASCII and WE8ISO8859P1 character sets keep the same order in UTF8 and AL32UTF8 but characters from other character sets do not. You may need to create custom linguistic definitions using the Oracle Local Builder utility to simulate the binary order of a legacy character set in an UTF8 or AL32UTF8 database. In general, Oracle recommends that you modify the application to not rely on a specific binary sort order instead of creating custom linguistic definitions, which increase complexity and cost of database administration and may impact query performance.

In addition to these changes, the database side of the application code might require further changes required by the new Unicode character encoding, as described in the next section.

> ✎ **See Also:**
>
> *Oracle Database Globalization Support Guide* for more information about creating custom linguistic definitions

5.7.2 Changes to SQL and PL/SQL Code

Because SQL and PL/SQL code runs inside the database, its processing character set always changes after a database migration to Unicode. The processing character set is the one in which the processed data is encoded, which in this case is the database character set.

This is different from the client-side application code, which can retain its processing character set after migration if `NLS_LANG` setting is left unchanged.

Most PL/SQL statements, expressions, functions, and procedures work independently of the database character set and require no adaptation. However, there are still of number of functions that provide different results for different processing character sets. SQL and PL/SQL code containing the following functions must be reviewed and modified as required to account for the migration to Unicode:

* Functions depending on specific binary character codes: `CHR`, `ASCII`, and `DUMP`

* Functions depending on character code widths in bytes: `LENGTHB`, `VSIZE`

* Functions working with byte offsets: `INSTRB`, `SUBSTRB`

* Character set conversion functions: `CONVERT`

- String-to-binary casting: `UTL_RAW.CAST_TO_VARCHAR2`, `UTL_RAW.CAST_TO_RAW`, `UTL_I18N.STRING_TO_RAW`, `UTL_I18N.RAW_TO_CHAR`

Also, the standard package `UTL_FILE` allows character data to be stored in external files in the database character set. Consumers of those files may need to be adapted to deal with the new file encoding.

The SQL and PL/SQL expressions must be reviewed in view definitions, PL/SQL stored modules, user-defined data type methods, triggers, check constraints, but also event rule conditions (see `DBMS_RULE_ADM`), row-level security (RLS/VPD) policies (see `DBMS_RLS`), fine-grained auditing policies (see `DBMS_FGA`) and any other auxiliary database objects definitions that may reference SQL or PL/SQL expressions.

> ⚠️ **Caution:**
>
> While uncommon, row-level security or fine-grained auditing policies may be defined using SQL expressions that are sensitive to character encoding. Ensure that you review all such policies to verify that protection of sensitive data is not compromised because of the database character set change.

5.8 Exporting and Importing Migration Profile

Before performing Unicode migration on a production database instance, you may want to perform multiple iterations of trial migrations on a test database instance cloned from your production database instance.

Once the trial migration is successful, you can then apply the same procedure and related settings for migrating the production database instance. DMU provides the functionality of exporting and importing of migration profiles, so as to enable the reuse of database migration settings across multiple trial migrations and for the production database migration.

Migration profile includes general information about the database, user-specified migration settings for scanning and conversion operations, and any scheduled cleansing actions.

The following migration profile information can be exported and imported using DMU:

- Database General Information
  - Current Database Character Set
  - Assumed Database Character Set
  - Target Database Character Set
- Database Scan Parameters
  - Number of Scanning Processes
  - Scan Buffer Size
- Database Conversion Parameters
  - Degree of Parallelism
  - Number of Converting Processes
  - Enable Row Movement for Partitioned Tables
  - Consider CTAS with Row Movement Disabled

- – Consider CTAS with User-named LOB Segments
- Database Post Conversion Settings
  - – Handling of Read-Only Materialized Views
  - – Handling of Updatable Materialized Views
  - – Error Handling for Refreshing Materialized Views
  - – Handling of Dropped Domain Indexes
  - – Error Handling for Recreating Domain Indexes
  - – Error Handling for Rebuilding Other Indexes
- Table Scan Parameters (Set by User)
  - – Rowids to Collect
- Table Conversion Parameters (set by user)
  - – User-preferred Conversion Method
  - – Target Tablespace
  - – Preserve Position of LONG Column
- Column Scheduled Cleansing Actions (set by user)
  - – Define assumed column character set (example, WE8ISO8859P15)
  - – Change column semantics (example, BYTE to CHAR)
  - – Modify column size (example, VARCHAR2(10) to VARCHAR2(20))
  - – Migrate data type (example, VARCHAR2(4000) to CLOB)
  - – Exclude from Conversion
  - – Allow Conversion of Data with Issues
- Attribute Scheduled Cleansing Actions (set by user)
  - – Modify column size (example, VARCHAR2(10) to VARCHAR2(20))
  - – Change column semantics

> ✎ **See Also:**
>
> - Exporting a Migration Profile
> - Importing a Migration Profile

5.8.1 Exporting a Migration Profile

You can export the migration profile information for a database instance using the Export Migration Profile Wizard of DMU.

To export migration profile information for a database instance:

1. Select **Export Migration Profile** from the **Migration** menu or the context menu. The Welcome page of Export Migration Profile Wizard is displayed.

2. Click **Next**. The Output File page is displayed.

3. Enter the values for the following fields on the Output File page:

- **File Name**: Name of the file to store the migration profile information.

- **File Directory**: Absolute directory path of the file.

 You can either use the default values provided by DMU or enter any other values for these fields.

4. Click **Finish**. A progress bar is displayed when the export process is running. The Export Complete dialog box is displayed once the export process is complete.

5.8.2 Importing a Migration Profile

You import an existing migration profile when installing the DMU repository.

To import migration profile information for a database instance:

1. Select **Configure DMU Repository** from the **Migration** menu. The Select Task page of Repository Configuration Wizard is displayed.

2. If there is no DMU repository in the database, then select the following option on the Select Task page:

 ```
 Install the repository in migration mode by importing from a profile
 ```
 If DMU repository already exists in the database, then select the following option on the Select Task page:

 ```
 Re-install the repository in migration mode by importing from a profile
 ```

3. Click **Next**. The Task Details page is displayed.

4. Enter the information for the following fields on the Task Details page:

- **File Path**: Enter the name and directory path of the file containing migration profile information that you want to import into the current database instance.

- **Import environment-specific parameter settings**: Select this check box, if you want to import environment-specific parameter settings, such as, number of scanning processes, scan buffer size, and degree of parallelism into the current database instance. If you do not select this check box, then DMU assigns default values for these parameters based on the current system environment.

- **Archive the existing repository**: This check box is displayed if DMU repository already exists. Select this check box if you want to take the backup of the existing repository before re-installing the new repository.

 > **✎ Note:**
 >
 > The archived repository can be used for diagnostic purpose only. It cannot be restored and used for performing any DMU tasks.

5. Click **Next**. A progress bar containing the message `Checking the DMU import file...` is displayed.

 If the check for the file to be imported fails due to any of the following reasons, then DMU displays Error dialog box with appropriate error message:

- The DMU import file is incompatible with the current DMU version.

- The database character set information is inconsistent between the DMU import file and the current database.

ORACLE®

- • The database schema information is inconsistent between the DMU import file and the current database.

For any of the above errors, click **OK** on the Error dialog box. The file import process stops and the Task Details page is displayed.

If the check for the file to be imported shows a Warnings dialog box due to any of the following mentioned differences between the database schema information present in the import file and that of the current database instance, then you may choose to either ignore these warnings and continue with the import process by clicking **Yes** button, or stop the import process by clicking **No** button on the Warnings dialog box.

- • The database schema information is inconsistent between the DMU import file and the current database.

- • The database schemas in the import file do not exist in the current database.

- • The database schemas in the import file do not have the same IDs as schemas in the current database.

6. When the check for the file to be imported is successful, or in case of warning messages, if you click the Yes button on the Warning dialog box, the Select Tablespace page is displayed.

7. Select the tablespace in which you want to install the DMU repository and click **Finish**. DMU displays the process of archiving and installing the repository using a progress bar.

> **Note:**
>
> You can terminate the import process by clicking the **Abort** button on the progress bar.

8. After the import process is successful, the Import Complete dialog box containing the following information is displayed:

```
The migration profile has been imported successfully.
```

5.9 Near-Zero Downtime Database Migration to Unicode

Oracle provides a solution to migrate Oracle Database to Unicode in near-zero downtime.

Oracle provides the following software tools:

- • DMU for doing database scanning, cleansing, and character set conversion.

- • Oracle Recovery Manager (RMAN) for creating the target database by cloning the source database using initial data load.

- • Oracle GoldenGate for replicating incremental changes on the source database to the target Unicode database.

> **Note:**
>
> To implement the near-zero downtime database migration for Unicode, you must have the following software versions:
>
> • Oracle Database version 11g or later
>
> • Oracle GoldenGate version 12.1.2.1.0 or later
>
> • Oracle DMU version 2.1 or later

At a high-level, the near-zero downtime database migration procedure involves the following steps:

1. Preparing the source database to be migrated by scanning the data and cleansing all the reported convertibility issues in it using DMU.

2. Generating the parameter files and script files required by Oracle GoldenGate based on any scheduled data cleansing actions performed on the source database using DMU.

3. Setting up Oracle GoldenGate to capture incremental data changes in the source database.

4. Setting up a target database by creating a clone of the source database.

5. Converting the target database to Unicode using DMU.

6. Replicating the incremental data changes in the source database to the target database using Oracle GoldenGate.

7. Switching over from the source non-Unicode database to the target Unicode database with the minimal system interruption, once the target database is ready and is fully synced-up with the source database.

In order to correctly replicate the incremental data changes in the source non-Unicode database to the target Unicode database, Oracle GoldenGate requires certain parameter files containing the source database information about schemas, tables, columns, and any scheduled data cleansing actions that were performed using DMU to address the convertibility issues to Unicode, such as, assumed character sets, scheduled column modifications, and so on. As DMU repository already contains this information, DMU provides the functionality of exporting this information in the form of the following parameter files and script files (called as *Obey* scripts):

• **Manager parameter file (mgr.prm)**: This file is used by the GoldenGate Manager process mgr, which is responsible for managing other GoldenGate processes.

• **Defgen parameter file (defgen.prm)**: This file is used by the GoldenGate utility defgen, which creates a file with data definitions for source or target database tables. The data definition information is required when the source and target database tables have different definitions, such as, when DMU scheduled cleansing actions are defined to address convertibility issues to Unicode.

• **Extract parameter file (ext.prm)**: This file is used by the GoldenGate process Extract, which captures the transactional data from the source database.

• **Replicat parameter file (rep.prm)**: This file is used by the GoldenGate process Replicat, which reads the data extracted by the Extract process from the source database and populates this data in the target database.

• **Obey start scripts files (start_extract and start_replicat)**: These files contain the Obey scripts for starting the GoldenGate processes - Extract and Replicat.

- **Obey stop scripts files (stop_extract and stop_replicat)**: These files contain the Obey scripts for stopping the GoldenGate processes - Extract and Replicat.
- **Defgen script files (defgen.bat and defgen.sh)**: These files contain the script for running the GoldenGate process defgen for Windows and UNIX platforms.

> **Note:**
>
> - If Extract and Replicat processes are running on different host systems, then DMU generates two sets of above parameter files - one set of files for the Extract process and another set of files for the Replicat process.
>
> - For replicating pluggable databases (PDBs) in Oracle Database 12c, the Oracle GoldenGate administrator needs to have additional privileges. Run the following command to provide additional privileges to the Oracle GoldenGate administrator:
>
>   ```
>   SQL> EXEC dbms_goldengate_auth.grant_admin_privilege(
>               'C##OGG_OWNER', CONTAINER=>'ALL');
>   ```

The following figure shows the detailed steps for migrating Oracle Database to Unicode in a near-zero downtime. In this diagram, OGG stands for Oracle GoldenGate.
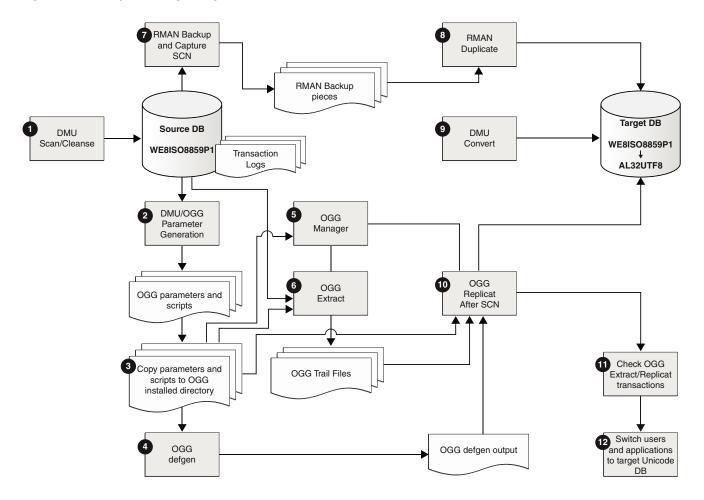
Figure 5-1   Steps for Migrating Oracle Database to Unicode in Near-Zero Downtime

1. Use DMU to scan and cleanse the source database until all the reported convertibility issues are resolved.

2. Use DMU for generating parameter files and script files required by Oracle GoldenGate. The following are the steps to generate Oracle GoldenGate parameter files and script files from DMU:

    a. In DMU, select **Generate GoldenGate Parameters** from the Migration menu or the context menu. The Welcome page of Generate GoldenGate Parameters Wizard is displayed.

    b. Click **Next**. The Extract Parameters page is displayed.

    c. Enter the following information on the Extract Parameters page that is required for the GoldenGate Extract process:

        •   Userid alias and domain name of the database user stored in the Oracle GoldenGate credential store.

        •   Name of the GoldenGate database schema.

    d. Click **Next**. The Replicat Parameters page is displayed.

    e. Enter the following information on the Replicat Parameters page that is required for the GoldenGate *Replicat* process:

- Userid alias and domain name of the database user stored in the Oracle GoldenGate credential store. You can either specify to use the same user information as that of the Extract process or enter different user information.

  - Host name where the Replicat process will be running. You can either specify to use the same host name as that of the Extract process or enter different host name.

  - Oracle SID and Oracle Home of the target database.

f. Click **Next**. The Output Files page is displayed.

g. On the Output Files page, enter the absolute directory path for storing the output parameter files. You can either use the default directory path provided by DMU or specify any other directory path.

h. Click **Finish**. A progress bar is displayed when the parameter files and script files are being generated.

i. If the parameter files and script files are generated successfully, the Generation Complete dialog box containing the information about all the files that are generated is displayed.

   If the parameter files generation process finishes with warnings, then those warnings are displayed on the Generation Complete dialog box.

   If IOException occurs during the parameter files generation process, then the Error dialog box is displayed with the IOException details, and the parameter files generation process is terminated.

3. Copy the DMU generated parameter files and script files that were generated in step 2 to the Oracle GoldenGate installed directory.

   The following table contains the information about the source directory and the destination directory for copying the parameter files and script files, when the Extract and Replicat processes are running on the same system. This document refers to this configuration as local Replicat configuration.

   In this table, DMU_DIR is the directory on the DMU system where parameter files and script files are stored during execution of step 2. OGG_HOME is the Oracle GoldenGate home directory on the system where Oracle GoldenGate software is installed.

Table 5-3   Source directory and the destination directory for copying parameter files and script files from DMU system to GoldenGate system for local Replicat process

Parameter/Script	Source Directory	Destination Directory
Manager parameter	*DMU_DIR*/dirprm/mgr.prm	*OGG_HOME*/dirprm/mgr.prm
Extract parameter	*DMU_DIR*/dirprm/ext.prm	*OGG_HOME*/dirprm/ext.prm
Replicat parameter	*DMU_DIR*/dirprm/rep.prm	*OGG_HOME*/dirprm/rep.prm
Defgen parameter	*DMU_DIR*/dirprm/defgen.prm	*OGG_HOME*/dirprm/defgen.prm
Defgen batch file	*DMU_DIR*/defgen.bat	*OGG_HOME*/defgen.bat
Defgen shell script	*DMU_DIR*/defgen.sh	*OGG_HOME*/defgen.sh
Obey start Extract script	*DMU_DIR*/start_extract	*OGG_HOME*/start_extract
Obey start Replicat script	*DMU_DIR*/start_replicat	*OGG_HOME*/start_replicat
Obey stop Extract script	*DMU_DIR*/stop_extract	*OGG_HOME*/stop_extract

ORACLE

Table 5-3   (Cont.) Source directory and the destination directory for copying parameter files and script files from DMU system to GoldenGate system for local Replicat process

Parameter/Script	Source Directory	Destination Directory
Obey stop Replicat script	*DMU_DIR*/stop_replicat	*OGG_HOME*/stop_replicat

The following table contains the information about the source directory and the destination directory for copying the parameter files and script files when the Extract and Replicat processes are running on different systems. This document refers to this configuration as remote Replicat configuration.

In this table, DMU_DIR is the directory on the DMU system where parameter files and script files are stored during step 2. OGG_HOME_EXT is the Oracle GoldenGate home directory on the Extract process host system. OGG_HOME_REP is the Oracle GoldenGate home directory on the Replicat process host system.

Table 5-4   Source directory and the destination directory for copying parameter files and script files from DMU system to GoldenGate system for remote Replicat process

Parameter/Script	Source Directory	Destination Directory
Manager parameter (Extract)	*DMU_DIR*/extract/dirprm/ mgr.prm	*OGG_HOME_EXT*/dirprm/ mgr.prm
Manager parameter (Replicat)	*DMU_DIR*/replicat/dirprm/ mgr.prm	*OGG_HOME_REP*/dirprm/ mgr.prm
Extract parameter	*DMU_DIR*/extract/dirprm/ ext.prm	*OGG_HOME_EXT*/dirprm/ ext.prm
Replicat parameter	*DMU_DIR*/ replicat/dirprm/ rep.prm	*OGG_HOME_REP*/dirprm/ rep.prm
Defgen parameter	*DMU_DIR*/extract/dirprm/ defgen.prm	*OGG_HOME_EXT*/dirprm/ defgen.prm
Defgen batch file	*DMU_DIR*/extract/ defgen.bat	*OGG_HOME_EXT*/defgen.bat
Defgen shell script	*DMU_DIR*/extract/defgen.sh	*OGG_HOME_EXT*/defgen.sh
Obey start Extract script	*DMU_DIR*/extract/ start_extract	*OGG_HOME_EXT*/ start_extract
Obey stop Extract script	*DMU_DIR*/extract/ stop_extract	*OGG_HOME_EXT*/stop_extract
Obey start Replicat script	*DMU_DIR*/replicat/ start_replicat	*OGG_HOME_REP*/ start_replicat
Obey stop Replicat script	*DMU_DIR*/replicat/ stop_replicat	*OGG_HOME_REP*/ stop_replicat

4. For the local Replicat configuration, on the Oracle GoldenGate system, execute the defgen script present in the OGG_HOME directory to generate source table definitions which were scheduled for cleansing by DMU. The defgen script generates the output file defgen.def in the OGG_HOME/dirdef directory.

For the remote Replicat configuration, on the Oracle GoldenGate system, execute the defgen script present in the OGG_HOME_EXT directory on the Extract process host system to generate source table definitions which were scheduled for cleansing by DMU.

The defgen script generates the output file defgen.def in the OGG_HOME_EXT/dirdef directory. Copy the output file defgen.def to the OGG_HOME_REP/dirdef directory on the Replicat process host system.

5. For the local Replicat configuration, start the Oracle GoldenGate Manager process in the Oracle GoldenGate Command Interface (GGSCI):

```
GGSCI>start manager
```

For the remote Replicat configuration, start the Oracle GoldenGate Manager processes on both, the Extract and the Replicat host systems.

6. For local and remote Replicat configurations, start the Oracle GoldenGate Extract process in GGSCI:

```
GGSCI>obey start_extract
```

7. Use RMAN to take backup of the source database and capture the checkpoint SCN value to be used by the Oracle GoldenGate Replicat process.

To take backup of the database, run the command:

```
RMAN>backup database plus archivelog;
```

To capture the checkpoint System Change Number (SCN) value, run the command:

```
RMAN>list backup;
```

or run the command:

```
RMAN>restore database preview summary;
```

8. Use RMAN to create the target database by cloning the source database.

9. Use DMU to convert the target database character set to Unicode, for example from WE8ISO8859P1 to AL32UTF8. You should be able to start the DMU conversion on the target database immediately, as all the convertibility issues on the source database have been cleansed in Step 1 and the corresponding migration-related metadata information in the DMU repository has been cloned from the source database.

> **✎ Note:**
>
> Restart the target database after the DMU conversion is complete.

10. Edit start_replicat Obey script by replacing the SCN value in it with the SCN value captured in step 7 and start the Oracle GoldenGate Replicat process.

In the following start_replicat Obey script sample, the SCN value of 123456 is entered:

```
START REPLICAT rep, AFTERCSN 123456
```

For the local Replicat configuration, start the Oracle GoldenGate Replicat process in the Oracle GoldenGate Command Interface (GGSCI):

```
GGSCI>obey start_replicat
```

For the remote Replicat configuration, start the Oracle GoldenGate Replicat process on the Replicat host system.

11. Ensure that all the transactions have been extracted and replicated by Oracle GoldenGate:

    a. Check that there are no active transactions in the source and the target databases:

    ```
    SQL> select count(*) from v$transaction;
    COUNT(*)
    ```

```
--------
       0
```

b. Query the current SCN of the source database:

```
SQL> select current_scn from v$database;
CURRENT_SCN
-----------
    1439706
```

c. Query the status of the Extract process and verify that it has read past the last SCN
    value (for example, the SCN value of 1439706 mentioned in the above step):

```
GGSCI> send extract ext, status
Sending STATUS request to EXTRACT EXT ...
EXTRACT EXT (PID 6312)
  Current status: Recovery complete: Processing data
  Current read position:
  Redo thread #: 1
  Sequence #: 161
  RBA: 10483752
  Timestamp: 2015-01-09 13:56:14.000000
  SCN: 0.1439718
  Current write position:
  Sequence #: 0
  RBA: 1534
  Timestamp: 2015-01-09 13:56:13.756998
  Extract Trail: ./dirdat/dm
```

d. Query the status of the Replicat process and verify that it shows At EOF status:

```
GGSCI> send replicat rep, status
Sending STATUS request to REPLICAT REP ...
  Current status: At EOF
  Sequence #: 0
  RBA: 1534
  0 records in current transaction
```

12. Switch users and applications to the target Unicode database.

5.10 Repairing Database Character Set Metadata

You database may be in what is commonly called a pass-through configuration. The client
character set may be defined, usually through the NLS_LANG client setting, to be equal to the
database character set. Consequently, the character data in your database could differ from
the declared database character set.

In this scenario, the recommended solution is to migrate your database to Unicode by using
the DMU assumed database character set feature to indicate the actual character set for the
data. In case migrating to Unicode is not immediately feasible due to business or technical
constraints, it would be desirable to at least correct the database character set declaration to
match with the database contents.

With Database Migration Assistant for Unicode, you can repair the database character set
metadata in such cases using the CSREPAIR script. The CSREPAIR script works in conjunction

with the DMU client and accesses the DMU repository. It can be used to change the database character set declaration to the real character set of the data only after the DMU has performed a full database scan by setting the Assumed Database Character Set property to the target character set and no invalid representation issues have been reported, which verifies that all existing data in the database is defined according to the assumed database character set. Note that CSREPAIR only changes the database character set declaration in the data dictionary metadata and does not convert any database data.

You can find the CSREPAIR script under the admin subdirectory of the DMU installation. The requirements when using the CSREPAIR script are:

1. You must first perform a successful full database scan in the DMU with the Assumed Database Character Set property set to the real character set of the data. In this case, the assumed database character set must be different from the current database character set or else nothing will be done. The CSREPAIR script will not proceed if the DMU reports the existence of data with invalid binary representation. It will, however, proceed if data that is changeless, convertible or exceeding length limits is reported in the scan results.

2. The target character set in the assumed database character set must be a binary superset of US7ASCII.

3. Only repairing from single-byte to single-byte character sets or multi-byte to multi-byte character sets is allowed as no conversion of CLOB data will be attempted.

4. If you set the assumed character set at the column level, then the value must be the same as the assumed database character set. Otherwise, CSREPAIR will not run.

5. You must have the SYSDBA privilege to run CSREPAIR. To run CSREPAIR on a pluggable database (PDB) in Oracle Database 12*c*, you must be either the SYS user or a common user with the SYSDBA privilege in both the local PDB and the CDB.

5.10.1 Example: Using CSREPAIR

A typical example is storing WE8MSWIN1252 data in a WE8ISO8859P1 database using the pass-through configuration.

To correct the database character set from WE8ISO8859P1 to WE8MSWIN1252:

1. Set up the DMU and connect to the target WE8ISO8859P1 database.

2. Open the Database Properties tab in the DMU.

3. Set the Assumed Database Character Set property to WE8MSWIN1252.

4. Use the DMU to perform a full database scan.

5. Open the Database Scan Report and verify there is no data reported under the Invalid Representation category.

6. Exit from the DMU client.

7. Start the SQL*Plus utility and connect as a user with the SYSDBA privilege.

8. Run the CSREPAIR script:

   ```
   SQL> @@CSREPAIR.PLB
   ```

   Upon completion, you should get the message:

   The database character set has been successfully changed to WE8MSWIN1252. You must restart the database now.

9. Shut down and restart the database.

5.11 DMU Accessibility Information

It is Oracle's goal to make Oracle Products, Services, and supporting documentation accessible to the disabled community.

Oracle Database Migration Assistant for Unicode supports accessibility features.

For additional accessibility information for Oracle products, see the Oracle Accessibility Program page at: `http://www.oracle.com/accessibility/`.

5.11.1 Using a Screen Reader and Java Access Bridge with the DMU

For assistive technologies such as screen readers to work with Java-based applications and applets, the Windows-based computer must also have Sun's Java Access Bridge installed.

To make the best use of our accessibility features, Oracle Corporation recommends the following minimum configuration:

- Windows XP, Windows Vista
- Java 7 Update 6

> **✎ Note:**
>
> Java 7 Update 6 includes the Java Access Bridge. For more information, including how to enable the Java Access Bridge, see:
>
> http://docs.oracle.com/javase/7/docs/technotes/guides/access/index.html
>
> However, if you are using Java 7, but earlier to Java 7 Update 6, then you must manually install Java Access Bridge 2.0.2 after you install the screen reader (if it is not already installed). Download Java Access Bridge 2.0.2 for Windows having the file name `accessbridge-2_0_2-fcs-bin-b06.zip` from the website:
>
> http://www.oracle.com/technetwork/java/javase/tech/index-jsp-136191.html
>
> Refer to the Java Access Bridge documentation available from this website for more information about Java Access Bridge and its installation.

- JAWS 12.0.522
- Microsoft Internet Explorer 7.0 or later
- Mozilla Firefox 3.5 or later

Please refer to the following procedures to set up a screen reader and Java Access Bridge.

1. Install the screen reader, if it is not already installed.

   Refer to the documentation for your screen reader for more information about installation.

2. Install the DMU.

3. If you are using Java 7, but earlier to Java 7 Update 6, then you must manually install Java Access Bridge 2.0.2 as described below:

   a. Download Java Access Bridge version 2.0.2 for Windows. The file you need to download is `accessbridge-2_0_2-fcs-bin-b06.zip`. This file is available at:

      `http://www.oracle.com/technetwork/java/javase/tech/index-jsp-136191.html`

Refer to the Java Access Bridge documentation available from this web site for more information about Java Access Bridge installation.

b. Extract (unzip) the contents to a folder, for example, `accessbridge_home`.

c. Install Java Access Bridge by running `Install.exe` from the `<accessbridge_home>\installer` folder.

The installer first checks the JDK version for compatibility, then the Available Java virtual machines dialog displays.

d. Click **Search Disks**. Then select to search only the drive that contains the JDK version in the program files directory (if it exists).

The search process can take a long time on a large disk with many instances of JDK, or when searching multiple disks. However, unless you complete an exhaustive search of your disk, Access Bridge will not be optimally configured, and will not be correctly installed to all of the Java VMs on your system. After selecting the disk to search, click **Search**.

e. Confirm that you want to install the Java Access Bridge into each of the Java virtual machines displayed in the dialog, by clicking **Install in All**.

f. Click **OK** when you see the Installation Completed message.

g. Confirm that the following files have been installed in the `Winnt\System32` directory (o the equivalent Windows XP or Windows Vista directory), or copy them from `<accessbridge_home>\installerfiles` as they must be in the system path in order to work with the DMU:

- `JavaAccessBridge.dll`
- `JAWTAccessBridge.dll`
- `WindowsAccessBridge.dll`

Note that the system directory is required in the `PATH` system variable.

h. Confirm that the following files have been installed in the `jdk\jre\lib\ext` directory, or copy them from `<accessbridge_home>\installerfiles` directory:

- `access-bridget.jar`
- `jaccess.jar`

i. Confirm that the file `accessibility.properties` has been installed in the `jdk\jre\lib` directory, or copy it from `<accessbridge_home>\installerfiles` directory.

4. Start your screen reader.

5. Start the DMU.

The steps above assume you are running Windows and using a Windows-based screen reader. A console window that contains error information (if any) will open first and then the main DMU window will appear, once the DMU has started. Any messages that appear will not affect the DMU functionality.

6

Using the DMU to Cleanse Data

You can use the Database Migration Assistant for Unicode (DMU) to perform cleansing tasks. The cleansing actions can be immediate or scheduled.

In certain specific cases, two column properties allow you to ignore data issues as described in "Ignoring Convertibility Issues".

An immediate action is performed directly on the database contents and its effects are immediately visible. A scheduled cleansing action is registered in the DMU repository and executed during the conversion step of the migration process. Therefore, the effects of a scheduled action are virtual until the conversion. The DMU accounts for scheduled actions when scanning the database for issues by presenting adjusted scan results under the heading "including effects of scheduled cleansing" or "scheduled".

Scheduled cleansing actions are defined in the Schedule Column Modification and Schedule Attribute Modification dialog boxes. All other cleansing actions are immediate.

Note that if your application enforces specific data dependencies in your database where the data values stored in multiple table columns must be kept consistent, then you will need to manually maintain the data dependencies during the data cleansing process. This is because the DMU would have no knowledge of such user application schema dependencies.

6.1 Cleansing Data

The Cleansing Editor tab is the starting point of most cleansing actions on a table. You can open the tab by selecting **Cleansing Editor** from the context menu of a table node in the Navigator pane or in the Database Scan Report.

The Cleansing Editor is not available for the data dictionary tables. The Cleansing Editor tab is shown in Figure 6-1.

The Cleansing Editor tab contains a toolbar, an editor grid, and two buttons, Save and Revert. The editor grid displays contents of the table for which the Cleansing Editor was opened. This table is called the edited table later in this guide. The columns of the grid correspond to character data type columns of the edited table and rows of the grid correspond to rows of the table.

The editor grid works in one of two selection modes: if you click a single cell, the cell is selected and the grid switches to the cell selection mode. Only one cell can be selected in this mode. The selected cell is called the current cell. If you click a column heading (containing the name of the column), the grid switches to the column selection mode and the clicked column becomes a selected column. You can add more columns to the selection by holding down the Ctrl key and clicking further column headings. Figure 6-1 shows the cell in the `JOB_ID` column of the 10th displayed table row selected as the current cell.

Cells in the grid are highlighted with specific colors to indicate identified data issues. See "Cleansing Data: Color Highlighting" for more information. Figure 6-1 shows the cell in the `EMAIL` column of the 10th displayed table row on light coral background to indicate an invalid representation issue and the cell in the `LAST_NAME` column of the same row on yellow background to indicate a column length issue.

You can use the Cleansing Editor to filter out rows of the edited table so that only the rows relevant to your current cleansing activity are displayed in the editor grid. See "Database Scan Report: Filtering" for more information.

Figure 6-1   Cleansing Editor

The Cleansing Editor enables you to start various cleansing actions on contents of the edited table. If you double-click a cell or you select **Edit Data** from the context menu of the current cell, the Edit Data dialog box is opened, displaying the value stored in the cell. You can edit the value, for example shortening it or removing illegal character codes, as described in "Editing Data". Edited values are not stored in the database until you click **Save** at the bottom of the Cleansing Editor tab. A value that has been edited is displayed using italic font face until it is saved permanently in the database. Figure 6-1 shows the cell in the FIRST_NAME column of the 10th displayed table row as modified and not yet saved. You can click **Revert** to recall all edit changes and revert cell values to current values from the database.

If you right-click a cell in the cell selection mode (which also makes the cell current) or you right-click a selected column in the column selection mode, a context menu appears that contains Modify Columns and Schedule Column Modification menu items. If the current cell or the selected columns correspond to attributes of a user-defined data type (ADT), the context menu contains Modify Attribute and Schedule Attribute Modification instead. If multiple columns are selected and they are not all of the same data type, none of the prior menu items is present in the context menu. Clicking one of the four menu items opens the corresponding dialog box described in "Modifying Columns", "Scheduling Column Modification", "Modifying Attributes", or "Scheduling Attribute Modification". These dialog boxes allow you to modify table structure to cleanse convertibility issues in one or more columns. The dialog boxes are not available for columns and attributes of the CLOB data type.

Restriction: If a column is used in the defining expression of a virtual column, then the Modify Columns and Schedule Column Modification functionalities are not available for that column, and the corresponding context menu items are not shown.

If you click the **Modify Column** or **Schedule Column Modification** menu item, the DMU displays a warning if it detects that the column to be modified belongs to one of the schemas comprising an Oracle e-Business Suite installation. Altering the structure of Oracle e-Business Suite tables is generally not supported. You can accept the warning and continue with modifications, if and only if the affected table is a custom table created by you or you have been advised to modify the table by Oracle Support.

Before modifying the structure of any table installed along with an Oracle or third-party product, ask the vendor for advice. Modification of such tables is usually not supported as it may cause the owning application to malfunction.

You can use the Cleansing Editor to change the character set that the DMU uses to interpret character codes contained in a column. See "Setting the Assumed Character Set".

6.1.1 Cleansing Data: Using the Toolbar

The Cleansing Editor toolbar enables you to rescan tables and columns, and various other cleansing operations.

The following items appear in the order shown in Figure 6-1:

- Rescan Table

  Opens the Scan Wizard (see "Scanning the Database with the Scan Wizard") with only the edited table marked for scanning. This allows the edited table to be easily rescanned.

- Rescan Selected Columns

  Opens the Scan Wizard with only the currently selected columns marked for scanning. This allows a set of edited table columns to be easily rescanned. If no columns are selected, that is, the editor grid is in the cell selection mode, the column containing the current cell is marked for rescanning.

- Refresh Data

  Reloads the grid with the most current data from the edited table.

- Select Columns to Show

  Displays a dialog box in which you can select the columns to include in the editor grid. To save screen space, you can deselect columns that have no issues themselves and that do not contain any data helpful in analysis of issues in other columns. You can also add a special read-only column to the grid that displays rowids of edited table rows. You can use the rowids to quickly locate relevant rows in other database tools, such as Oracle SQL Developer.

- Show Impact of Scheduled Cleansing

  Highlights the effects of scheduled cleansing actions using a different color.

  See "Cleansing Data: Color Highlighting".

- Use Collected Rowids to Filter Data

  Filters data using rowids collected during scanning.

  See "Filtering Data".

- Customize Filtering Condition

  Displays a dialog box where you can customize the filtering condition. See "Filtering Data".

- Filter drop-down list with the following choices:

  - All

- Requiring Conversion

- Exceeding Column Limits

- Exceeding Data Type Limit

- Invalid Binary Representation

If you select one of the options other than All, the Cleansing Editor applies a filter to display only rows with the corresponding convertibility issues. See "Filtering Data".

- Previous Cell with Issue

Selects the previous cell with a convertibility issue as the current cell. The cell is searched for first in the preceding columns of the same row, and then in the preceding rows, from the last column to the first.

- Next Cell with Issue

Selects the next cell with a convertibility issue as the current cell. The cell is searched for first in the following columns of the same row, and then in the following rows, from the first column to the last.

- Character Set drop-down list

Contains character sets that you can select as assumed column character sets. See "Setting the Assumed Character Set".

6.1.2 Cleansing Data: Color Highlighting

Cells that contain data with convertibility issues are highlighted by using colors different from the standard colors of other cells.
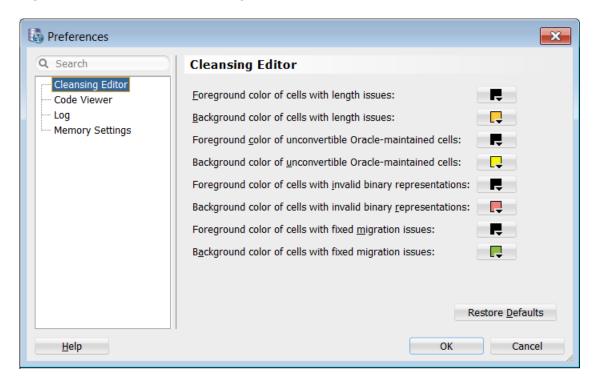
By default, cells with length issues, that is, exceeding column limit or exceeding data type limit after conversion, are displayed on yellow background. Cells with invalid binary representation issues, that is, containing character codes illegal in their declared (assumed) character set, are displayed on light coral background.

If an issue in a cell has been cleansed by a scheduled cleansing action, for example lengthening of the containing column, and the Show Impact of Scheduled Cleansing toolbar button is clicked, the cell background color changes to green, by default. An immediate cleansing action, such as editing a value, changing the assumed column character set, or modifying the column or attribute through a Modify Columns or Modify Attributes dialog box, permanently removes the issue, so that the previously affected cells become convertible and are no longer highlighted.

The preceding default colors can be customized on the Cleansing Editor page of the Preferences dialog box, as shown in Figure 6-2. You can open the Preferences dialog box by selecting **Preferences** from the Tools menu. The first two choices are for cells with length issues, the next two are for cells with unconvertible data dictionary cells, the next two are for cells with invalid binary representation, and the last two are for cells with fixed migration issues. In addition, you can restore default setting by clicking on **Restore Defaults**.

Figure 6-2   Preferences: Cleansing Editor

6.1.3 Filtering Data

The Cleansing Editor tab enables you to filter rows included in the editor grid on the types of the convertibility issues that they contain or on an arbitrary SQL condition.

6.1.3.1 Filtering on Convertibility Status

You can select the types of convertibility issues to include from the Filter drop-down list on the Cleansing Editor toolbar.

The choices are:

* All

  This option switches off filtering on convertibility issues and causes rows with any contents to be included.

* Requiring Conversion

  This option shows all rows that are not changeless, that is, one or more columns contain data that needs conversion and, optionally, has length expansion or invalid representation issues.

* Exceeding Column Limits

  This option shows all rows in which one or more column values exceed their column limit after conversion.

* Exceeding Data Type Limit

  This option shows all rows in which one or more column values exceed their data type limit after conversion.

- Invalid Binary Representation

  This option shows all rows in which one or more columns contain invalid character codes.

The DMU can identify rows with a given type of convertibility issues by doing either of the following:

- Looking at rowids collected in the DMU repository by a previous scanning task.

- Analyzing the column values as they are fetched from the database.

When only a small percentage of rows in the edited table has the type of convertibility issues to pass through the filter, the DMU might need to fetch and analyze many rows before it finds enough rows to fill the editor grid. This could negatively and significantly impact the speed of browsing through a large table when filtering is on. Locating rows by collected rowids is usually much faster. On the other hand, collected rowids correspond to the state of a table at the time of the last scan of the table. This state might already be stale because rows might have been added, updated, or deleted. Nevertheless, rescanning a large table with sparsely distributed issues periodically during a longer cleansing session is usually more convenient than trying to filter rows without using collected rowids.

You tell the DMU to use the collected rowids by clicking the **Use Scan Log to Filter Data** button on the toolbar of the Cleansing Editor. This button is enabled only if rowid collection was switched on during the last scan of the table. You can set the rowid collection level either on the Scanning subtab of the Table Properties tab (see "Table Properties: Scanning") or on the Scan Details page of the Scan Wizard (see "Scanning the Database with the Scan Wizard"). The setting on the Table Properties tab is persistent. The setting in the Scan Wizard is only for the current scan and overrides the value on the Table Properties tab. You can set the level to **All to Convert** or **With Issues**. The first option collects rowids for all rows that are not changeless. The second option collects rowids only for rows with issues, that is, exceeding column limit, exceeding data type limit, or having invalid binary representation. Use the first option if you plan to set the filter to **Requiring Conversion**. Use the second option if you plan to set the filter only to **Exceeding Column Limit**, **Exceeding Data Type Limit**, or **Invalid Binary Representation**.

Collecting rowids for all convertible rows might be very costly if many rows need conversion in the table, for example, because the table contains a CLOB column and the database character set is single-byte. Generally, use this option only for data dictionary tables to locate convertible data, which could prevent conversion. Rowids are collected per cell, so if a row contains multiple cells with data fitting the collection criteria, the rowid of the row is stored in multiple copies, each time with the ID of another column.
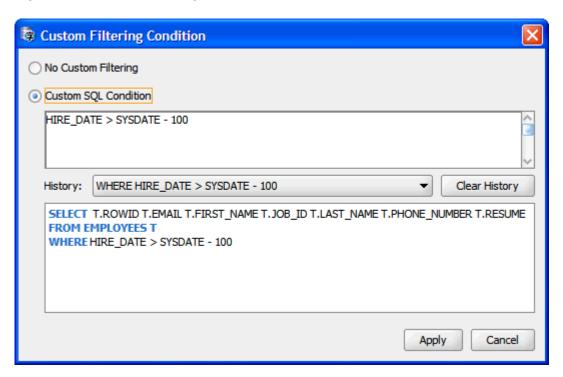
6.1.3.2 Filtering on SQL Condition

In addition to the filtering based on convertibility status of rows, you can specify an arbitrary SQL condition to be added to the query that the DMU sends to the database to retrieve data from the edited table.

Specify the condition in the Custom Filtering Condition dialog box, shown in Figure 6-3. Open the dialog box by clicking on **Customize Filtering Condition** on the toolbar of the Cleansing Editor tab. Select the **Custom SQL Condition** option in the dialog box and enter the desired condition. Click **Apply**. The condition is applied to the table to create a subset of rows to fetch and then the filtering on convertibility issues is applied to this subset. To remove the filtering condition, reopen the Custom Filtering Condition dialog box, select the No Custom Filtering option and click **Apply**.

Figure 6-3   Custom Filtering Condition

6.1.4 Setting the Assumed Character Set

The **assumed column character set** is the declared character set of data in a column that you can select in the DMU.

Applications sometimes use a pass-through configuration to store data directly in the client character set without any conversion and without knowing the database character. This situation can cause invalid representation issues. The DMU helps identify such client character sets, and then uses them as the conversion source when they differ from the database character set. In this way, incorrectly stored data is correctly converted to Unicode. The standard conversion from the database character set would corrupt the data and make it illegible.

The assumed column character set is also used when column data is interpreted for display in the Cleansing Editor, Edit Data dialog box, or Show Data dialog box. The displayed data looks correct to a human reader only if interpreted according to its real character set or a binary superset of the real character set. Otherwise, displayed values contain unexpected characters or replacement characters. (Replacement characters are usually represented by rectangular shapes of the size of an uppercase letter. This representation is platform-dependent.) If all data in the column looks correct, then you can verify the assumed column character set has been chosen correctly.

You might want to set the assumed character set of a column even if the column scan results report only convertible contents without issues. The reason being that data in a real character set that is different from the database character set might still consist of byte sequences that could be interpreted as valid in the database character set – even if this interpretation does not produce reasonable text. This is especially true for single-byte database character sets in which most byte values (0..255) have a valid character assigned in the character set definition. CL8MSWIN1251 is the prominent example, with only one byte value not assigned to any character.

You can identify the real character set of a column by doing either of the following:
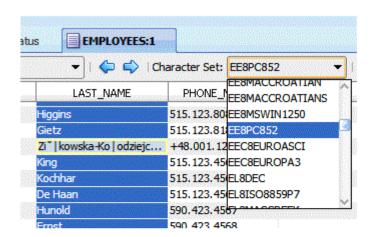
- Collecting the necessary information about the relevant application configuration, database clients, and client character sets (the preferred method)

- Selecting successive character sets as the assumed character set of the column and verifying the correctness (legibility) of displayed text
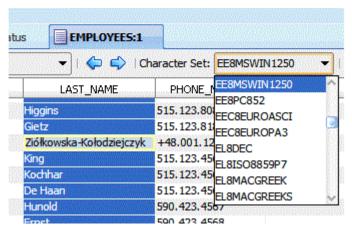
The second method might be very useful if you know the language of the data, but not the specific character set used to encode it. For example, you know that text in a column is in Japanese but you do not know if it is encoded in EUC-JP (JA16EUC) or Shift-JIS (JA16SJIS).

If you know the language of the data, then the DMU makes the character set detection easier by providing language selection option in the General subtab of the database properties panel described at "Database Properties: General". The languages selected in the database properties panel are used in the Cleansing Editor to list the possible character sets for columns.

To set the assumed character set of a column in the Cleansing Editor, select the column by clicking its heading and select the appropriate character set in the Character Set list on the toolbar, as shown in the following figure. The contents of the selected column are redisplayed according to the new interpretation. If you select multiple columns in the Cleansing Editor, you could set all of their assumed character sets to the same value at one time.

Figure 6-4   Setting Assumed Column Character Set

If you are not sure about the character set of the column or if the column data is illegible even after selecting a character set (the color highlighting of the data as described at "Cleansing Data: Color Highlighting" will tell you if some bytes in the column cannot be interpreted according to the selected character set), then you can select the column and click the **Detect Character Set** button present beside the Character Set list on the toolbar.

Figure 6-5   Detect Character Set Button

The DMU starts the process of detecting the possible character sets for the selected column by displaying a progress bar. After the character set detection process has completed, an appropriate message is displayed on the screen, and the Character Set list is rearranged. The rearranged list shows the most possible character sets (detected character sets) on top followed by a divider line, and the other valid character sets are listed below the divider line. You can then select the possible candidate character sets from the list to see how the column data is rendered in a particular character set. After verifying the correctness of the column contents, you can save the real character set of the column as its assumed character set.

Figure 6-6   Detected Character Sets for a Column

> **Note:**
>
> - The Detect Character Set button is disabled by default, and is enabled only when all the rows of a character type column (`CHAR`, `VARCHAR2`, `CLOB`, or `LONG`) are selected.
>
> - If a column contains a large amount of binary data, then the detected character set list shows `BINARY DATA` on top. If you select `BINARY DATA` from the list, then the column's data is interpreted according to the database character set when displaying it in the Cleansing Editor. As binary data is excluded from conversion, the Conversion tab is not shown in the properties panel for the column whose character set is set to `BINARY DATA`.
>
> - Each detected character set is assigned a *confidence ratio* specific to a column. The confidence ratio for each detected character set is shown as a tooltip in the Character Set list as a percentage value. Higher the confidence ratio, higher is the likelihood of a character set to be the actual character set of the selected column.

Click **Save** to save this information in the DMU repository.

6.2 Viewing Data

The Data Viewer is a read-only version of the Cleansing Editor tab.

The Data Viewer is described in "Cleansing Data: Using the Toolbar".

To view data:

- Select **Data Viewer** from the context menu of a table node under the Oracle-Maintained Schemas group node in the Navigator pane or in the Database Scan Report.

  The Data Viewer is not available for the standard application tables. It looks identical to the Cleansing Editor tab, shown in Figure 6-1, except that the button Show Impact of Scheduling Cleansing is disabled. There is no Save or Revert button. Also, the Edit Data, Modify Column, Schedule Column Modification, Modify Attribute, and Schedule Attribute Modification context menu items are not available. The Show Data menu item is available in place of Edit Data to open the Show Data dialog box, which is described in "Displaying Data". All viewing and filtering functionality of the Cleansing Editor is available in the Data Viewer. You can set the assumed character set of a column in the Data Viewer in order to analyze the real character set of column contents, but you cannot save this choice.

6.3 Editing Data

Open the Edit Data dialog box by double-clicking a table cell in the Cleansing Editor or by selecting **Edit Data** from the context menu of the current cell.

The dialog box shown in Figure 6-7. The value of the cell is shown in the main text area of the dialog box and can be edited by adding, changing, or deleting characters. If the value contains end-of-line characters (LF=0x0A), it will be displayed in multiple lines. The Wrap Lines check box determines how lines longer than the width of the text area are displayed. If the check box is selected, the dialog box wraps the long lines into multiple lines of the text area. If the check box is not selected, the lines are not wrapped and the dialog box shows a horizontal scroll bar to let you scroll the text area.

The Edit Data dialog box shows the following data about the text value being edited:

- Data Type

  This is the data type of the column to which the value belongs, such as `VARCHAR2`, `CHAR`, `LONG`, or `CLOB`.

- Data Length in Bytes

  This is the length in bytes that the edited value would have, if it were stored in the database.

- Data Length in Characters

  This is the length in characters that the edited value would have, if it were stored in the database.

- Post-Conversion Length in Bytes

  This is the length in bytes that the edited value would have, if it were stored in the database after conversion to the target Unicode character set.

- Current Column Limit in Bytes

  This is the maximum length in bytes that the value might currently have to fit into its column in the database.

- Current Column Limit in Characters

  This is the maximum length in characters that the value might currently have to fit into its column in the database. Nothing is displayed if the column uses byte-length semantics.
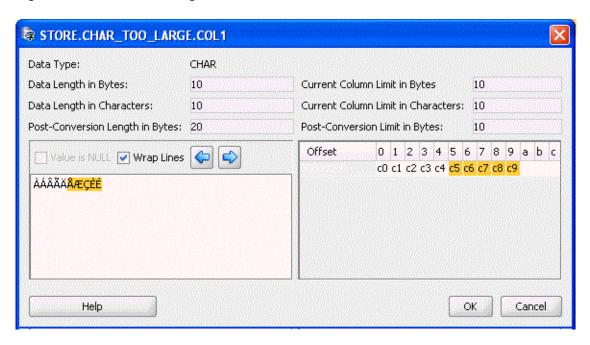
- Post-Conversion Limit in Bytes

  This is the maximum length in bytes that the value will be allowed to have to fit into its column in the database after conversion to Unicode.

When editing a value, you must always ensure that Data Length in Bytes is not greater than Current Column Limit in Bytes and that Data Length in Characters is not greater than Current Column Limit in Characters (if specified). To resolve the "exceeds column limit" issue for the value, you must also ensure that Post-Conversion Length in Bytes is not greater than Post-Conversion Limit in Bytes.

Characters that cause the current value to exceed its Post-Conversion Limit in Bytes are shown on yellow background. This lets you quickly estimate how much a value must be truncated to resolve the "exceeds column limit" issue.

Figure 6-7   Edit Data Dialog Box

In the bottom right, the edited text is displayed as a sequence of hexadecimal numbers corresponding to bytes that comprise (encode) the text when it is stored in the database.

By looking at numeric values of bytes comprising the edited text you can:

- Identify binary values, such as images or encryption results, stored invalidly in a character column

  Such values usually contain some bytes from ranges 0x00 to 0x08 and 0x10 to 0x1f. Because bytes in these ranges are control codes rarely used today, values containing such bytes are almost never pure text values.

- Identify corrupted characters

  If you see replacement characters in the edited text in the text, you can view the binary display to analyze particular bytes corresponding to these characters. By identifying the language of the data and looking at definitions of character sets valid for this language, you

can diagnose the issue as an incorrect character set of data or as corrupted character codes, that is, bytes missing from multibyte codes or random bytes resulting from software bugs or hardware malfunction.

You can insert, modify, or delete bytes in the edited value to fix corruption issues. Click a byte to make it current. Right-click anywhere in the edit area to open the context menu. Select **Insert** to add a byte before the current one. Select **Delete** to remove the current byte from the value. Select **Modify** to change the value of the current byte. Double-clicking a byte is equivalent to selecting it as current and selecting **Modify**.

After making all your intended modifications in the Edit Data dialog box, click **OK** to accept them and to close the dialog box. Click **Cancel**, if you want to close the dialog box and discard the modifications. After accepting the modifications, you still have to click **Save** on the Cleansing Editor tab to save them in the database.

6.4 Displaying Data

The Show Data dialog box is a read-only version of the Edit Data dialog box.

The Show Data dialog box displays values in the data dictionary tables. You are not allowed to modify these values directly.

As in the Edit Data dialog box, you can view the data in text form or in binary form. You can also wrap or unwrap long lines.

6.5 Modifying Columns
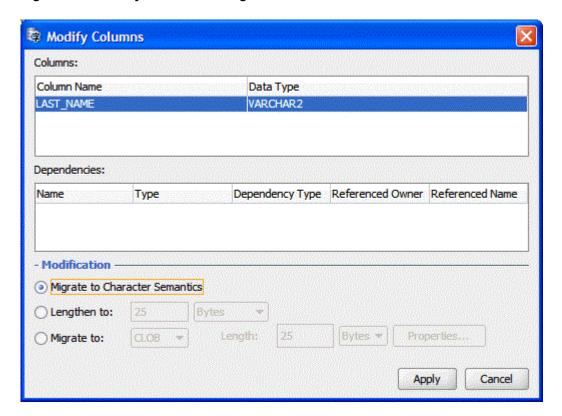
The cleansing actions that you can perform using the Modify Columns dialog box are immediate changes to column definitions (metadata) of a table.

The Modify Columns dialog box is shown in Figure 6-8. You can open it by selecting **Modify Column** from the context menu of the Cleansing Editor (see "Cleansing Data").

Restriction: If a column is used in the defining expression of a virtual column, then the Modify Columns and Schedule Column Modification functionalities are not available for that column, and the corresponding context menu items are not shown.

Figure 6-8   Modify Columns Dialog Box

Columns that will be modified through the Modify Columns dialog box are displayed in the Columns list box. The Dependencies list box shows database objects, such as views and PL/SQL procedures, that depend on those columns and could therefore need updating or recompilation after the columns are altered.

In the Modify Columns dialog box, you can command the following cleansing actions on the columns:

- Migrate to Character Semantics

  You can change the length semantics of the columns from byte to character. If a column to be modified is defined as `VARCHAR2(n BYTE)`, it will become `VARCHAR2(n CHAR)`. Such modification solves "exceed column limit" issues because a n `BYTE` column never contains more than *n* characters and the underlying byte length limit of an n `CHAR` column is automatically increased when the database character set changes to accommodate any string of the length *n* characters (but only up to the data type limit; that is, values reported as "exceed data type limit" will not fit).

- Lengthen to:

  You can choose to lengthen the columns to a new limit, which you specify in the edit field that follows the option label. You can change the length semantics between bytes and characters at the same time by choosing the desired semantics from the following drop-down list. To solve "exceed column limit" issues, specify a limit that is greater than maximum post-conversion lengths reported for all the columns.

- Migrate to:

  You can migrate some data types to a new data type. A `CHAR` column can be migrated to `VARCHAR2`, `CLOB`, or `RAW`. A `VARCHAR2` column can be migrated to `CLOB` or `RAW`. A `LONG` column can be migrated to `LONG RAW` or `CLOB`. If you choose migration to `CLOB`, a properties dialog

box is available that enables you to modify the tablespace and storage properties of the new `CLOB` segment. If you choose migration to `RAW` or `VARCHAR2`, you can specify the new length limit, provided it is not smaller than any of the byte length limits of the columns to be migrated. If you choose migration to `VARCHAR2`, you can specify the length semantics in addition to the new length. Migration to `VARCHAR2` or `CLOB` solves the "exceed data type limit" issues, because the new data types are larger than the old ones. Migration to `RAW` or `LONG RAW` solves the "invalid binary representation" issues caused by binary data stored in character columns by ensuring that the data is indeed treated as binary by the database.

Migrations to `VARCHAR2`, `RAW`, and `LONG RAW` are metadata-only operations and are therefore very fast. Migration to `CLOB` requires data to be updated in all rows of the table. It could, therefore, be very resource-consuming for large tables and it might fail, if enough undo space is not available in the database.

After you have defined the cleansing action, click **Apply**. You are asked to confirm that you indeed want to modify the columns immediately. Click **Yes** and the appropriate SQL DDL statements are sent to the database for immediate execution.

6.6 Scheduling Column Modification

The Schedule Column Modification dialog box is almost identical to the Modify Columns dialog box, described previously.

The two differences are:

- The cleansing actions specified in this dialog box are scheduled actions. They are not executed immediately when you select **Apply**. Instead, they are saved in the DMU repository and are executed in the conversion step of the character set migration process. This enables you to synchronize any necessary application upgrades related to changes in table structure with the database conversion.

- The additional No Modification option is available. It enables you to recall a previously scheduled cleansing action before the database conversion starts.

To schedule column modification:

- Open the Schedule Column Modification dialog box by selecting **Schedule Column Modification** from the context menu of the Cleansing Editor (see "Cleansing Data: Using the Toolbar"). The Schedule Column Modification dialog box appears, as illustrated in Figure 6-33.

6.7 Modifying Attributes

The cleansing actions that you can perform using the Modify Attributes dialog box are immediate changes to attributes of user-defined object types (ADT).

A change to an object attribute affects all tables that reference that attribute. You must use the Modify Attributes dialog box to change table columns that store object attribute values. The Modify Columns dialog box is not available for such columns.

Attributes that will be modified through the Modify Attributes dialog box are displayed in the Attributes list box. The Dependencies list box shows database objects, such as tables, views and PL/SQL procedures, that depend on those attributes and could therefore need updating or recompilation after the attributes are altered.
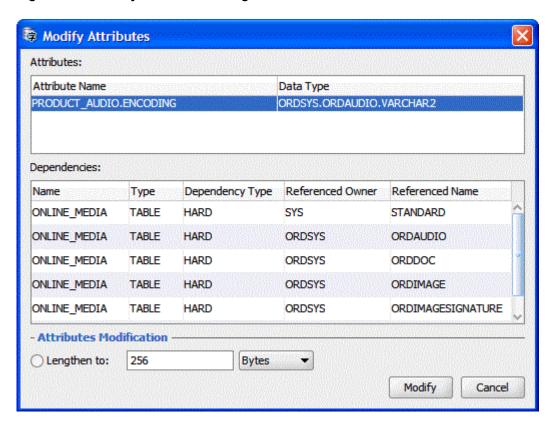
At present, you can define only one cleansing action in the Modify Attributes dialog box, which is to lengthen the attributes to a new limit. You specify the limit in the edit field. You can change

the length semantics between bytes and characters at the same time by choosing the desired semantics from the following drop-down list. To solve "exceed column limit" issues, specify a limit that is greater than maximum post-conversion lengths reported for all columns storing values of the attributes in all tables in the database.

To modify attributes:

1. Select **Modify Attribute** from the context menu of the Cleansing Editor (see "Cleansing Data: Using the Toolbar"). The Modify Attributes dialog box appears, as is illustrated in Figure 6-9.

2. After you have defined the cleansing action, click **Modify**.

3. You are asked to confirm that you indeed want to modify the attributes immediately. Click **Yes** and the appropriate SQL DDL statements are sent to the database for immediate execution.

Figure 6-9   Modify Attributes Dialog Box

6.8 Scheduling Attribute Modification

The Schedule Attribute Modification dialog box is almost identical to the Modify Attributes dialog box.

The two differences are:

•   The cleansing action specified in this dialog box is a scheduled action. It is not executed immediately when you select **Save**. Instead, it is saved in the DMU repository and is executed in the conversion step of the character set migration process. This enables you

to synchronize any necessary application upgrades related to changes in table structure with the database conversion.

- The additional No Modification option is available. It enables you to recall a previously scheduled cleansing action before the database conversion starts.

You can open the Schedule Attribute Modification dialog box by selecting **Schedule Attribute Modification** from the context menu of the Cleansing Editor (see "Cleansing Data: Using the Toolbar").

6.9 Ignoring Convertibility Issues

While not strictly cleansing actions, two column properties allow you to handle convertibility issues reported in a column.

You can set these properties in the Converting subtab of the Column Properties tab, described in "Viewing and Setting Column Properties".

The first property is called Allow Conversion of Data with Issues. If you set this column property to Yes, any convertibility issues reported for the column are ignored. That is, they do not prevent you from starting the conversion step. The column undergoes the conversion to Unicode and the resulting values are automatically truncated to fit into the column. Invalid character codes are converted to replacement characters. This option is useful if you encounter some convertibility issues in auxiliary or historical data that you do not require to be fully correct after conversion, and that you do not want to remove from the database yet.

The second property is called Exclude from Conversion. If you set this column property to Yes, the column is skipped in the conversion step. Column values remain identical to what they were before character set migration. This option is useful to prevent conversion of a column that:

- Contains a mixture of values in multiple character sets, which came in the pass-through configuration from various clients working in different character sets. Such values must be converted manually after the DMU converts the rest of the database and updates the database character set to Unicode.

- Has the data type `VARCHAR2` and a length constraint greater than 2000 bytes, and stores binary data. You cannot migrate such a column to `RAW`, which is limited to 2000 bytes, therefore, you might decide to keep accessing the column in the pass-through configuration after migration to Unicode, before you can update your applications to support this data through the `BLOB` data type.

Convertibility issues reported in columns marked for exclusion from conversion do not prevent you from starting the conversion step. See "Excluding Columns and Tables From Migration" for more information about setting this property.

6.10 Bulk Cleansing

You can perform some cleansing operations in bulk.

You can perform the following actions in bulk:

- Alter the length semantics of `CHAR` and `VARCHAR2` table columns from byte semantics to character semantics

  Character semantics is useful for defining the storage requirements for multibyte strings of varying widths. For example, in a Unicode database (AL32UTF8), suppose that you need to define a `VARCHAR2` column that can store up to five Chinese characters along with five English characters. Using byte semantics, this column requires 15 bytes for the Chinese

characters, which are three bytes long, and 5 bytes for the English characters, which are one byte long, for a total of 20 bytes. Using character semantics, the column requires 10 characters. Character semantics allows the specification of the column size in terms of characters independent of the database character set. The benefits of using character length semantics in Unicode migration are further described in "Changing the Length Semantics".

You can define bulk cleansing actions for migrating to character length semantics in either the immediate or scheduled cleansing mode. In the scheduled mode, the migration to character length semantics will take place during the conversion phase without any interruption to the production environment. When the DMU is used in validation mode, only the immediate bulk cleansing to character length semantics is available.

Bulk cleansing supports the migration of the following data type columns to character length semantics:

– `CHAR` built-in data type columns

– `VARCHAR2` built-in data type columns

– `VARCHAR2` attribute columns of ADT

The following columns are excluded:

– `CHAR` attribute columns of ADT

– Columns in clusters

– Columns on which partition keys are defined

As in regular cleansing, bulk cleansing does not support character length semantics migration for Oracle-supplied schemas, such as Oracle e-Business Suite schemas.

• Set the "Allow Conversion of Data with Issues" column conversion property

If the scan results of the column show that some values have conversion issues, that is, the converted values will contain replacement characters or they will be truncated, you can still let the DMU convert the data by setting this property to Yes. This might be useful if you want to automatically truncate data that exceeds the column or data type limit or you are not concerned about a few corrupted values that happen to exist in the column but are of no real significance for applications that use this data.

• Replace or remove occurrences of a byte or character pattern in column values

In some cases, after analyzing the nature of individual data exceptions, you may decide to modify the data by replacing the offending bytes or characters with a set of valid equivalent bytes or characters or removing them altogether. If the same problem symptom exists in multiple database objects, it may be desirable to perform such replacement operation in bulk. The pattern-based replacement cleansing option allows you to replace all occurrences of a source pattern with a target pattern for the selected database objects. You can perform the replacement for single bytes or characters or specify substrings of bytes or characters.

> **Note:**
>
> The following restrictions apply when bulk cleansing blockchain and immutable tables:
>
> - Setting the "Allow Conversion of Data with Issues" column conversion property is supported only for scheduled bulk cleansing, and only for invalid data.
>
> - Replacing or removing occurrences of a byte or character pattern in column values is not supported.
>
> See "Migrating Blockchain and Immutable Tables" for more information.

6.10.1 Bulk Cleansing Data from Byte Semantics to Character Semantics

You can bulk cleanse data from byte semantics to character semantics.

To bulk cleanse the data:

1. Under the Migration menu, select **Bulk Cleansing**. Figure 6-10 appears.

**Figure 6-10   Bulk Cleansing Wizard - Welcome**

2. Click **Next** and Figure 6-11 appears.
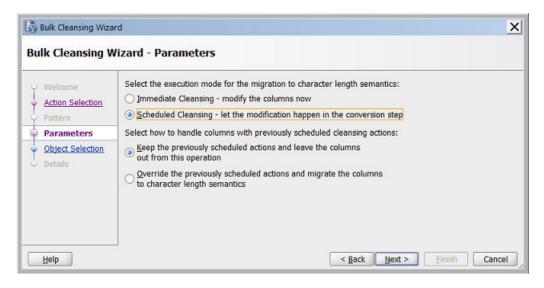
**Figure 6-11   Bulk Cleansing Wizard - Action Selection**

Select **Migrate to character length semantics** and click **Next**.

3.   Figure 6-12 appears.

Figure 6-12   Bulk Cleansing Wizard - Parameters

Select the execution mode for bulk cleansing. For the scheduled cleansing mode, you can select how to handle columns with previously scheduled cleansing actions by either keeping the previously scheduled actions or override them and migrate the columns to character length semantics.

4.   Click **Next** and Figure 6-13 appears.

Figure 6-13   Bulk Cleansing Wizard - Object Selection

Select the objects to be migrated to character length semantics. The Data Type and Scheduled Cleansing Action columns show the current data types of the columns and any existing scheduled cleansing actions. Objects that are already in character length semantics or objects that are not supported for character length semantics migration are not selectable in the navigator tree.

5. Click **Next** and appears.

Figure 6-14   Bulk Cleansing Wizard - Details

The Bulk Cleansing Details page displays a list of the table columns and type attributes that will be migrated to character length semantics.

6. To confirm and start the bulk cleansing operations, click **Finish**. Figure 6-15 appears.

Figure 6-15   Bulk Cleansing Progress

You can monitor the status of the bulk cleansing on the Bulk Cleansing Progress panel. If an operation fails on a table column or a type attribute during the process, you can highlight the entry to see the error message in the Status column as well as at the bottom of the panel. To view only the entries with errors, select the **Show Only Errors** check box at the top.

You can suspend the cleanse by clicking **Stop**.

6.10.2 Bulk Cleansing Data by Allowing Conversion of Data with Issues

You can bulk cleanse data by allowing conversion of data with issues.

To bulk cleanse the data:
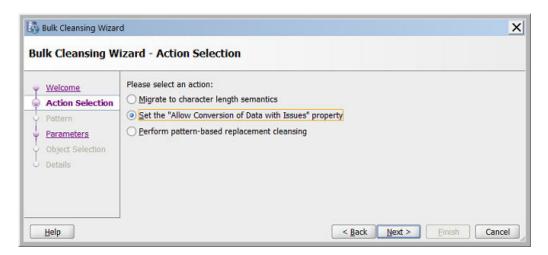
1. i
   Under the Migration menu, select **Bulk Cleansing**. Figure 6-16 appears.

Figure 6-16   Bulk Cleansing Wizard - Welcome
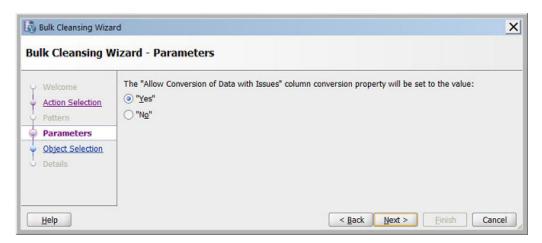
2. Click **Next** and Figure 6-17 appears.

Figure 6-17   Bulk Cleansing Wizard - Action Selection
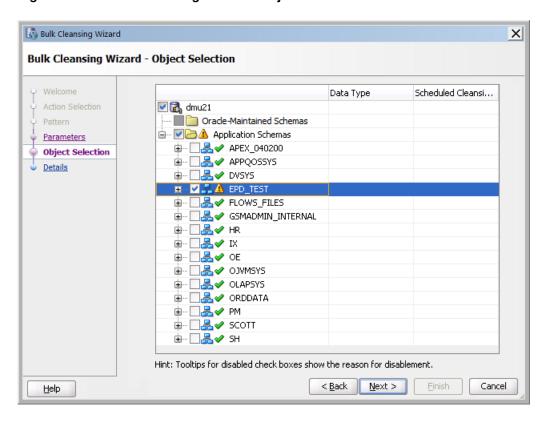
3. Select **Set the "Allow Conversion of Data With Issues"**. Click **Next** and Figure 6-18 appears.

Figure 6-18   Bulk Cleansing Wizard - Parameters

4. Select the desired value and click **Next** and Figure 6-19 appears.

Figure 6-19   Bulk Cleansing Wizard - Object Selection
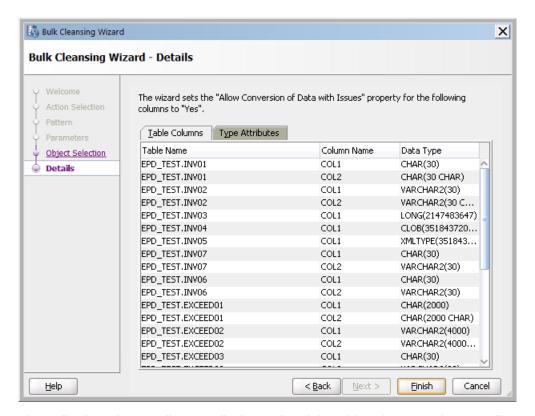
Select the objects for which the "Allow Conversion of Data with Issues" property is to be set. The Data Type and Scheduled Cleansing Action columns show the current data types of the columns and any existing scheduled cleansing actions.
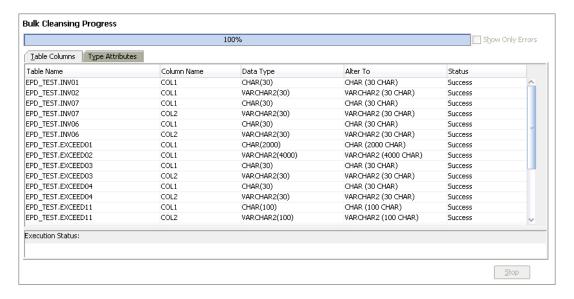
5. Click **Next** and Figure 6-20 appears.

Figure 6-20   Bulk Cleansing Wizard - Details

6. The Bulk Cleansing Details page displays a list of the table columns and type attributes for which the "Allow Conversion of Data with Issues" property will be set. To confirm and start the bulk cleansing operations, click **Finish**. Figure 6-21 appears.

Figure 6-21   Bulk Cleansing Progress

You can monitor the status of the bulk cleansing on the Bulk Cleansing Progress panel. If an operation fails on a table column or a type attribute during the process, you can

highlight the entry to see the error message in the Status column as well as at the bottom of the panel. To view only the entries with errors, select the **Show Only Errors** check box at the top.

You can suspend the cleanse by clicking **Stop**.

6.10.3 Bulk Cleansing Data with Pattern-Based Replacement

You can bulk cleanse data by using pattern-based replacement.

To bulk cleanse data:

1. Under the Migration menu, select **Bulk Cleansing**. Figure 6-22 appears.

Figure 6-22   Bulk Cleansing Wizard - Welcome

2. Click **Next** and Figure 6-23 appears.
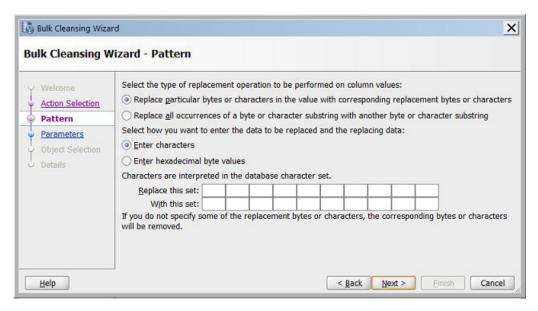
Figure 6-23   Bulk Cleansing Wizard - Action Selection

Select **Perform pattern-based replacement cleansing** and Figure 6-24 appears.

3. Select the type of replacement operation to be performed and how you want to enter the data. You can perform the replacement in terms of single bytes or characters or use substrings of bytes or characters. The source and replacement values can be entered in either characters or hexadecimal byte representation. If you leave some of the replacement bytes or characters as empty, the corresponding source bytes or characters will be removed from the selected target columns.
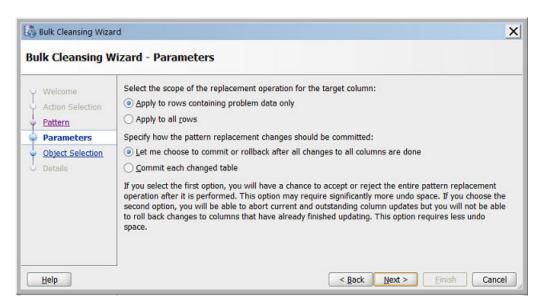
Figure 6-24　Bulk Cleansing Wizard - Pattern

4. Click **Next** and Figure 6-25 appears. The Parameters page allows you to specify two things - scope of pattern replacement changes and how you want to commit those changes.

   • Specify the scope for the pattern replacement changes: Select the first option if you want to do pattern replacement for the rows that contain problem data only. Select the second option if you want to do the pattern replacement for all the rows.

   • Specify how to commit pattern replacement changes: Select the first option if you want to have a chance to accept or reject the entire operation at the end. Select the second option if you want all changes to be committed immediately.
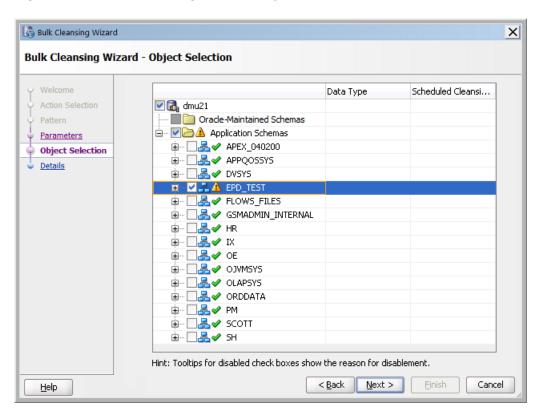
Figure 6-25   Bulk Cleansing Wizard - Parameters

5. Specify the scope of the replacement and how to commit. Click **Next** and Figure 6-26 appears. If you select pattern-based replacement for the rows that contain problem data only, then the check boxes for the nodes in the selection tree are disabled, if they do not contain any problem data or do not have collected rowids. Mouse over a disabled node to see the tooltip with more information.

Figure 6-26   Bulk Cleansing Wizard - Object Selection

6. Select the objects on which the pattern-based replacement is to be performed. Click **Next** and Figure 6-27 appears.

Figure 6-27   Bulk Cleansing Wizard - Details

7. The Bulk Cleansing Details page displays a list of the table columns and type attributes that are to be cleansed. To confirm and start the bulk cleansing operations, click **Finish**. Figure 6-28 appears.
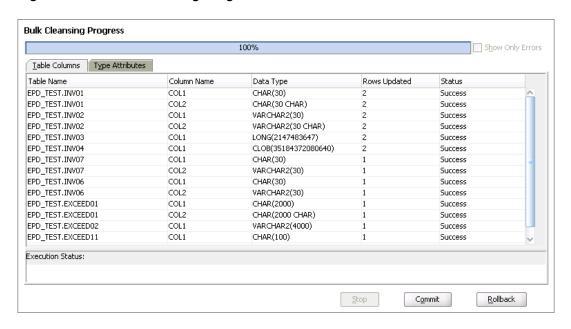
Figure 6-28   Bulk Cleansing Progress

You can monitor the status of the bulk cleansing on the Bulk Cleansing Progress panel. If an operation fails on a table column or a type attribute during the process, you can highlight the entry to see the error message in the Status column as well as at the bottom

of the panel. To view only the entries with errors, select the **Show Only Errors** check box at the top.
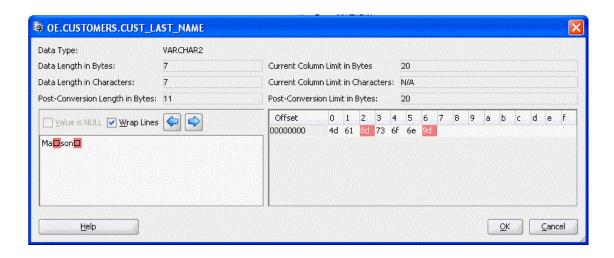
You can suspend the cleanse by clicking **Stop**.

6.10.4 Bulk Cleansing Examples

This section includes examples of the bulk cleansing functionality.

Example 6-1   Replacing Invalid Bytes in Bulk

In this example, the database contains occurrences of invalid bytes (0x8D and 0x9D in a WE8MSWIN1252 database), as illustrated in the following graphic.
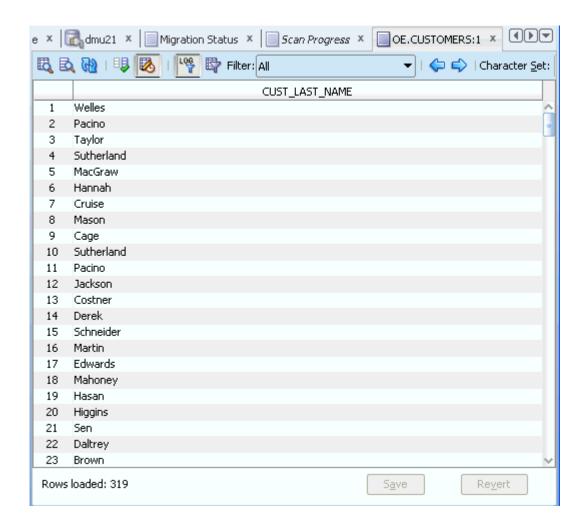

If there are multiple occurrences of the same invalid data symptom, you may decide to perform pattern-based cleansing to resolve the issues in bulk. In this case, assuming the byte values 0x8d and 0x9d are ignorable and insignificant to the data contents, you may define a pattern-based replacement to remove them from your data as below.
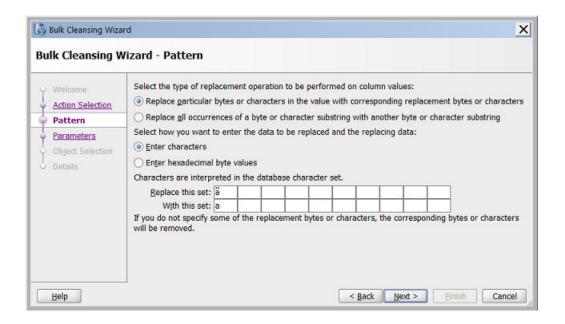
After this has been done, a similar view of the data is illustrated in the following screenshot.

Note that the invalid data in line 8 has been removed.

Example 6-2   Replacing Characters in Bulk

In some cases, it's possible you may decide to replace 8-bit characters with their 7-bit equivalents to deal with data size expansion issues when converting to Unicode without changing the column definitions. For example, you may replace occurrences of the character ä with a using the bulk cleansing method as is illustrated in the following screenshot.
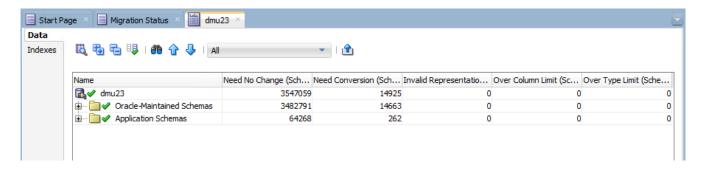
6.11 Cleansing Scenario 1: A Database with No Issues

This scenario demonstrates the results when there are no issues with data in your database.

The Scan Report resembles Figure 6-29.

Figure 6-29   Database Scan Report: No Data Issues

The report column Need No Change (Scheduled) shows the number of character column values (cells) in the database that are already valid in the target Unicode character set. This means that these cells are either null or they contain only ASCII characters (except when the migration is from UTF8 to AL32UTF8, in which case the characters might be other valid UTF-8 codes from the Basic Multilingual Plane of Unicode.). The report column Need Conversion (Scheduled) shows the number of character cells that need conversion to become Unicode but whose conversion will not cause any errors or data loss.

The report columns Invalid Representation (Scheduled), Over Column Limit (Scheduled), and Over Type Limit (Scheduled) show zero, which means that the database content has no convertibility issues. The green OK icon to the left of the database node means the same thing. If the Migration Status tab shows no other issues in the database (see "Following the Status of the Migration") you can proceed to the actual conversion phase of the migration process, which is described in "Converting the Database".

6.12 Cleansing Scenario 2: Cleansing Expansion Issues

This scenario illustrates the results when there are expansion issues with the data.

The Scan Report will resemble Figure 6-30. In addition to the columns Need No Change (Scheduled) and Need Conversion (Scheduled), described in "Cleansing Scenario 1: A Database with No Issues", the columns Over Column Limit (Scheduled) and Over Type Limit (Scheduled) contain nonzero values. A yellow triangle icon next to the database node signals convertibility issues with the data.

After conversion to the target character set, the cells classified as Over Column Limit (Scheduled) will no longer fit into their columns unless the columns are lengthened. The cells classified as Over Type Limit (Scheduled) will become longer than the limit of their data types. The lengthening of their columns will not help.

You can expand the object tree in the Name column to locate which schemas, which tables in those schemas, and which columns in those tables contain the problematic cells. The yellow triangle icons show which nodes to expand. The columns Over Column Limit (Scheduled) and Over Type Limit (Scheduled) show for each schema, table, and column how many problematic cells are in the corresponding database object.
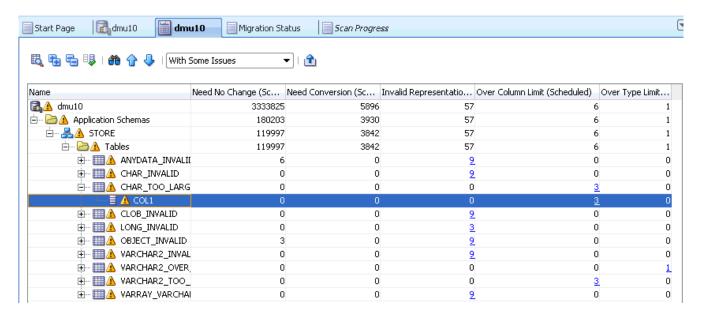
Figure 6-30   Database Scan Report: Expansion Issues

You can also reduce the amount of information on the Scan Report tab and let the DMU show scan results only for columns that have issues. To do this, click the filter drop-down list to open it, as shown in Figure 6-31. Select the option **With Some Issues** to see all database columns with data issues or the option **With Length Issues** to see only columns that have data expansion issues discussed in this section.

Figure 6-31   Filtering Scan Results

The non-zero counter values for Over Column Limit (Scheduled) and Over Type Limit (Scheduled) cells at the table and column level are active links. If you click a counter, it will open the Cleansing Editor tab in the client pane of the DMU window, as shown in Figure 6-32. Another way to open this tab is to right-click the table node in the Navigator pane or the Scan Report tree and then select **Cleansing Editor**.
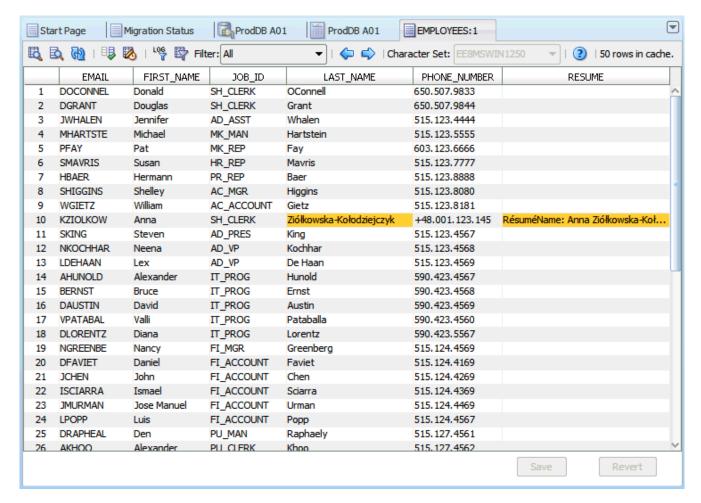
The Cleansing Editor tab shows the contents of the table for which it was opened. By default, the data cells that have length expansion issues are shown on a yellow background. You can change this color in the Preferences dialog box, which can be opened by selecting **Preferences** from the Tools menu.

Various filtering and search features of the Cleansing Editor can help you locate the problematic data cells. See "Filtering Data" for more information about these features.

With the Cleansing Editor tab open, you can now try to cleanse the reported expansion issues as described in the following sections.

Repeat the cleansing process for all tables with issues reported by the DMU. As soon as no more issues are reported in either the Scan Report or the Migration Status tab, you can start the conversion phase, described in "Converting the Database".

Figure 6-32   Cleansing Editor Tab: Expansion Issues

6.12.1 Over Column Limit Issues

You can cleanse the Over Column Limit issues in a column.

You have the following options:

- Lengthen the column.

- Change the length semantics of the column from bytes to characters.

- Shorten the stored values manually.

- Allow the DMU to truncate the values during conversion.

- Edit the value to replace characters that expand in conversion.

- Migrate to a larger data type.

The choice of cleansing action depends mainly on your ability to modify applications accessing the cleansed table. If you are unable to modify applications, for example, because they are not developed in your company, then you usually cannot lengthen a column, change its length semantics, or migrate to another data type, because all these actions require modifications to the application code to adapt buffer sizes or data access methods to the changed column data types.
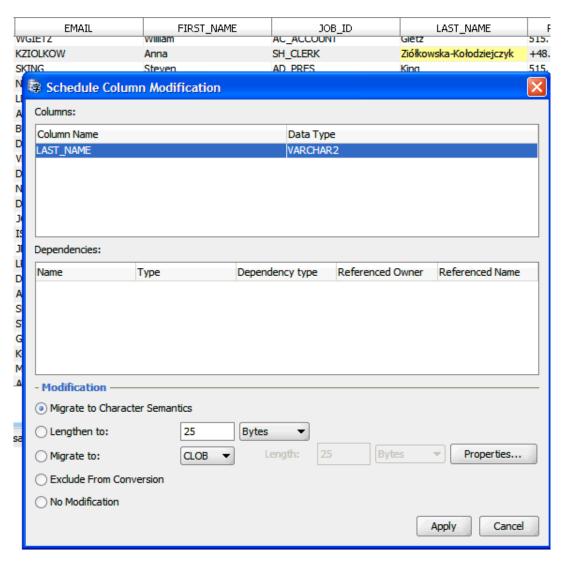
6.12.1.1 Lengthening a Column

Lengthening a column is the most efficient way to deal with "over column limit" issues.

This technique is viable if applications accessing the column either adapt to the increased length automatically or require only a few manual changes in a few well-defined places in code.

To lengthen a column:

1. To lengthen a column, right-click any cell of the column in the Cleansing Editor to open the context menu. In the menu, select either **Schedule Column Modification** or **Modify Column**. The former opens the Schedule Column Modification dialog box, shown in Figure 6-33. The latter opens the Modify Column dialog box, which looks almost the same, except that it does not include the Exclude From Conversion and No Modification options.

Figure 6-33   Schedule Column Modification: Lengthen the Column Size

2. The Schedule Column Modification dialog box lets you define a scheduled cleansing action on a column, which is to be performed only during the actual conversion phase of the migration process. The action is saved in the DMU repository and applied to the table just

before or while the table is converted. The table is not modified before that time and defining the action does not disturb the current processing in the database. New application versions adapted to the change need to be installed only at the time of conversion.

If you prefer the cleansing action to be performed immediately, use the Modify Column dialog box. After you click **Apply** in this dialog box, the DMU immediately sends the appropriate SQL DLL statements to the database. The resulting modification cannot be rolled back.

To lengthen a column in the Schedule Column Modification or in the Modify Column dialog box, select **Lengthen to** and enter a new length value that is equal to or higher than the maximum length of all post-conversion values in the column. The maximum post-conversion length is reported on the Scanning subtab of the Column Properties tab. See "Scanning the Database" in the "Column Properties" section.

When lengthening a column, leave the drop-down list set to "Bytes". Click **Apply**.

If a scheduled lengthening of a column will cause the maximum allowed index key size to be exceeded for an index defined on the column, a warning dialog is displayed. In this dialog, you have the option to continue to schedule the change (**Yes**) or cancel the change (**No**). If you choose to continue, you can rescan the database and use the index check report to fix the problematic index. See "Database Scan Report: Index Check Report" for more details.

6.12.1.2 Changing the Length Semantics

Changing the length semantics of a `VARCHAR2` or `CHAR` column from bytes to characters is a special form of lengthening the column.i

By changing the length semantics of a column, you tell the database to constrain values stored in the column to a given number of characters and not to a given number of bytes that comprise the encoding of these characters. The length constraint becomes independent of the character set. The database sets the implicit byte limit of each character length semantics column to the maximum possible byte length of a character value that has the declared maximum number of characters. This implicit byte limit is equal to the maximum number of characters multiplied by the maximum number of bytes in a character code in the database character set.

The number of characters does not change in conversion from one character set to another (except for conversion of supplementary Unicode characters from ZHT16HKSCS31 to surrogate pairs in UTF8). Therefore, because the number of characters fitting into n bytes is not greater than n, if a column length constraint is changed from n bytes to n characters, then all column values comply with this new constraint after database character set conversion. The implicit byte limit is automatically adjusted by the database character set change. One caveat is that the maximum length of a character value is still restricted by the byte limit of its data type: 4000 bytes for `VARCHAR2` and 2000 bytes for `CHAR`. Values that exceed this limit after conversion are classified as Over Type Limit. The handling of such values is described in "Handling Over Type Limit Issues".

Choose migration to character length semantics over lengthening a column if, when adapting your applications to process Unicode, you make them manage character value limits in characters and not bytes.

To change length semantics:

To change the length semantics of a column, follow the previous steps described in "Lengthening a Column" to open either the Schedule Column Modification or Modify Column dialog box.

1. First, instead of selecting **Lengthen to**, select **Migrate to Character Semantics**.

2. Although not usually necessary, you can migrate a column to character length semantics and at the same time make the declared maximum number of characters greater than the current declared maximum number of bytes. To do so, select **Lengthen to**, enter the new character limit for the column into the adjacent text field, and select **Characters** from the drop-down list.

3. Click **Apply**.

   If a scheduled change to the length semantics of a column will cause the maximum allowed index key size to be exceeded for an index defined on the column, a warning dialog is displayed. In this dialog, you have the option to continue to schedule the change (**Yes**) or cancel the change (**No**). If you choose to continue, you can rescan the database and use the index check report to fix the problematic index. See "Database Scan Report: Index Check Report" for more details.

6.12.1.3 Shortening Character Values Manually

If the column data types cannot be altered, then you must modify the stored character values so that they do not expand over the column length limit in conversion to the target character set.

There are three ways to achieve this goal:

- Shorten the values manually.
- Let the DMU shorten the values during conversion.
- Replace some expanding characters with non-expanding equivalents.

Manual shortening of values requires that the number of cells with the Over Column Limit issue is small enough so that the operation can be completed within a reasonable time. Manual shortening of values gives you the best control over the cleansing results, but it is very time-consuming. Shortening values is not a feasible cleansing action for data that has precisely defined contents, such as family names.

To shorten a character value manually:

1. To shorten a character value in a given cell, which is marked with a yellow background in the Cleansing Editor, double-click the cell or right-click it and select **Edit Data** from the context menu. The Edit Data dialog box that contains the value to be edited is displayed, as shown in Figure 6-34.

Figure 6-34   Edit Data Dialog Box

In the Edit Data dialog box, the trailing part of the character value being edited that causes the value to exceed the maximum column length after conversion is marked with a yellow background. Therefore, delete characters from the value until no yellow background is visible. You can also compare the number shown in Post-Conversion Length in Bytes with the number shown in Post-Conversion Limit in Bytes and delete characters until the first number is smaller than or equal to the second number.

2. After you are ready with the changes, click **OK** to accept the new value and close the dialog box. The new value is displayed in the Cleansing Editor but it is not yet stored in the database. You can now edit further cells. If you want to store the new values permanently in the database, click **Save** on the Cleansing Editor tab. You can reverse any edit changes made since the last save by clicking **Revert**.

Editing values is always an immediate cleansing action. You cannot schedule manual editing changes to be applied only during conversion.

6.12.1.4 Truncating Column Values During Conversion

Data in a column may be historical and no longer required for business purposes. Alternatively, the essential information may be always contained at the very beginning of a cell value. In such cases, you can choose to let the DMU automatically truncate the values during conversion so that they comply with the existing length constraint of the column.

This approach is not considered a cleansing action because its effects are not reflected in the scan results. However, the DMU ignores columns that you allow to automatically truncate when deciding if conversion should be disallowed due to data convertibility issues.

To truncate column values during conversion:

You can have the DMU automatically truncate column values to comply with the existing length constraint of a column. The steps are:

1. Open the Column Properties tab.

2. Open the Readiness subtab.

3. Select **Yes** for the value of the property Allow Conversion of Exceptional Data.

> ✎ **See Also:**
>
> Viewing and Setting Object Properties in the DMU for more information about setting column properties

6.12.1.5 Replacing Expanding Characters

If neither changing the data type of a column nor shortening column values is possible, consider as a last resort avoiding the expansion by replacing the expanding non-ASCII characters with ASCII equivalents.

Examples are:

- Replace nonbreaking space (0xA0 in most single-byte ISO and Microsoft Windows code pages) with the standard space (0x20 in all ASCII-based character sets).
- Replace *smart* quotation mark characters, such as ', ', ", ", and „ with ASCII single- and double-quotation mark characters ' and ".
- Replace currency symbols with approximated representation: © (Yen) becomes Y, £ (Pound) becomes L, ¢ (cent) becomes c, and so on.
- Remove accents from Latin letters, if values remain acceptable after this operation.

You can replace characters in cells by editing them in the Edit Data dialog box, as described in "Shortening Character Values Manually". This is feasible if the number of cells to edit is not large. To replace characters in many cells, use a database tool, such as Oracle SQL Developer, and issue an `UPDATE` SQL statement with the `TRANSLATE` function to replace all relevant characters in a table at one time.

6.12.1.6 Migrating to a Larger Data Type

Migration of a column data type is not required if the column does not contain values that exceed the data type limit.

But if such values are expected in the future, you can choose to migrate the column data type to a larger one. The choices are:

- Migrate a `CHAR` column to `VARCHAR2`.
- Migrate a `CHAR` column to `CLOB`.
- Migrate a `VARCHAR2` column to `CLOB`.
- Migrate a `LONG` column to `CLOB`.

The DMU does not support migration to the deprecated `LONG` data type.

Migration of column data types usually requires code changes to adapt applications to the comparison and padding semantics difference between `CHAR` and `VARCHAR2`, or to different ways of fetching `CLOB` values.

> ✎ **See Also:**
>
> *Oracle Database SQL Language Reference* for more information about the LOB data types

To migrate the data type of a column:

1. To migrate the data type of a column, open either the Schedule Column Modification dialog box or the Modify Column dialog box for this column, as described in "Lengthening a Column". Then, select **Migrate to** and select the target data type from the adjacent drop-down list. If the target data type is VARCHAR2, enter the desired target length and length semantics. If the target data type is CLOB, click **Properties** to open the CLOB Properties dialog box. You can enter storage parameters for the CLOB segment in this dialog box. Click **Apply** to close the dialog box and save the parameters.

2. Click **Apply** in the Schedule Column Modification or Modify Column dialog box to accept the defined cleansing action. If you chose to migrate the data type immediately by using the Modify Column dialog box and the migration is to the CLOB data type, the following processing might take a long time depending on the amount of column data to migrate.

The migration from CHAR to VARCHAR2 does not remove trailing blanks from the CHAR values. If you want to remove the blanks, use a database tool, such as Oracle SQL Developer, and issue an UPDATE SQL statement with the RTRIM function.

6.12.2 Handling Over Type Limit Issues

The next step is to resolve the problems with Over Type Limit cells.

To cleanse the Over Type Limit issues in a column, you have the following options, which are a subset of options available to cleanse the Over Column Limit issues:

- Migrate to a larger data type.
- Shorten the stored values manually.
- Allow the DMU to truncate the values during conversion.
- Edit the value to replace characters that expand in conversion.

Similar to what is described in "Over Column Limit Issues", the choice of the cleansing action depends mainly on your ability to modify applications accessing the cleansed table. If you are unable to modify applications, then you cannot migrate to a larger data type.

See "Over Column Limit Issues" for the steps required to perform any of the preceding available cleansing actions.

6.12.3 Refreshing Scan Results

When you define a column cleansing action in the Schedule Column Modification or Modify Column dialog box, or you edit a value in the Edit Data dialog box, the DMU invalidates the scan results for the column. The exception is when the DMU can adjust the results based on the performed cleansing action.

In case the results are invalidated, you should rescan the column.

To refresh scan results:

1. To rescan a column, right-click the column node in the Scan Report tree or the Navigator tree, and select **Scan**. The Scan Wizard appears with only this column selected for scanning.

2. Accept the scan parameters and click **Finish** to start the scan.

   The new scan will account for the just-performed cleansing action when analyzing the issues. If the column is cleansed properly, no issues will be reported for it in the Over Column Limit (Scheduled) report column.

If you define a scheduled cleansing action, you are still able to view the results for the current table structure and contents in the Over Column Limit optional report column on the Scan Report tab (see "Scanning the Database") or in the Current Data column of scan results displayed on the Scanning subtab of the Column Properties tab.

6.12.4 Revoking Scheduled Cleansing Actions

A scheduled cleansing action for a column can be revoked at any time before the database is converted.

To revoke scheduled cleansing actions:

1. Open the Schedule Column Modification dialog box for the affected column.

2. Select **No Modification**.

3. Click **Apply**.

   The scan results for the column will be invalidated and you will have to rescan the column.

6.13 Cleansing Scenario 3: Cleansing Invalid Representation Issues

There may be invalid representation issues with your data. Filtering and search features of the Cleansing Editor can help you locate problematic data cells.

In this case, the Scan Report resembles what is shown in "Overview of the Database Scan Report", but in addition to or in place of the report columns Over Column Limit (Scheduled) and Over Type Limit (Scheduled), the report column Invalid Representation (Scheduled) contains nonzero values. A yellow triangle icon next to the database node signals convertibility issues with the data.

Analogously to how columns with length expansion issues are located in the Scan Report, you can locate columns containing cells with invalid binary representation issues. See "Scanning the Database" for more details.

Invalid binary representation issue in a cell value means that the value is not valid in its assumed (declared) character set, which is by default the database character set. Some bytes in the value do not form a valid character code in the declared character set.

There are three possible reasons for a value to have invalid binary representation:

- An application stores binary values, that is, values that are not to be interpreted as pure text, in a pass-through configuration. The most common examples are images or documents in proprietary binary formats stored in `LONG` columns and encrypted text values

stored in `VARCHAR2` columns (for example, results from `DBMS_OBFUSCATION_TOOLKIT` procedures that return `VARCHAR2` results).

- An application stores character values in a pass-through configuration and the values are encoded in one or more character sets different from the database character set. The most common examples are data in various Microsoft Windows client character sets stored in databases with the US7ASCII or various ISO 8859 database character sets.

- An application stores values that have some character codes corrupted due to software defects or hardware failures.

See "Invalid Binary Storage Representation of Data" for a description of the pass-through configuration.

To analyze which of the three possible reasons caused invalid binary representation data to appear in a given column, open the Cleansing Editor tab for this column by clicking the nonzero Invalid Representation (Scheduled) counter value for the column. When you click the counter value, the DMU opens the Cleansing Editor tab in the client pane of the DMU window, as shown in Figure 6-35. Another way to open this tab is to right-click the table node in the Navigator pane or the Scan Report tree and to select **Cleansing Editor**.

Various filtering and search features of the Cleansing Editor can help you locate the problematic data cells. See "Cleansing Data" for more information about these features.

By default, the data cells that have invalid binary representation issues are shown on a light coral background. You can change this color in the Preferences dialog box, which can be opened by selecting **Preferences** from the Tools menu.
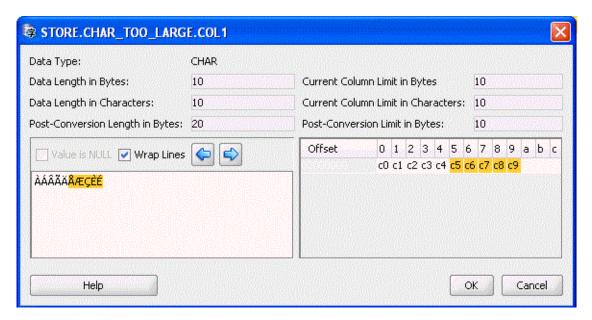
Figure 6-35   Cleansing Editor Tab: Invalid Binary Representation Issues

As in the case of expansion issues, you can double-click a cell with invalid binary representation issues to open the Edit Data dialog box, as shown in Figure 6-34.

In the Edit Data dialog box, you can view the binary data by looking in the bottom-right pane. The bytes comprising the edited value are displayed as 2-digit hexadecimal numbers, as shown in Figure 6-36.

Figure 6-36   Edit Data Dialog Box: Viewing Binary Data

With the Cleansing Editor tab and the Edit Data dialog box open, you can now try to identify the reason for invalid binary representation issues and cleanse the issues as described in the following sections. The considerations discussed in "Refreshing Scan Results" and "Revoking Scheduled Cleansing Actions" apply to the cleansing of invalid binary representation issues as well.

Repeat the cleansing process for all tables with issues reported by the DMU. When no more issues are reported in either the Scan Report or the Migration Status tab, you can proceed to the conversion phase, described in "Converting the Database".

6.13.1 Cleansing Binary Values in Character Columns

In the bottom-right of the Edit Data dialog box, you can recognize binary values stored incorrectly in a character column because such values show as garbage in the bottom left of the dialog box.

In the binary display, if the dialog box shows a lot of bytes in the ranges 0x00 to 0x08 and 0x10 to 0x1f, the value is binary. These bytes come up in character values very seldom because they correspond to ASCII control codes with only historical importance.

A binary value can also be recognized by a very long series of a single byte (other than the ASCII code for a space or the ASCII code for any character commonly used to draw separator lines, such as minus, plus, underscore, equal sign, or period).

To cleanse the invalid binary representation issues for a column storing binary values, migrate the data type of the column to RAW or LONG RAW. The DMU does not support migration to BLOB. The migration to RAW is supported for CHAR columns and for VARCHAR2 columns up to 2000 bytes in length. The migration to LONG RAW is supported for LONG columns.

To migrate a column to a binary type:

1. To migrate a column to a binary data type, right-click any cell of the column in the Cleansing Editor. The context menu is displayed.

2. In the menu, select either **Schedule Column Modification** or **Modify Column**. The former opens the Schedule Column Modification dialog box; the latter opens the Modify Column dialog box. The difference between these dialog boxes and the difference between associated cleansing modes – scheduled versus immediate – are described in "Lengthening a Column".

3. In the opened dialog box (see Figure 6-33), select **Migrate to**, and from the adjacent drop-down list, select the target data type. If the target data type is RAW, enter the desired length of the column in bytes. The length must be equal to or greater than the maximum byte length of any value in the column to be migrated.

4. Click **Apply**.

6.13.2 Cleansing Incorrect Character Set Declaration

If the binary view in the Edit Data dialog box does not reveal any byte patterns characteristic for binary data, then thei edited value is most probably character data in an incorrectly used character set.

If in doubt, consult application developers or application administrators and identify the source and type of information stored in the analyzed column.

To correct incorrect character set declarations:

1. To identify the actual character set of data in the column, select the column in the Cleansing Editor and repeatedly change the selected value in the Character Set drop-down list, until all values in the column are legible. See Figure 6-37. To save time, apply the knowledge gathered before starting the migration, as recommended in "Review Your Preparations for Migration", to restrict the character sets to only those that can possibly be used by any data source of the database (clients, input files).

   If none of the attempted character sets makes all values in the column appear to display correctly, but each of the values does seem to be correct in one of the character sets, the column contains a mixture of data in different character sets. Such issues must be resolved outside of the DMU.

2. After you identify the correct character set, assign it permanently to the column by clicking **Save** in the Cleansing Editor tab. Then, rescan the column. No more invalid representation issues should be reported for the column.

Figure 6-37   Assumed Column Character Set

6.13.3 Cleansing Corrupted Character Values

If only a very small number of values in a column are not legible in the assumed character set of the column, and if the characters illegible in one value are legible in another, then the invalid binary representation issue results from random corruption of the character values caused by a software or hardware malfunction.

In the case of software, applications that use single-byte algorithms to process multibyte data are a common reason for such corruption in multibyte databases. Such applications might split or truncate strings in the middle of a multibyte character, producing invalid partial codes.

To cleanse randomly corrupted values, edit and correct them in the Edit Data dialog box, as described in "Shortening Character Values Manually".

A

Third-Party License Information

Oracle Database Migration Assistant for Unicode contains third-party code.

Oracle is required to provide the following notices. Note, however, that the Oracle program license that accompanied this product determines your right to use the Oracle program, including the third-party software, and the terms contained in the following notices do not change those rights.

Apache POI 5.2.3

This program contains Apache POI 5.2.3, which has the following associated attribution text:

```
------------------------------ Top-level license ---------------------------
                             Apache License
                       Version 2.0, January 2004
                     http://www.apache.org/licenses/

   TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

   1. Definitions.

      "License" shall mean the terms and conditions for use, reproduction,
      and distribution as defined by Sections 1 through 9 of this document.

      "Licensor" shall mean the copyright owner or entity authorized by
      the copyright owner that is granting the License.

      "Legal Entity" shall mean the union of the acting entity and all
      other entities that control, are controlled by, or are under common
      control with that entity. For the purposes of this definition,
      "control" means (i) the power, direct or indirect, to cause the
      direction or management of such entity, whether by contract or
      otherwise, or (ii) ownership of fifty percent (50%) or more of the
      outstanding shares, or (iii) beneficial ownership of such entity.

      "You" (or "Your") shall mean an individual or Legal Entity
      exercising permissions granted by this License.

      "Source" form shall mean the preferred form for making modifications,
      including but not limited to software source code, documentation
      source, and configuration files.

      "Object" form shall mean any form resulting from mechanical
      transformation or translation of a Source form, including but
      not limited to compiled object code, generated documentation,
      and conversions to other media types.

      "Work" shall mean the work of authorship, whether in Source or
      Object form, made available under the License, as indicated by a
      copyright notice that is included in or attached to the work
```

(an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.

3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.

4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

(a) You must give any other recipients of the Work or
     Derivative Works a copy of this License; and

(b) You must cause any modified files to carry prominent notices
     stating that You changed the files; and

(c) You must retain, in the Source form of any Derivative Works
     that You distribute, all copyright, patent, trademark, and
     attribution notices from the Source form of the Work,
     excluding those notices that do not pertain to any part of
     the Derivative Works; and

(d) If the Work includes a "NOTICE" text file as part of its
     distribution, then any Derivative Works that You distribute must
     include a readable copy of the attribution notices contained
     within such NOTICE file, excluding those notices that do not
     pertain to any part of the Derivative Works, in at least one
     of the following places: within a NOTICE text file distributed
     as part of the Derivative Works; within the Source form or
     documentation, if provided along with the Derivative Works; or,
     within a display generated by the Derivative Works, if and
     wherever such third-party notices normally appear. The contents
     of the NOTICE file are for informational purposes only and
     do not modify the License. You may add Your own attribution
     notices within Derivative Works that You distribute, alongside
     or as an addendum to the NOTICE text from the Work, provided
     that such additional attribution notices cannot be construed
     as modifying the License.

You may add Your own copyright statement to Your modifications and
may provide additional or different license terms and conditions
for use, reproduction, or distribution of Your modifications, or
for any such Derivative Works as a whole, provided Your use,
reproduction, and distribution of the Work otherwise complies with
the conditions stated in this License.

5. Submission of Contributions. Unless You explicitly state otherwise,
   any Contribution intentionally submitted for inclusion in the Work
   by You to the Licensor shall be under the terms and conditions of
   this License, without any additional terms or conditions.
   Notwithstanding the above, nothing herein shall supersede or modify
   the terms of any separate license agreement you may have executed
   with Licensor regarding such Contributions.

6. Trademarks. This License does not grant permission to use the trade
   names, trademarks, service marks, or product names of the Licensor,
   except as required for reasonable and customary use in describing the
   origin of the Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty. Unless required by applicable law or
   agreed to in writing, Licensor provides the Work (and each
   Contributor provides its Contributions) on an "AS IS" BASIS,
   WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or
   implied, including, without limitation, any warranties or conditions
   of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A
   PARTICULAR PURPOSE. You are solely responsible for determining the

appropriateness of using or redistributing the Work and assume any
risks associated with Your exercise of permissions under this License.

8. Limitation of Liability. In no event and under no legal theory,
   whether in tort (including negligence), contract, or otherwise,
   unless required by applicable law (such as deliberate and grossly
   negligent acts) or agreed to in writing, shall any Contributor be
   liable to You for damages, including any direct, indirect, special,
   incidental, or consequential damages of any character arising as a
   result of this License or out of the use or inability to use the
   Work (including but not limited to damages for loss of goodwill,
   work stoppage, computer failure or malfunction, or any and all
   other commercial damages or losses), even if such Contributor
   has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability. While redistributing
   the Work or Derivative Works thereof, You may choose to offer,
   and charge a fee for, acceptance of support, warranty, indemnity,
   or other liability obligations and/or rights consistent with this
   License. However, in accepting such obligations, You may act only
   on Your own behalf and on Your sole responsibility, not on behalf
   of any other Contributor, and only if You agree to indemnify,
   defend, and hold each Contributor harmless for any liability
   incurred by, or claims asserted against, such Contributor by reason
   of your accepting any such warranty or additional liability.

END OF TERMS AND CONDITIONS

APPENDIX: How to apply the Apache License to your work.

   To apply the Apache License to your work, attach the following
   boilerplate notice, with the fields enclosed by brackets "[]"
   replaced with your own identifying information. (Don't include
   the brackets!) The text should be enclosed in the appropriate
   comment syntax for the file format. We also recommend that a
   file or class name and description of purpose be included on the
   same "printed page" as the copyright notice for easier
   identification within third-party archives.

Copyright [yyyy] [name of copyright owner]

Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.

APACHE POI SUBCOMPONENTS:

Apache POI includes subcomponents with separate copyright notices and license terms. Your use of these subcomponents is subject to the terms and conditions of the following licenses:

Office Open XML schemas (poi-ooxml-full-*.jar)

    The Office Open XML schema definitions used by Apache POI are
    a part of the Office Open XML ECMA Specification (ECMA-376, [1]).
    As defined in section 9.4 of the ECMA bylaws [2], this specification
    is available to all interested parties without restriction:

        9.4 All documents when approved shall be made available to
            all interested parties without restriction.

    Furthermore, both Microsoft and Adobe have granted patent licenses
    to this work [3,4,5].

    [1] http://www.ecma-international.org/publications/standards/Ecma-376.htm
    [2] http://www.ecma-international.org/memento/Ecmabylaws.htm
    [3] http://www.microsoft.com/openspecifications/en/us/programs/osp/
default.aspx
    [4] http://www.ecma-international.org/publications/files/ECMA-ST/
Ecma%20PATENT/
        Patent%20statements%20ok/
ECMA-376%20Edition%202%20Microsoft%20Patent%20Declaration.pdf
    [5] http://www.ecma-international.org/publications/files/ECMA-ST/
Ecma%20PATENT/
        Patent%20statements%20ok/ECMA-376%20Adobe%20Patent%20Declaration.pdf

Bouncy Castle library (bcprov-*.jar, bcpg-*.jar, bcpkix-*.jar)

    Copyright (c) 2000 - 2021 The Legion of the Bouncy Castle Inc.
    (https://www.bouncycastle.org)

    Permission is hereby granted, free of charge, to any person obtaining a
    copy of this software and associated documentation files (the "Software"),
    to deal in the Software without restriction, including without limitation
    the rights to use, copy, modify, merge, publish, distribute, sublicense,
    and/or sell copies of the Software, and to permit persons to whom the
    Software is furnished to do so, subject to the following conditions:

    The above copyright notice and this permission notice shall be included in
    all copies or substantial portions of the Software.

    THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
    IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
    FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL
    THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
    LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING
    FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER
    DEALINGS IN THE SOFTWARE.

CurvesAPI / Curve API

BSD License

Copyright (c) 2000-2015 www.hamcrest.org
All rights reserved.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions are
met:

Redistributions of source code must retain the above copyright notice,
this list of conditions and the following disclaimer. Redistributions in
binary form must reproduce the above copyright notice, this list of
conditions and the following disclaimer in the documentation and/or other
materials provided with the distribution. Neither the name of Hamcrest nor
the names of its contributors may be used to endorse or promote products
derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS
IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO,
THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR
CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO,
PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR
PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF
LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING
NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS
SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Log4j 2 library (log4j-api-*.jar)

Apache License Version 2.0

inbot-utils (https://github.com/Inbot/inbot-utils)

The MIT License (MIT)

Copyright (c) 2015 Inbot

Permission is hereby granted, free of charge, to any person obtaining a
copy
of this software and associated documentation files (the "Software"), to
deal
in the Software without restriction, including without limitation the
rights
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included
in all
copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,

.
.
.
----------------------- Copyright notices -------------------------
.

Apache POI
Copyright 2003-2021 The Apache Software Foundation

This product includes software developed at
The Apache Software Foundation (https://www.apache.org/).

This product contains parts that were originally based on software from BEA.
Copyright (c) 2000-2003, BEA Systems, <http://www.bea.com/> (dead link),
which was acquired by Oracle Corporation in 2008.
<http://www.oracle.com/us/corporate/Acquisitions/bea/index.html>
<https://en.wikipedia.org/wiki/BEA_Systems>

This product contains W3C XML Schema documents. Copyright 2001-2003 (c)
World Wide Web Consortium (Massachusetts Institute of Technology, European
Research Consortium for Informatics and Mathematics, Keio University)

This product contains the chunks_parse_cmds.tbl file from the vsdump program.
Copyright (C) 2006-2007 Valek Filippov (frob@df.ru)

This product contains parts of the eID Applet project
<http://eid-applet.googlecode.com> and <https://github.com/e-Contract/eid-applet>.
Copyright (c) 2009-2018
FedICT (federal ICT department of Belgium), e-Contract.be BVBA (https://www.e-contract.be),
Bart Hanssens from FedICT

=========================== Sub Components ============================
    poi-examples
    poi-excelant
    poi-javadoc
    poi-ooxml
    poi-ooxml-full
    poi-ooxml-lite
    poi-scratchpad

All subcomponents utilize the same license and copyright as the top level
component

--------------------- Fourth-party information ---------------------

```
== NAME OF DEPENDENCY
Commons-math
== License
Apache License - See above
== Copyright Notices
Apache Commons Math
Copyright 2001-2016 The Apache Software Foundation

This product includes software developed at
The Apache Software Foundation (http://www.apache.org/).

This product includes software developed for Orekit by
CS Systèmes d'Information (http://www.c-s.fr/)
Copyright 2010-2012 CS Systèmes d'Information
------------------------- Copyright notices -------------------------
-----------------------------(separator)-----------------------------
== NAME OF DEPENDENCY
xmlbeans
== License
Apache License - See above
== Copyright Notices
=========================================================================
==  NOTICE file corresponding to section 4(d) of the Apache License,   ==
==  Version 2.0, in this case for the Apache XmlBeans distribution.     ==
=========================================================================

   This product includes software developed at
   The Apache Software Foundation (http://www.apache.org/).

   Portions of this software were originally based on the following:
     - software copyright (c) 2000-2003, BEA Systems, <http://www.bea.com/>.

   Aside from contributions to the Apache XMLBeans project, this
   software also includes:

    - one or more source files from the Apache Xerces-J and Apache Axis
      products, Copyright (c) 1999-2003 Apache Software Foundation

    - W3C XML Schema documents Copyright 2001-2003 (c) World Wide Web
      Consortium (Massachusetts Institute of Technology, European Research
      Consortium for Informatics and Mathematics, Keio University)

    - resolver.jar from Apache Xml Commons project,
      Copyright (c) 2001-2003 Apache Software Foundation
```

Apache Mina SSHD 2.12.0

This program contains Apache Mina SSHD 2.12.0, which has the following associated attribution text:

```
Apache License
                        Version 2.0, January 2004
                     http://www.apache.org/licenses/

   TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION
```

1. Definitions.

   "License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

   "Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

   "Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

   "You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

   "Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

   "Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

   "Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

   "Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

   "Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

   "Contributor" shall mean Licensor and any individual or Legal Entity

on behalf of whom a Contribution has been received by Licensor and
subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of
   this License, each Contributor hereby grants to You a perpetual,
   worldwide, non-exclusive, no-charge, royalty-free, irrevocable
   copyright license to reproduce, prepare Derivative Works of,
   publicly display, publicly perform, sublicense, and distribute the
   Work and such Derivative Works in Source or Object form.

3. Grant of Patent License. Subject to the terms and conditions of
   this License, each Contributor hereby grants to You a perpetual,
   worldwide, non-exclusive, no-charge, royalty-free, irrevocable
   (except as stated in this section) patent license to make, have made,
   use, offer to sell, sell, import, and otherwise transfer the Work,
   where such license applies only to those patent claims licensable
   by such Contributor that are necessarily infringed by their
   Contribution(s) alone or by combination of their Contribution(s)
   with the Work to which such Contribution(s) was submitted. If You
   institute patent litigation against any entity (including a
   cross-claim or counterclaim in a lawsuit) alleging that the Work
   or a Contribution incorporated within the Work constitutes direct
   or contributory patent infringement, then any patent licenses
   granted to You under this License for that Work shall terminate
   as of the date such litigation is filed.

4. Redistribution. You may reproduce and distribute copies of the
   Work or Derivative Works thereof in any medium, with or without
   modifications, and in Source or Object form, provided that You
   meet the following conditions:

   (a) You must give any other recipients of the Work or
       Derivative Works a copy of this License; and

   (b) You must cause any modified files to carry prominent notices
       stating that You changed the files; and

   (c) You must retain, in the Source form of any Derivative Works
       that You distribute, all copyright, patent, trademark, and
       attribution notices from the Source form of the Work,
       excluding those notices that do not pertain to any part of
       the Derivative Works; and

   (d) If the Work includes a "NOTICE" text file as part of its
       distribution, then any Derivative Works that You distribute must
       include a readable copy of the attribution notices contained
       within such NOTICE file, excluding those notices that do not
       pertain to any part of the Derivative Works, in at least one
       of the following places: within a NOTICE text file distributed
       as part of the Derivative Works; within the Source form or
       documentation, if provided along with the Derivative Works; or,
       within a display generated by the Derivative Works, if and
       wherever such third-party notices normally appear. The contents
       of the NOTICE file are for informational purposes only and
       do not modify the License. You may add Your own attribution
       notices within Derivative Works that You distribute, alongside

or as an addendum to the NOTICE text from the Work, provided
that such additional attribution notices cannot be construed
as modifying the License.

You may add Your own copyright statement to Your modifications and
may provide additional or different license terms and conditions
for use, reproduction, or distribution of Your modifications, or
for any such Derivative Works as a whole, provided Your use,
reproduction, and distribution of the Work otherwise complies with
the conditions stated in this License.

5. Submission of Contributions. Unless You explicitly state otherwise,
   any Contribution intentionally submitted for inclusion in the Work
   by You to the Licensor shall be under the terms and conditions of
   this License, without any additional terms or conditions.
   Notwithstanding the above, nothing herein shall supersede or modify
   the terms of any separate license agreement you may have executed
   with Licensor regarding such Contributions.

6. Trademarks. This License does not grant permission to use the trade
   names, trademarks, service marks, or product names of the Licensor,
   except as required for reasonable and customary use in describing the
   origin of the Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty. Unless required by applicable law or
   agreed to in writing, Licensor provides the Work (and each
   Contributor provides its Contributions) on an "AS IS" BASIS,
   WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or
   implied, including, without limitation, any warranties or conditions
   of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A
   PARTICULAR PURPOSE. You are solely responsible for determining the
   appropriateness of using or redistributing the Work and assume any
   risks associated with Your exercise of permissions under this License.

8. Limitation of Liability. In no event and under no legal theory,
   whether in tort (including negligence), contract, or otherwise,
   unless required by applicable law (such as deliberate and grossly
   negligent acts) or agreed to in writing, shall any Contributor be
   liable to You for damages, including any direct, indirect, special,
   incidental, or consequential damages of any character arising as a
   result of this License or out of the use or inability to use the
   Work (including but not limited to damages for loss of goodwill,
   work stoppage, computer failure or malfunction, or any and all
   other commercial damages or losses), even if such Contributor
   has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability. While redistributing
   the Work or Derivative Works thereof, You may choose to offer,
   and charge a fee for, acceptance of support, warranty, indemnity,
   or other liability obligations and/or rights consistent with this
   License. However, in accepting such obligations, You may act only
   on Your own behalf and on Your sole responsibility, not on behalf
   of any other Contributor, and only if You agree to indemnify,
   defend, and hold each Contributor harmless for any liability
   incurred by, or claims asserted against, such Contributor by reason
   of your accepting any such warranty or additional liability.

```
   END OF TERMS AND CONDITIONS

   APPENDIX: How to apply the Apache License to your work.

      To apply the Apache License to your work, attach the following
      boilerplate notice, with the fields enclosed by brackets "[]"
      replaced with your own identifying information. (Don't include
      the brackets!)  The text should be enclosed in the appropriate
      comment syntax for the file format. We also recommend that a
      file or class name and description of purpose be included on the
      same "printed page" as the copyright notice for easier
      identification within third-party archives.

   Copyright [yyyy] [name of copyright owner]

   Licensed under the Apache License, Version 2.0 (the "License");
   you may not use this file except in compliance with the License.
   You may obtain a copy of the License at

       http://www.apache.org/licenses/LICENSE-2.0

   Unless required by applicable law or agreed to in writing, software
   distributed under the License is distributed on an "AS IS" BASIS,
   WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
   See the License for the specific language governing permissions and
   limitations under the License.

=========CopyRight            ===========
Apache MINA SSHD
Copyright 2008-2021 The Apache Software Foundation

This product includes software developed at
The Apache Software Foundation (http://www.apache.org/).

Please refer to each LICENSE.<component>.txt file for the
license terms of the components that Apache MINA depends on.


========= 4th party Dependencies ==========


Dependency #1

NAME: SLF4J-API


===License====

Name: MIT

All rights reserved.

Permission is hereby granted, free  of charge, to any person obtaining
a  copy  of this  software  and  associated  documentation files  (the
"Software"), to  deal in  the Software without  restriction, including
without limitation  the rights to  use, copy, modify,  merge, publish,
```

distribute, sublicense, and/or sell copies of the Software, and to
permit persons to whom the Software is furnished to do so, subject to
the following conditions:

The above copyright notice and this permission notice shall be
included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND,
EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF
MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND
NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE
LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION
OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION
WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
===CopyRight====
Copyright (c) 2004-2022 QOS.ch Sarl (Switzerland)
Copyright (c) 2004-2023 QOS.ch

Dependency #2

NAME: JCL-OVER-SLF4J

===License====

                           Apache License
                     Version 2.0, January 2004
                  http://www.apache.org/licenses/

    TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

    1. Definitions.

       "License" shall mean the terms and conditions for use, reproduction,
       and distribution as defined by Sections 1 through 9 of this document.

       "Licensor" shall mean the copyright owner or entity authorized by
       the copyright owner that is granting the License.

       "Legal Entity" shall mean the union of the acting entity and all
       other entities that control, are controlled by, or are under common
       control with that entity. For the purposes of this definition,
       "control" means (i) the power, direct or indirect, to cause the
       direction or management of such entity, whether by contract or
       otherwise, or (ii) ownership of fifty percent (50%) or more of the
       outstanding shares, or (iii) beneficial ownership of such entity.

       "You" (or "Your") shall mean an individual or Legal Entity
       exercising permissions granted by this License.

       "Source" form shall mean the preferred form for making modifications,
       including but not limited to software source code, documentation
       source, and configuration files.

       "Object" form shall mean any form resulting from mechanical
       transformation or translation of a Source form, including but
       not limited to compiled object code, generated documentation,
       and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or
Object form, made available under the License, as indicated by a
copyright notice that is included in or attached to the work
(an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object
form, that is based on (or derived from) the Work and for which the
editorial revisions, annotations, elaborations, or other modifications
represent, as a whole, an original work of authorship. For the purposes
of this License, Derivative Works shall not include works that remain
separable from, or merely link (or bind by name) to the interfaces of,
the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including
the original version of the Work and any modifications or additions
to that Work or Derivative Works thereof, that is intentionally
submitted to Licensor for inclusion in the Work by the copyright owner
or by an individual or Legal Entity authorized to submit on behalf of
the copyright owner. For the purposes of this definition, "submitted"
means any form of electronic, verbal, or written communication sent
to the Licensor or its representatives, including but not limited to
communication on electronic mailing lists, source code control systems,
and issue tracking systems that are managed by, or on behalf of, the
Licensor for the purpose of discussing and improving the Work, but
excluding communication that is conspicuously marked or otherwise
designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity
on behalf of whom a Contribution has been received by Licensor and
subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of
   this License, each Contributor hereby grants to You a perpetual,
   worldwide, non-exclusive, no-charge, royalty-free, irrevocable
   copyright license to reproduce, prepare Derivative Works of,
   publicly display, publicly perform, sublicense, and distribute the
   Work and such Derivative Works in Source or Object form.

3. Grant of Patent License. Subject to the terms and conditions of
   this License, each Contributor hereby grants to You a perpetual,
   worldwide, non-exclusive, no-charge, royalty-free, irrevocable
   (except as stated in this section) patent license to make, have made,
   use, offer to sell, sell, import, and otherwise transfer the Work,
   where such license applies only to those patent claims licensable
   by such Contributor that are necessarily infringed by their
   Contribution(s) alone or by combination of their Contribution(s)
   with the Work to which such Contribution(s) was submitted. If You
   institute patent litigation against any entity (including a
   cross-claim or counterclaim in a lawsuit) alleging that the Work
   or a Contribution incorporated within the Work constitutes direct
   or contributory patent infringement, then any patent licenses
   granted to You under this License for that Work shall terminate
   as of the date such litigation is filed.

4. Redistribution. You may reproduce and distribute copies of the

Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

(a) You must give any other recipients of the Work or Derivative Works a copy of this License; and

(b) You must cause any modified files to carry prominent notices stating that You changed the files; and

(c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and

(d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS,

WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or
implied, including, without limitation, any warranties or conditions
of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A
PARTICULAR PURPOSE. You are solely responsible for determining the
appropriateness of using or redistributing the Work and assume any
risks associated with Your exercise of permissions under this License.

8. Limitation of Liability. In no event and under no legal theory,
   whether in tort (including negligence), contract, or otherwise,
   unless required by applicable law (such as deliberate and grossly
   negligent acts) or agreed to in writing, shall any Contributor be
   liable to You for damages, including any direct, indirect, special,
   incidental, or consequential damages of any character arising as a
   result of this License or out of the use or inability to use the
   Work (including but not limited to damages for loss of goodwill,
   work stoppage, computer failure or malfunction, or any and all
   other commercial damages or losses), even if such Contributor
   has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability. While redistributing
   the Work or Derivative Works thereof, You may choose to offer,
   and charge a fee for, acceptance of support, warranty, indemnity,
   or other liability obligations and/or rights consistent with this
   License. However, in accepting such obligations, You may act only
   on Your own behalf and on Your sole responsibility, not on behalf
   of any other Contributor, and only if You agree to indemnify,
   defend, and hold each Contributor harmless for any liability
   incurred by, or claims asserted against, such Contributor by reason
   of your accepting any such warranty or additional liability.

   END OF TERMS AND CONDITIONS

=====Copyright =========

Copyright (c) 2004-2022 QOS.ch Sarl (Switzerland)

Glossary

A to B conversion

A type of character set migration. In an A to B conversion, you create a new Unicode database and move all data from the existing database to the new one using the Export and Import utilities or Data Pump.

See also inline migration.

Bulk Cleansing Wizard

A wizard that enables you to perform three things in bulk:

- alter the length semantics of `CHAR` and `VARCHAR2` table columns from byte semantics to character semantics in bulk

  You can perform the bulk cleansing for migrating to character length semantics in either immediate or scheduled cleansing mode. In the scheduled mode, the character length semantics migration will take place during the conversion phase without any interruption to the production environment. When the DMU is used in validation mode, only the immediate cleansing option will be available

- Set the "Allow Conversion of Data with Issues" column conversion property

  If the scan results of the column show that some values have conversion issues, that is, the converted values will contain replacement characters or they will be truncated, you can still let the DMU convert the data by setting this property to Yes. This might be useful if you want to automatically truncate data that exceeds the column or data type limit or you are not concerned about a few corrupted values that happen to exist in the column but are of no real significance for applications that use this data.

- Replace or remove occurrences of a byte or character pattern in column values

  In some cases, after analyzing the nature of individual data exceptions, you may decide to modify the data by replacing the offending bytes or characters with a set of valid equivalent bytes or characters or removing them altogether. If the same problem symptom exists in multiple database objects, it may be desirable to perform such replacement operation in bulk. The pattern-based replacement cleansing option allows you to replace all occurrences of a source pattern with a target pattern for the selected database objects. You can perform the replacement for single bytes or characters or specify substrings of bytes or characters.

character semantics

Character semantics is useful for defining the storage requirements for multibyte strings of varying widths. For example, in a Unicode database (AL32UTF8), suppose that you need to

define a `VARCHAR2` column that can store up to five Chinese characters along with five English characters. Using byte semantics, this column requires 15 bytes for the Chinese characters, which are three bytes long, and 5 bytes for the English characters, which are one byte long, for a total of 20 bytes. Using character semantics, the column requires 10 characters. Character semantics allows the specification of the column size in terms of characters independent of the database character set.

character set migration

The process of moving from one database character set to another. In this guide, we discuss moving to Unicode.

Check for Updates Wizard

You can check for and install the latest version of the DMU by using the DMU live update feature. This is done with a Check for Updates Wizard, and it enables you to find out if a new version of the DMU is available from the DMU update center.

cleansing

The process of identifying and resolving the data issues that were found in the scanning process. These issues must be resolved before the database can be converted

CSALTER script

The `CSALTER` script is part of the Database Character Set Scanner utility. It is the most straightforward way to migrate a character set, but it can be used only if all of the schema data is a strict subset of the new character set.

CSREPAIR script

The `CSREPAIR` script enables you to repair the database character set metadata when using the pass-through configuration. In the pass-through configuration, the client character set is defined (usually through the `NLS_LANG` client setting) to be equal to the database character set, though the character data in your database could be stored in a different character set from the declared database character set. The `CSREPAIR` script works with the DMU client by accessing the DMU repository. Note that it only changes the database character set declaration in the data dictionary metadata and does not convert any database data.

CSSCAN script

The Database Character Set Scanner (`CSSCAN`) is a command-line utility that assesses the feasibility of migrating an Oracle database to a new database character set. The Database Character Set Scanner checks all character data in the database and tests for the effects and problems of changing the character set encoding. A summary report is generated at the end of the scan that shows the scope of work required to convert the database to a new character set.

Database Migration Assistant for Unicode (DMU)

An intuitive and user-friendly GUI tool to migrate your character set. It helps you streamline the migration process through an interface that minimizes the workload and ensures that all migration issues are addressed.

Database Scan Report

The Database Scan Report contains the output from a scan of the database in a form that enables you to view results in many different ways. You can customize this report as well as filter by the status of the data. As an example, you can filter the report to only display all objects that contain some data that exceeds its data type limit after conversion.

Diagnostic Package Creation Wizard

A wizard that enables you to create a diagnostic package for later analysis by Oracle to help resolve any problems that occurred while using the DMU.

DMU repository

A repository that contains the objects to be processed, details on data that had an error flagged, the progress of a scan or conversion, and some other items necessary to migrate to Unicode.

expansion

When you migrate from a legacy encoding to Unicode, character values will likely expand in conversion because their encodings will have more bytes. This results in increased space requirements for the database. A further issue is that the widths for `CHAR` and `VARCHAR2` columns may not be sufficient after the character set has been migrated to Unicode. Thus, there is a risk of the data being truncated. The column length constraints have to be increased, or, if they are already at the data type limit, the columns might need to be migrated to the `CLOB` data type.

external tables

External tables do not usually require conversion. If the character set of the source file for an external table (SQL*Loader or Data Pump) is properly declared, the database will automatically convert character data to the new database character set each time the data is fetched from the file. For performance reasons, consider converting the file permanently to avoid this conversion.

inline migration

A type of migration where you identify non-ASCII data first, and then only that data and its metadata is updated during the conversion phase of the migration. The amount of modified database values is usually only a small percentage of all values stored in the database.

See also A to B conversion.

initialization

The process of installing required patches and supporting packages, as well as creating a tablespace for the DMU repository, a database connection, and then installing the DMU repository.

invalid binary representation

A common problem with user data is that the data in a column may not actually be in the declared database character set. Instead, it may be in another character set or it is binary (for example, if it is composed of images, documents in proprietary word processor formats, or text encrypted with custom methods), or perhaps there are multiple character sets in a single column. This problem is possible in a pass-through configuration, which has the client character set defined (usually through the `NLS_LANG` client setting) to be equal to the database character set.

migration

See character set migration.

migration workflow

The typical steps, and their order, for migrating the database character set to Unicode.

pass-through configuration

In a pass-through configuration, client character set is defined (usually through the `NLS_LANG` client setting) to be equal to the database character set. Because the character sets are equal, the Oracle communication protocol does not attempt any client/server character set conversion, logically assuming that the conversion is not needed. This configuration is correct as long as data coming from a client application is indeed in the declared character set. But because no character set conversion nor character validation is performed on the data, any sequence of bytes can actually be stored in the database and retrieved unchanged. As long as only the client application interprets the data, the stored byte sequences could correspond to a character encoding different from the database character set or they may not correspond to readable text at all.

preparation

Data preparation is the process to ensure that no database data to be migrated will cause problems during or after the actual conversion. The elements of data preparation are scanning and cleansing.

read-only objects

Tables and tablespaces that are read-only cannot be converted unless they are made read-write. Certain tables are read-only, such as tables explicitly marked as read-only, external

tables, and tables stored in read-only tablespaces. Such tables cannot be converted before they are made read-write. Except for the explicit marking of a table as read-only, which the migration process can temporarily remove, the other mentioned reasons have to be dealt with manually.

Repository Configuration Wizard

A wizard that enables you to manage the information necessary for each step of the migration through the use of a repository called as DMU repository. For any database without a DMU repository, a repository is automatically created each time you connect to that database. You can also start the wizard from the DMU user interface.

See also DMU repository.

scanning

Scanning is the process of assessing the feasibility of migrating the data to Unicode. As part of this process, the DMU reads character values from the database, converts them to the target character set, and counts how many values change in conversion, do not fit into their columns, do not fit into their data types, or contain invalid character codes. Additional statistics, such as the maximum post-conversion length of values in a column, are calculated as well.

Scan Wizard

A wizard that guides you through the scanning process, and informs you if any data needs cleansing before it can be converted permanently without data loss.

subset

A database character set that is included within a superset character set. An example of this is ASCII being contained within Unicode.

superset

A database character set that includes the subset character set. An example of this is that Unicode contains the subset of ASCII.

truncation

To shorten something by cutting off or removing a part. In the context of data migration, the important consideration is that there is a risk of data being truncated when `CHAR` and `VARCHAR2` columns are not sufficiently wide after the migration.

Unicode UTF-8

UTF8 (CESU-8) is a varying-width character set, where a character can be 1-3 bytes, and supplementary characters are represented by a pair of 3 bytes.

Unicode AL32UTF8

AL32UTF8 is a varying-width character set, which means that a character can be 1, 2, 3, or 4 bytes long, and supplementary characters are represented as 4 bytes.

validation

The DMU can validate the contents of an existing AL32UTF8 or UTF8 database. Such a database might have been converted to Unicode in the past or initially created in Unicode. In either case, you can check the data for current problems.

Index

ORACLE

ORACLE