Oracle® Database

Migration Assistant for Unicode Guide

Release 1.0.2

E14853-02

February 2011

ORACLE®

Oracle Database Migration Assistant for Unicode Guide, Release 1.0.2

E14853-02

Primary Author: Paul Lane

Contributor: Sergiusz Wolicki, Weiran Zhang

Contents

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.

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

Our goal is to make Oracle products, services, and supporting documentation accessible to all users, including users that are disabled. To that end, our documentation includes features that make information available to users of assistive technology. This documentation is available in HTML format, and contains markup to facilitate access by the disabled community. Accessibility standards will continue to evolve over time, and Oracle is actively engaged with other market-leading technology vendors to address technical obstacles so that our documentation can be accessible to all of our customers. For more information, visit the Oracle Accessibility Program Web site at http://www.oracle.com/accessibility/.

Accessibility of Code Examples in Documentation

Screen readers may not always correctly read the code examples in this document. The conventions for writing code require that closing braces should appear on an otherwise empty line; however, some screen readers may not always read a line of text that consists solely of a bracket or brace.

Accessibility of Links to External Web Sites in Documentation

This documentation may contain links to Web sites of other companies or organizations that Oracle does not own or control. Oracle neither evaluates nor makes any representations regarding the accessibility of these Web sites.

Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/support/contact.html or visit http://www.oracle.com/accessibility/support.html if you are hearing impaired.

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.

Printed documentation is available for sale in the Oracle Store at

`http://oracle.com/store`

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

`http://www.oracle.com/technetwork`

1

Overview of Migrating Databases to Unicode

This chapter discusses the database character set migration process. First, it explains what a database character set is and what it means to migrate one. It discusses the basic issues related to the migration process and the tools available to help with those tasks.

This chapter contains the following sections:

What Is a Database Character Set?

In Oracle, a database character set defines a set of characters that can be stored in the database and defines how each character maps to a particular sequence of bytes stored in memory or disk to represent this character. What Oracle calls a character set is also known in the industry as a character encoding, which can be thought of as a set of characters plus their encoding in bytes.

A database character set is used for `VARCHAR2`, `CHAR`, `LONG`, and `CLOB` data, as well as for 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 (that is, Unicode UTF-16 encoding in big-endian form, which is abbreviated as UTF-16BE).

A database character set may be single-byte, where each character has one byte, or multibyte varying width, where each character may be 1, 2, 3, or 4 bytes with the maximum dependent on the particular character set. Single-byte character sets are easier and faster to process but have a very limited capability of 256 codes -- too few even for all European languages. Multibyte character sets require more complex string processing but can accommodate thousands of characters.

Almost all character sets other than Unicode (frequently called legacy character sets) were each designed for a specific language or group of languages. For single-byte character sets, this was logical because of the limited number of available codes. For multibyte character sets, this was presumably 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.

An important concept when working with character sets is that of subsets and supersets. We say that character set A is a **superset** of character set B and that character set B is a **subset** of the character set A, if A contains or defines all characters of B plus some other characters. We say that character set A is a strict or binary superset of character set B and that character set B is a strict or binary subset of the character set A, if A contains or defines all characters of B and each such character has identical byte representation in both A and B. 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 the process of character set conversion. That is, the code of a character in the source character set must be replaced with the code of the same character in the target character set. Characters that are not available in the target character set are converted to replacement characters. The replacement character 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, 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 per definition, 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, so that if the text to be converted contains byte sequences that do not form valid character codes in the source character set, these sequences are converted to the replacement characters in the target character set.

Introduction to Character Set Migration

A database defined in a legacy character set can store data only in one or a few languages as determined by its character set. If a need arises to store data in other languages in this database, the character set of the database must be changed to a superset that defines both the existing and the new required characters. The need may come from internationalization of operations, moving to an Internet sales model, or from consolidating databases of international subsidiaries or buying and acquired companies into one database.

If the identified superset is also a binary superset, the change of the database character set declaration must happen only in metadata (schema information) in the data dictionary. All existing data, per definition, remains valid in the binary superset. The change is simple and fast. If the identified superset is not a binary superset, binary character codes (byte representation) must be converted from the representation in the old character set to the representation in the new character set, that is, re-encoded.

An important observation, relevant to determining if existing data requires conversion, is that even if the old database character set is not a binary subset of the new database character set, many characters may actually have the same binary codes in both character sets. In particular, all standard ASCII characters with codes between 0 and 127 form a binary subset of each Oracle database character sets supported on ASCII-based platforms (that is, all platforms except EBCDIC-based IBM z/OS and Fujitsu BS2000). Therefore, even if the planned database character set change is, for example, from WE8ISO8859P1 to AL32UTF8 - which are not in a binary subset-superset relationship - but all characters that are in the database have codes in

the range 0 to 127, then no characters really need to be converted, because all characters will have the same codes after the change.

If character data in a database to be converted can be classified into data that needs no conversion, called **changeless** throughout this guide, and data that requires conversion, called **convertible**, a lot or resources can be saved by skipping the conversion of the changeless data.

When the character codes are converted, various issues may arise. The most common one is that 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, those issues have to be identified and cleaned up first, for example, by lengthening columns, shortening texts, or migrating to a larger data type.

The process of identifying the suitable superset, classifying character data as changeless or convertible, checking for issues and conversion requirements, fixing the issues, changing the declaration in the data dictionary and converting data as needed is called character set migration. The issues and the subset/superset relationship of existing data in the database may be identified with the help of the Database Character Set Scanner utility, discussed in *Oracle Database Globalization Support Guide*, or with the help of the Database Migration Assistant for Unicode.

The migration process can be performed as an **A-to-B migration**, when data is copied from a source database into a new database defined with the target character set, or as an **inline migration**, when data is converted inside a database and the character set information of the database is changed.

The character set migration should not be confused with 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 goal of the repair process is to match the database character set declaration with the expectations of the applications. Such repair processes, per definition, do not involve conversion of character codes.

Why Unicode Is the Right Choice

An important step in the character set migration process is to gather the requirements for what you need to achieve. For example, a very common situation is that your company is internationalizing its operation and you now must support data storage of more world languages than are available with your existing database character set. Because many legacy computer systems initially required support for only one or possibly a few languages, the original character set chosen might have had a limited repertoire of characters that could be supported. A common case was in America, where 7-bit ASCII was satisfactory for supporting English data exclusively. Another common situation occurred in 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, however, these choices are 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, you would logically choose Unicode as the 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 various symbols, for example, those used in technical, scientific, and musical notations. It is the

native or recommended character set of many technologies, such as Java, Windows, HTML, or XML. There is no other character set that is so universal. In addition, Unicode adoption 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 and CESU-8 through the UTF8 character set. (Note the use of the hyphen in the name of Unicode encoding and the lack of a hyphen in the name of Oracle character set. This differentiation is used throughout Oracle documentation.) UTF-8 is a multibyte varying width Unicode encoding using 1 to 4 bytes per character. CESU-8 is a compatibility-only encoding, discouraged for information exchange by the Unicode standard. CESU-8 is very 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 11*i*.

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).

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, Unicode is the only choice.

Oracle recommends migrating to Unicode for its universality and compatibility with contemporary and future technologies and language requirements. 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 *Oracle Database Globalization Support Guide* for more coverage of NCHAR.

About Character Set Migration Tools

You can change the database character set of an existing database. The following sections discuss how to achieve this:

- Identifying Migration Issues with the Database Character Set Scanner
- Changing Character Set Metadata Using the CSALTER Script
- Converting Character Data Using the Export/Import and Data Pump Utilities
- Migrating a Database Using the Database Migration Assistant for Unicode

Identifying Migration Issues with the Database Character Set Scanner

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.

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 database character set declaration is being repaired (see "Introduction to Character Set Migration" on page 1-2).
- The scan results are required by the CSALTER script.

> **See Also:** *Oracle Database Globalization Support Guide* for more information about the Database Character Set Scanner

Changing Character Set Metadata Using the CSALTER Script

The CSALTER script is part of the Database Character Set Scanner utility. The script has two tasks:

1. Change the character set metadata information stored in various system tables in the data dictionary to the new character set.
2. Convert character data in the CLOB columns in the data dictionary tables, if the database character set is changed from single-byte to multibyte or from multibyte to single-byte.

The CSALTER script regards more than just tables owned by SYS as the data dictionary. It considers most of the schemas present in default template databases, which are created during Oracle Database software installation, as the data dictionary as well.

The CSALTER script can be run only if none of the character data in the database, with the exception of 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, the CSALTER script requires that the Database Character Set Scanner be run first to confirm that no data requires conversion, that is, the target character set is a binary superset of the data in the database. CSALTER aborts with an error message if the scan results contain information about any "exceptional" data anywhere in the database or any "convertible" data outside of the data dictionary CLOB columns.

The CLOB columns in the data dictionary are treated especially because AL16UTF16, which is used to store CLOB data in multibyte databases, is never a binary superset or subset of any single-byte character set. So even if the target character set is a superset, the CLOBs still need to be converted. As data dictionary tables cannot be dropped or truncated, no single-byte database character set could ever be changed to any multibyte character set without CSALTER converting the data dictionary CLOBs.

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 of the CSALTER functionality has been included in the Database Migration Assistant for Unicode, discussed in this guide, but CSALTER must still be used, if:

- migration is to a legacy character set, or
- the national (NCHAR) character set is being migrated, or
- the database character set declaration is being repaired (see "Introduction to Character Set Migration" on page 1-2)

> **See Also:** *Oracle Database Globalization Support Guide* for information about how to change the database character set using the CSALTER script

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

If the Database Character Set Scanner finds some data requiring conversion in the database, the CSALTER script cannot change the database character set. Some other method must also be applied to convert the data.

One of the common ways to convert the character data is to use the Export utility to export database objects from the database and then the Import utility to import these objects either to the same or to another database. If the database character set at the time of the export differs from the database character set at the time of the import, the Import utility automatically converts the data as required.

The advantage of using the Export and Import utilities to convert data compared to other methods, such as spooling data and loading it back with SQL*Loader, is that

they automatically take 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

Full Versus Selective Export/Import

Export and import may be applied as steps in the following two methods of the character set migration:

- Migration through full database export and import

  In this method, an empty database in the target character set is created first. Then, all contents of the database is exported from the source database and imported into the target database. The created copy of the original database has all character data converted as required. The source database is dropped and the copy becomes the production database.

- Migration through selective export and import

  In this method, all tables identified by the Database Character Set Scanner utility as containing convertible data – and only these tables – are exported, and then truncated. The Database Character Set Scanner is run again and this time, it finds no convertible data. The `CSALTER` script can now be executed to change the database character set. Then, the previously exported data is imported back into the database, undergoing the required character set conversion.

Circumstances that favor the full export/import method are:

- 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 point to the selective export/import method instead:

- The full export/import converts all text in the database, more than may be needed; it also needs to 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.

Data Pump Utility

The Export and Import utilities come in two versions, the original Export (`exp`) and Import (`imp`) and the Data Pump Export (`expdp`) and Import (`impdp`). The Data Pump versions were introduced in Oracle Database Release 10*g*. The original versions

are desupported for general use as of Oracle Database Release 11*g*. Both versions are command-line tools and both can be used for the purpose discussed here.

The original utilities are not compatible with Data Pump utilities. Files produced by `exp` cannot be imported by `impdp` and files produced by `expdp` cannot be imported by `imp`.

The original Export and Import are pure client-side utilities. The Export utility retrieves data over an Oracle Net network connection and creates the export file on the client. Data Pump Export contains a small client-control utility, but the actual export job is performed by a database server process and the export file is produced on the database server host. Similarly, the original Import utility reads the export file on the client and inserts data over an Oracle Net network connection into the target database, while the Data Pump utility reads the export file directly on the database server host. The Data Pump architecture makes the Data Pump utility export and import process much faster, but if the export file is to be created or read on a computer other than the source database host, 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

Conversion Issues

Conversion, as mentioned in "Introduction to Character Set Migration" on page 1-2 and described in "Character Set Migration: Data Integrity" on page 1-9, 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 methods discusses, the import process might fail.

When using the Export/Import utilities, restrictions regarding objects not supported by the given utility version must be considered, as well. For example, Oracle Data Pump Release 10*g* does not support exporting an XML schema. Such objects might need to be moved manually.

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

Migrating a Database Using the Database Migration Assistant for Unicode

The DMU offers an intuitive and user-friendly GUI that helps you streamline the migration process through an interface that minimizes the workload and ensures that all migration issues are addressed, along with guaranteeing that the data conversion is carried out correctly and efficiently.

Some advantages of the DMU are that it does 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:** Chapter 4, "Performing Basic DMU Tasks" for more information

Overview of Character Set Migration Considerations

The following sections discuss the matters you should consider before migrating your current character set using the Database Migration Assistant for Unicode:

- Character Set Migration: Data Integrity
- Character Set Migration: Dependent Objects
- Character Set Migration: Read-Only and Inaccessible Objects
- Character Set Migration: Downtime
- Character Set Migration: Failure Recovery
- Character Set Migration: Application Impact

Character Set Migration: Data Integrity

There are several data integrity issues that might arise from re-encoding of character data in the database, including the following:

- Data Expansion
- Invalid Binary Storage Representation of Data
- Partitioning
- Maximum Index Key Size
- Unique Keys and Primary Keys
- Derived or Encrypted Data
- Character Data Stored in Binary Data Types

Data 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.

A related issue is with non-ASCII object names, which typically have a 30-byte limit. The names of such objects must be manually truncated or changed to use only ASCII characters, in both the database and affected applications, because it is impossible to increase the limit of non-ASCII object names.

Invalid Binary Storage Representation of Data

For user data, a common problem is that the data in a column is not actually in the declared database character set. Instead, it is in another character set or it is binary (for example, 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.

The preceding situation is possible in the *pass-through* configuration. In this 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.

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, 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.

Partitioning

Partitioning by range compares partitioning key values with partition boundary values to distribute rows into partitions. This comparison looks at the binary storage

representation of character values. The binary storage representation of both the keys and the boundary values could change when converted to Unicode in a way that would not preserve their ordering. This 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.

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

Maximum Index Key Size

The key size of an index is calculated using maximum lengths of key columns. The maximum supported index key size is limited by the database block size. To ensure that expanded column data can be held in a given column after conversion to Unicode, users may try to expand the column byte length or alter the column length semantics from `BYTE` to `CHAR`. If an index was created on such a column, the resulting larger index key might exceed the supported maximum index key size.

You can try to solve this problem by:

- Increasing the block size
- Dropping the index
- Removing some columns from a composite index

Unique Keys and Primary Keys

There are two situations where uniqueness of key values of a unique constraint or a primary key constraint might be violated after conversion of the key values to Unicode, even though the pre-conversion key values satisfy the constraint:

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

Derived or Encrypted Data

Applications might store information that has been derived from character data in a way that depends on the binary storage representation of this data. For example, an application might store text length separately from the text in a numerical column. If the text expands, such a column will not be updated automatically. Or, an application might calculate and store a checksum or an encrypted form of a character value. The

checksum or cipher text, if based on the binary storage representation in the database character set, becomes invalid when the representation changes because of character set conversion.

Such derived values must be resynchronized after the migration. For encrypted text, this usually means that the text must be decrypted just before the conversion, then converted, and then immediately re-encrypted. For security reasons, the time during which the text remains decrypted must be kept to minimum and the database server might require extra protection to prevent a security breach.

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

Character Data Stored in Binary Data Types

Applications might store character data in binary data types, such as `RAW`, `LONG RAW`, and `BLOB`. The applications might use the database character set to interpret and process this data. If the database character set is migrated but the character data in the binary data types is not re-encoded to match the new database character set, the applications will fail. Automatic migration tools, such as the Database Migration Assistant for Unicode described in this guide, 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. Therefore, if required, you must manually locate and convert any character data in binary data types.

Character Set Migration: Dependent Objects

When data in a table is modified during the conversion process, the modification impacts all objects that store a copy of this data, such as indexes (standard, functional, and domain) and materialized views. Depending on the conversion method, these objects might need to be dropped and re-created or they will be synchronized automatically by the database. For performance reasons, it might be more efficient to drop the dependent objects and re-create them after migration, instead of relying on the automatic synchronization.

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

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

Character Set Migration: Read-Only and Inaccessible Objects

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.

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. If the character set of the source file is not defined correctly, and 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" on page 1-10, the character set declaration of the file must be corrected. In the case of Data Pump files, this is not a trivial task.

Read-only tablespaces might be stored on read-only media, such as CD-ROM or DVD-ROM. If convertible data is present in such a tablespace, you must move the tablespace to standard disk storage, make it read/write, convert the database, make the tablespace read-only again, and copy it back to the read-only media. If there are many such tablespaces in a database, for example, used for archival, and making them all read/write before the conversion is not feasible due to disk storage requirements, consider creating a dedicated archival database in the old database character set, and moving the read-only tablespaces one-by-one to this new database using the transportable tablespace feature. Then, the original database can be migrated. The read-only data in the archival database can be accessed through a database link, which will take care of the necessary character set conversion when the data is retrieved.

Tables can be stored in tablespaces or data files that have been put into the offline state. Data in such tables are not accessible. You can neither scan the data for migration issues nor actually convert it. Offline tablespaces and data files containing tables with character columns must be brought online before the character set migration process can begin.

Character Set Migration: Downtime

The downtime window is the time when all applications accessing the production database are shut down and the database is accessible only to migration tools. A downtime window is required to perform the actual conversion of the database. Scanning for issues and cleansing them may be performed, to a great extent, parallel to standard production work, though preferably during off-peak hours.

Keep the downtime windows as short as possible during the migration process. In addition to the actual time that the migration takes, 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 actual conversion of the database. If the last scan is performed in a production database before the downtime window starts, the downtime window is smaller but 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, which enables you to scan only those tables in the downtime window for which the risk of introducing new issues is significant.

Character Set Migration: Failure Recovery

When you perform a migration, the process may abruptly terminate for many reasons, such as a software defect or hardware malfunction. In that case, the database could be left in an inconsistent state. If the migration utility is not able to resume the migration, you may need to bring your database back to a consistent state, usually by recovering from a backup or using the Flashback Database feature. Dealing with failures during the migration process is a necessary part of your migration planning.

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

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 impact on applications using this database. The following could affect the application:

- If table columns are lengthened or migrated to other data types to solve length expansion issues, buffer variables might need to be enlarged in the application code and application logic might need to be adapted 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, it might be necessary to adapt single-byte applications to process multibyte text correctly. This is not a trivial change.

All the preceding application issues must usually be handled manually. Application developers have to 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.

This chapter contains the following sections:

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" on page 2-2, "Overview of Java Runtime Requirements" on page 2-3, and "Overview of DMU Security Considerations" on page 2-3.

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" on page 2-6, "Creating a Database Connection" on page 2-6, "Installing the Migration Repository" on page 2-8, and "Introduction to the DMU User Interface" on page 2-13.

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

   See "Scanning the Database" on page 4-4.

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

See "Cleansing the Data" on page 4-21.

5. Convert the database to Unicode.

   See "Converting the Database" on page 4-21.

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

   See "Validating Data as Unicode" on page 4-29.

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" on page 2-13.

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.

Overview of Requirements and Security Considerations

This section describes requirements and restrictions you should consider when using the Database Migration Assistant for Unicode, as well as the first steps you must take to start the utility.

Overview of Database Requirements

A database must meet certain requirements to be supported by the DMU. These requirements are:

- 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, so, 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. It must be created manually by running the script `?/rdbms/admin/prvtdumi.plb` from the Oracle home of the database. You must log in as `SYSDBA` to run the script.

- Oracle Database Vault must be disabled before starting the migration process, because DMU has not been certified to work with it enabled.
- The database must be opened in read/write mode.

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

- 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. In other words, the data dictionary of the database cannot contain non-ASCII characters except in a few selected tables.
- 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.

Overview of Java Runtime Requirements

The Java JDK 1.6 or later must be available on the host running the DMU. The DMU will ask for the location of the Java Runtime executable file (`java.exe` or `java`, depending on the platform) the first time it is started.

From the Java SE download page on `http://www.oracle.com`, you can download the newest Java 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/US or IBM AIX, visit the vendor's website.

Overview of DMU Security Considerations

This release of the Database Migration Assistant for Unicode requires that the database users connecting to a database have the `SYSDBA` privilege.

As such, a user should not grant any privileges for the migration repository database objects to any other database users. This applies to any object with a name starting with `DUM$` or `DBMS_DUMA`.

If the DMU is installed on a shared workstation or on a server, you ensure that only the owning operating system user can modify the installed files. This security measure prevents malicious users from modifying the executable files to take advantage of the fact that the DMU connects to a database with the `SYSDBA` privilege. The modified utility could compromise the database security by running unexpected SQL statements with administrative privileges.

Review Your Preparations for Migration

Before you start the migration process, collect the following information about your database and its use by applications to help you set up the DMU properly:

- 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" on page 2-2. 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" on page 1-10) 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 might 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" on page 6-25 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). 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 SID or a service name of the database. For performance reasons, Oracle recommends that you connect to a service that is configured for dedicated connections, that is, not using the Shared Server feature.

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

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

Performing First Tasks With the DMU

The basic introductory steps are:

- Installing the DMU
- Creating a Database Connection
- Installing the Migration Repository
- Following the Status of the Migration

Installing the DMU

Before installing the DMU, always download the current release notes available for the DMU version you are using. Oracle recommends that the DMU be run on the database server host or on a workstation connected to the database server host with a fast and reliable local area network.

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.

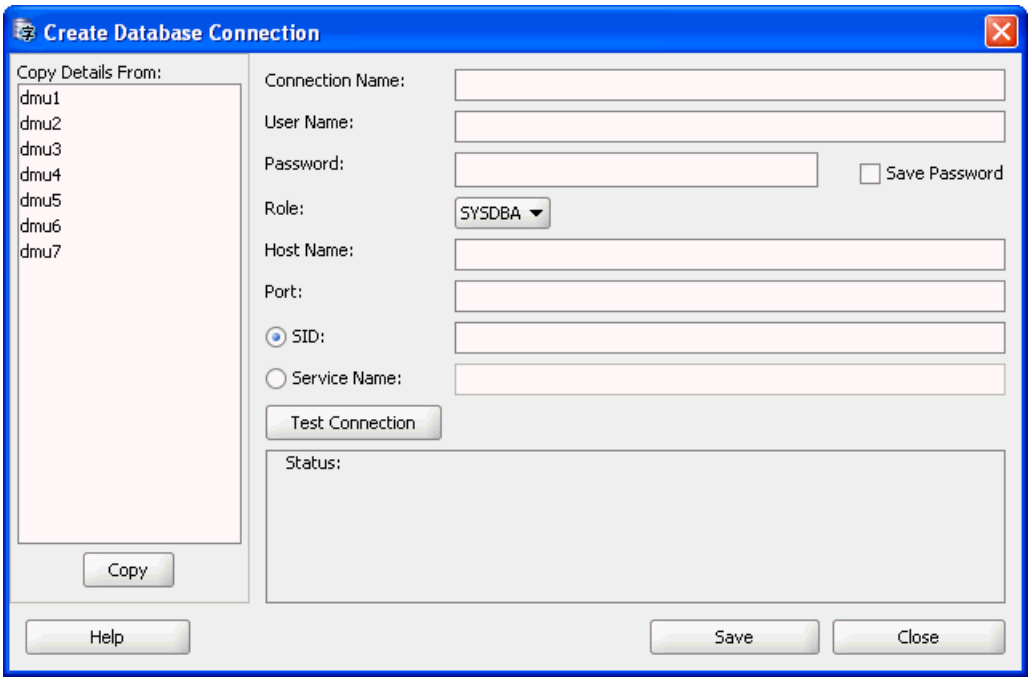
Creating a Database Connection

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

To create a database connection:

1. Before you can start migrating a database with the DMU, you must provide database connection details to the tool. To do this, right-click the **Databases** node in the Navigator pane to open the context menu, or open the **File** menu from the menu bar, and select **New Database Connection**. The Create Database Connection dialog box, shown in Figure 2–1 is displayed. In this dialog, enter connection details, such as login ID, password, host name, port number, and the database SID.

Figure 2–1 Creating a Database Connection



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.

   - Role

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

   - Host Name

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

   - Port

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

   - SID or Service Name

     Specify the System Identifier (SID) or the service name of the database instance to which you want to connect.

   If you select the **Save Password** check box, the password you specify 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 Unix platforms, the files are in the directory `$HOME/.dmu/`. On Microsoft Windows, the files are in the directory `%USERPROFILE%\Application Data\DMU\`. 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.

You can test the connection by clicking **Test Connection**. 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.

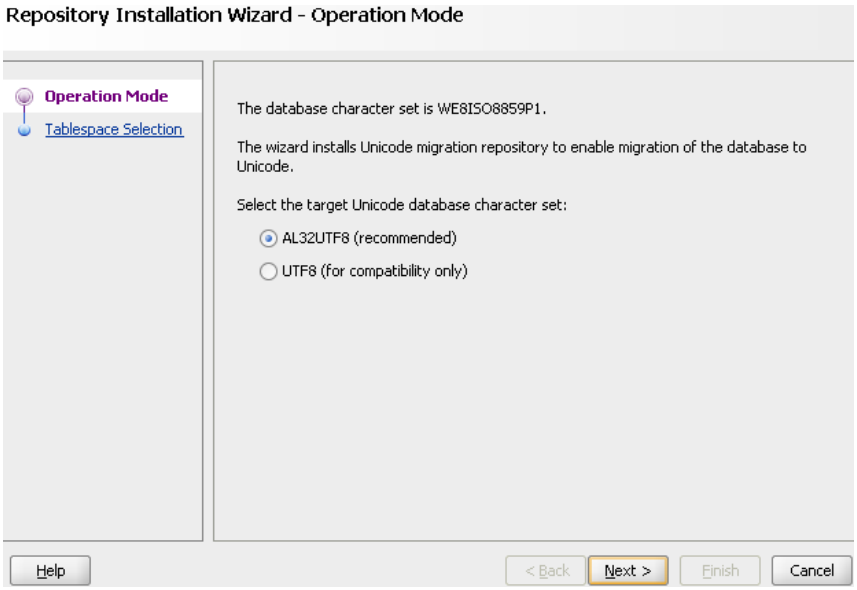
Installing the Migration Repository

The DMU uses a repository to manage the information necessary for each step of the migration. 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 migration 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 Migration 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 **Install Migration Repository**. You can also select **Install Migration Repository** from the Migration menu. In all of these cases, the Repository Installation Wizard appears. Figure 2–2 shows how it will appear.

Figure 2–2 Repository Installation Wizard - Operation Mode

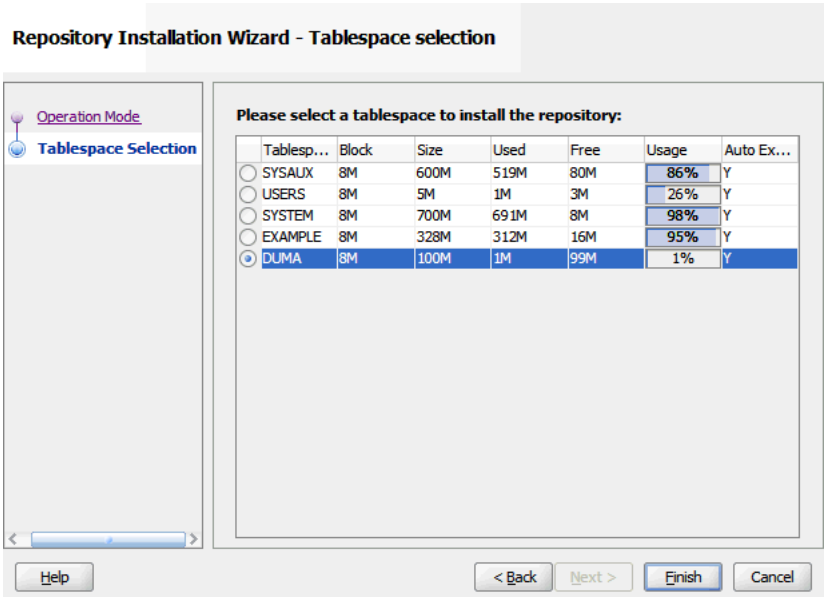


2. On the first page of the wizard, you select the target character set for the migration. You can choose between AL32UTF8, the normally recommended character set, and UTF8, which might be required for compatibility with certain applications, such as Oracle Applications 11*i*. After selecting the character set, click **Next**.

   If the character set of the database to be migrated is UTF8, the first page of the wizard will give you a choice between migration of the database to AL32UTF8 and installation of the migration repository in the Unicode validation mode. See "Validating Data as Unicode" on page 4-29. If the character set of the database is already AL32UTF8, the migration repository will be automatically installed in the Unicode validation mode.

   After you click **Next**, the second page of the Repository Installation Wizard is shown. Figure 2–3, "Repository Installation Wizard - Tablespace Selection" is an example of this page.

Figure 2–3 Repository Installation Wizard - Tablespace Selection



3. On the second page, you can select the tablespace in which you want to install the repository. The default tablespace is `SYSAUX`. Oracle recommends that a separate, empty tablespace be used for the migration repository. This helps to avoid fragmentation of the production tablespaces. Ensure that the amount of free space available in the chosen tablespace is sufficient to hold the repository contents. Because the amount of space required by the repository heavily depends on the rowid collection level you choose for scanning and the amount of character data in the database that requires conversion or has convertibility issues, the space required by the repository is difficult to estimate. Therefore, Oracle recommends that you enable the autoextend option on the data files of the repository tablespace to allow the tablespace to grow, if necessary. Define the maximum size constraints for the data files to avoid exhausting the entire disk space.

   Click **Finish** to install the repository.

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

Following the Status of the Migration

After a database connection has been established, the Migration Status tab is displayed in the Client pane of the DMU main window. You can follow the progress of the migration process by looking at the information on this tab. This tab shows the important milestones in the DMU process: installing the repository, scanning the database for problems, resolving any problems encountered, and then 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 2–4. The Migration Status tab describes each of the four main steps of the migration: installation of the migration repository, scanning of the database, resolution of migration issues, and conversion of the database. The tab shows the status of each of the steps and suggests

the next action to take. You can click the **More** links to open help pages that expand the presented information.

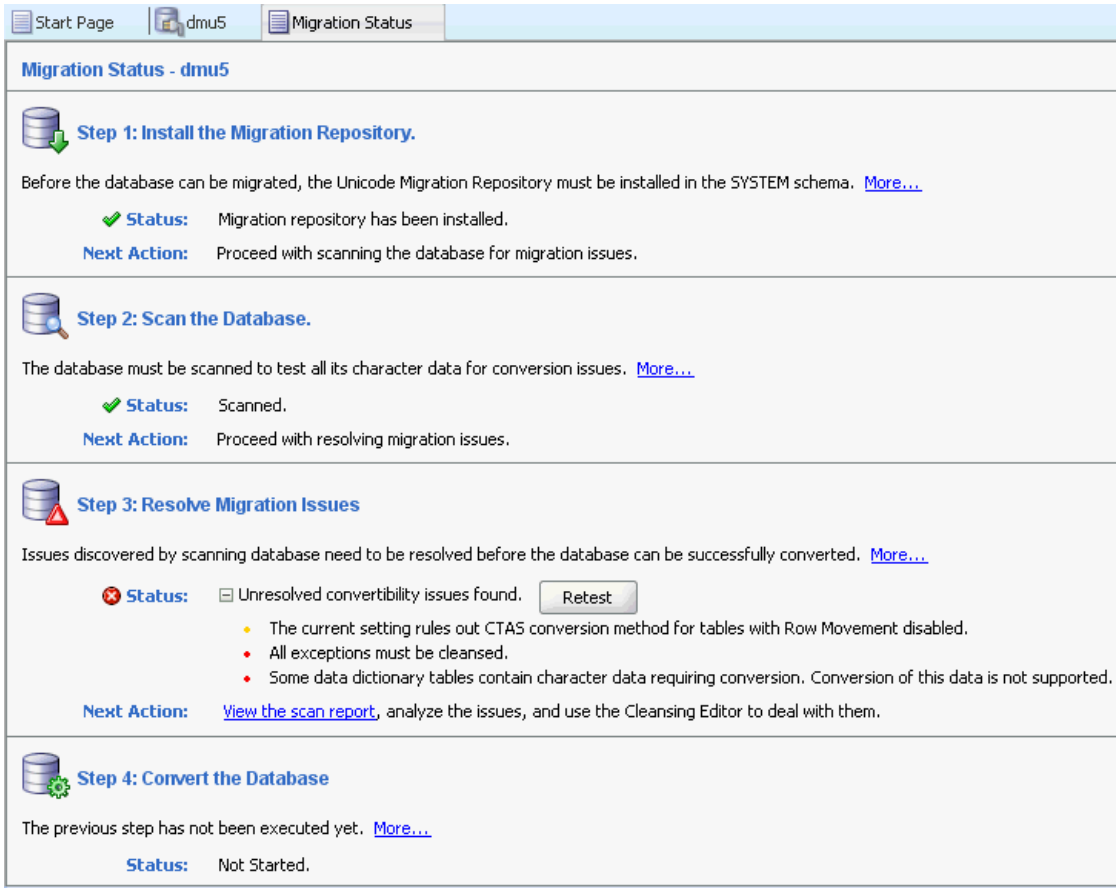
The Status area of the section "Step 1: Install Migration Repository" informs you if the connected database contains a migration repository. It can also report that the installed repository is in an incompatible version. Until the repository is installed in the correct version, you cannot proceed to 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 Migration Repository" on page 2-8.

The Status area of the section "Step 2: Scan the Database" reports 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 migration 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" on page 4-4 and scan all those tables that have missing or invalid scan results.

Figure 2–4 Migration Status Tab



The information presented in the Status area of the "Step 3: Resolve Migration Issues" section might contain a 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 for the "Step 3: Resolve Migration Issues" section shows "No unresolved convertibility issue found" and proceeding to the conversion is considered safe.

The Status area of the section "Step 4: Convert Database" informs you if the database conversion has ready been started and, later, if it has already finished successfully.

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.

Introduction to the DMU User Interface

The default layout of the main window of the DMU user interface is shown in Figure 2–5. The window 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 *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" on page 2-6.

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 Data Dictionary 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.

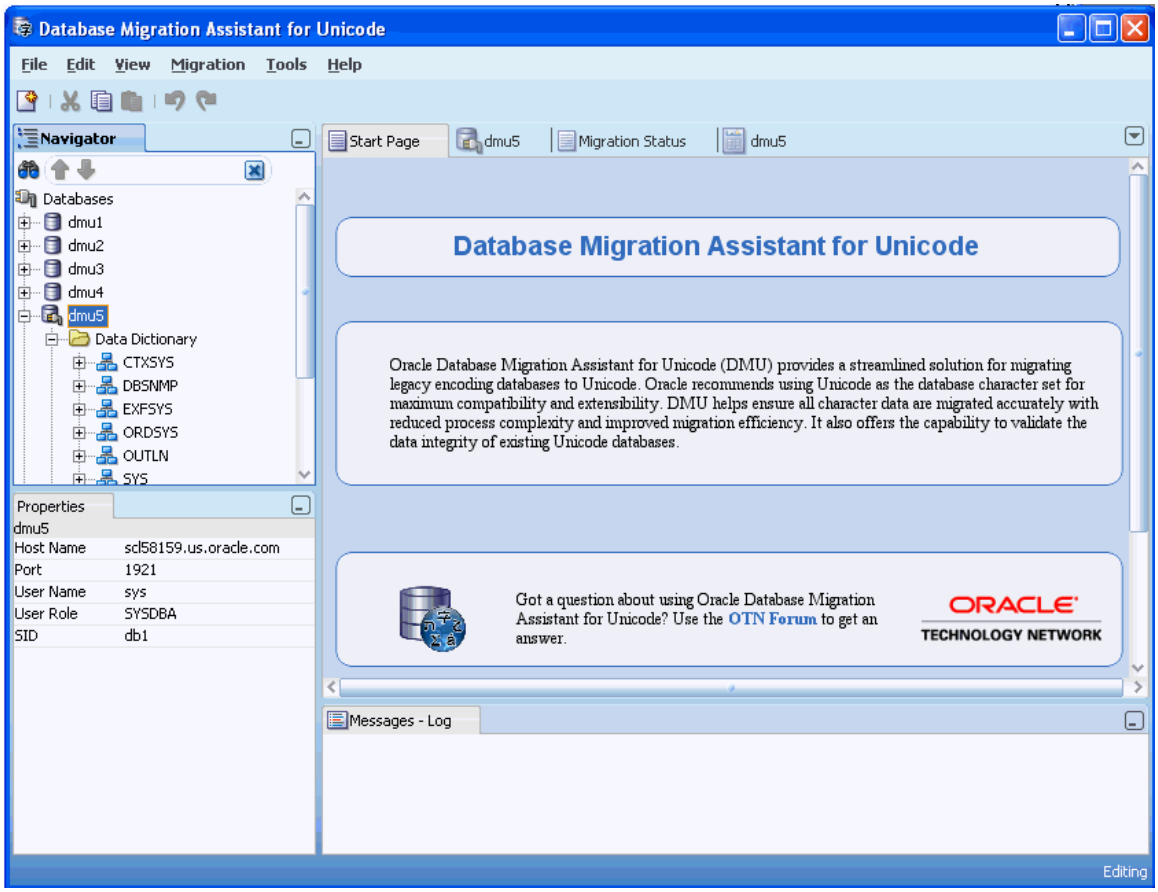
A status icon might be displayed to the left of a node name in the Navigator tree. Table 2–1 shows the possible icons and their meanings for particular node types.

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" on page 4-4). 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 course of the migration process.

Figure 2–5 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.

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

Icon	Column Node	Table, Schema, Database, and Grouping Nodes
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.
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.

Overview of Data Preparation

Data preparation ensures 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.

Data Preparation: Scanning

In this step, tables are scanned for problems of various kinds. This scanning assesses the feasibility of migrating the data to Unicode. The most common types of problems are with your data, including issues such as 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:

- Need no conversion

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

- Need 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.

- Exceed column limit

  The data will not fit into a column after migration.

- Exceed data type limit

  The data will exceed a data type limit after migration.

Each column value is assigned to only one of the above 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 (see "Ignoring Convertibility Issues" on page 6-15), 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. See "Column Properties: Scanning" on page 3-13 for more information about the property.

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.

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

See Also: "Scanning the Database" on page 4-4 for details

It is important to remember that, as mentioned in "Overview of Character Set Migration Considerations" on page 1-9, there are certain data issues that 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, the test conversion during a scan might not show invalid binary representation problems, if the incorrectly stored data happens not to contain any of the undefined byte values. 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.

Data Preparation: Cleansing

In this step, the data in tables is cleaned based on identified issues. You can define cleansing actions for immediate or for delayed execution (immediate versus scheduled cleansing mode). Certain types of cleansing can be done in a production environment with no side effects to applications, but there are other cleansing operations that involve metadata changes and require applications to be adapted 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:

- Chapter 6, "Using the DMU to Cleanse Data" for details

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. The conversion consists of the following steps:

- Preparing the Conversion
- Converting Data

Preparing the Conversion

When you tell the DMU to start the conversion phase by selecting **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" on page 2-2 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**.

Converting Data

The next step is to actually convert 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. Convert the data in user tables and in selected data dictionary tables to Unicode.
6. Convert `CLOB` columns in the data dictionary.
7. Issue the `ALTER DATABASE CHARACTER SET` statement.

8. Enable triggers and constraints; and re-create indexes and constraints.
9. Restore the database instance parameters.

The conversion of a table is performed either by updating its columns with an `UPDATE` statement or by converting the columns while re-creating the table using the `CREATE TABLE AS SELECT` statement. The re-creation of a table is faster that an update if most of the table rows must be converted.

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

> **See Also:** "Converting the Database" on page 4-21 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.

This chapter contains the following sections:

Viewing and Setting Database Properties

The Database Properties tab has four subtabs: General, Scanning, Readiness, and Converting, as shown in Figure 3–1.

Database Properties: General

On the General subtab of the Database Properties tab, you can see the following properties:

- Connection details, such as user name and host name

The connection details are described in "Creating a Database Connection" on page 2-6. The connection details can be changed only as described in that section.

- Information about character sets of the database

  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

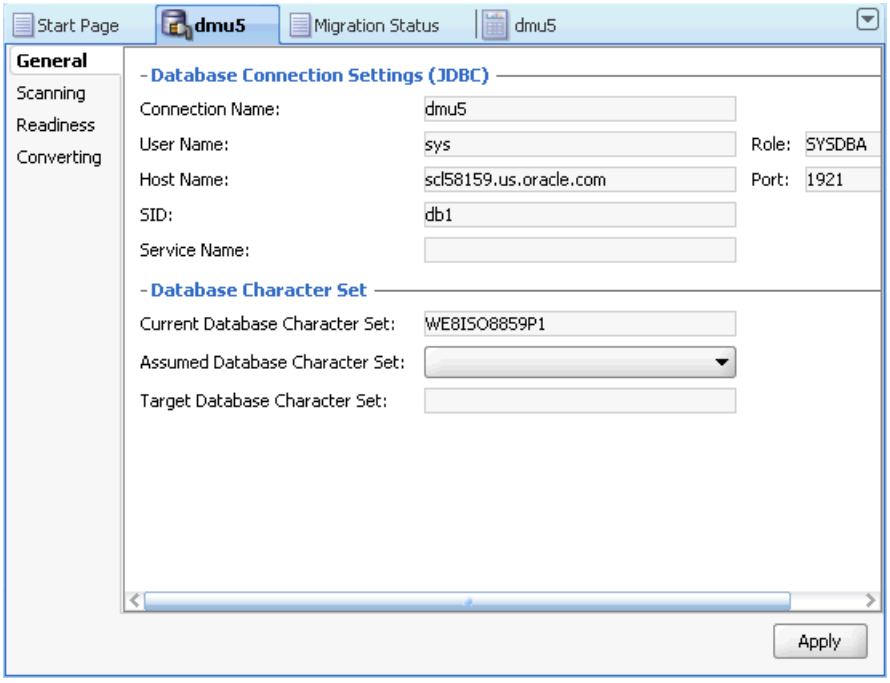
    The DMU uses the assumed database character set as the default source character set when scanning and converting character data during the migration. If the database has been used in a correct character set configuration, then the assumed character set should be the same as the database character set. However, in a pass-through configuration, as described in "Invalid Binary Storage Representation of Data" on page 1-10, 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 character set of the migration, either UTF8 or AL32UTF8. You can change this property only by uninstalling the migration repository and installing it again.

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

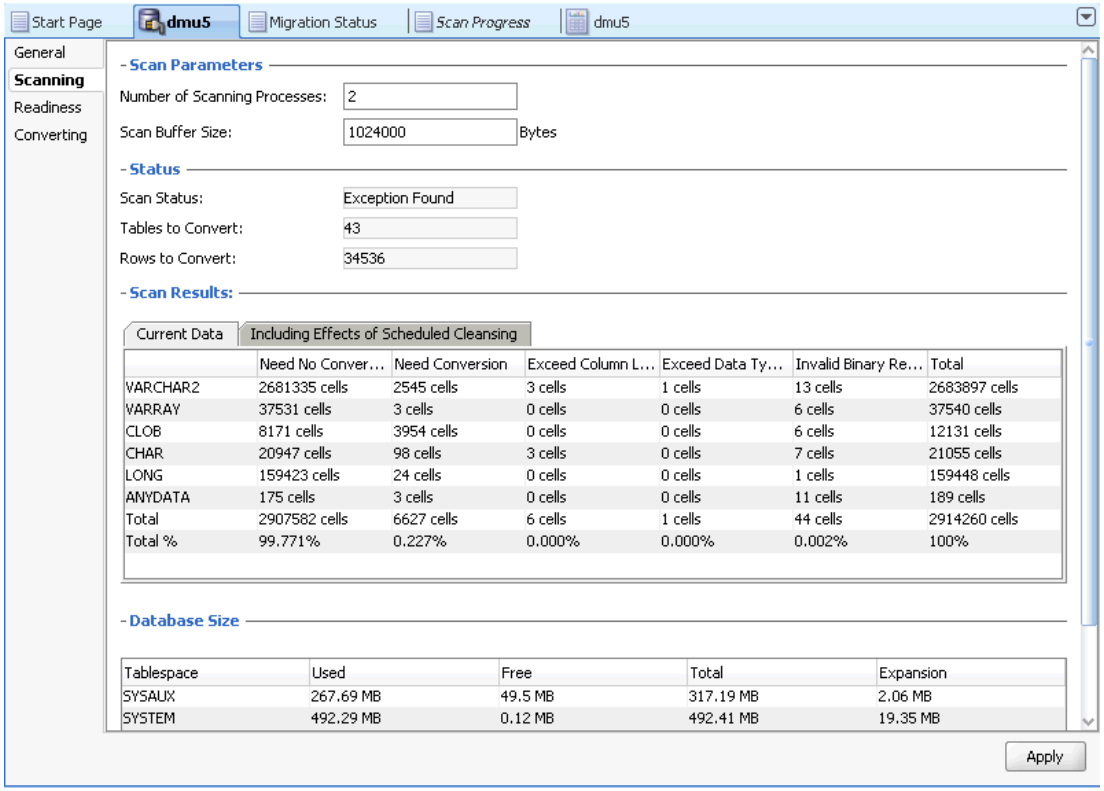
Figure 3–1 Database Properties: General



Database Properties: Scanning

On the Scanning subtab of the Database Properties tab, you can set general parameters controlling the process of scanning in this database, and 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" on page 4-15 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" on page 4-6 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 migration 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 needs to 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.

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

Database Properties: Readiness

This subtab shows the readiness of data for conversion. See Figure 3–3 for an illustration. 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" on page 2-18.

- Ready for conversion

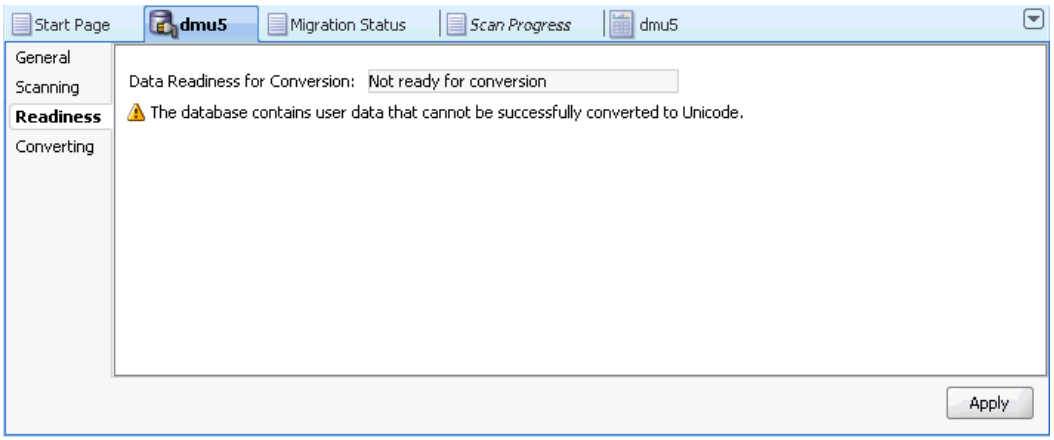
  All data in the database is classified as either needing no conversion, or needing conversion, and the database passed all conversion feasibility tests.

- 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.

Figure 3–3 Database Properties: Readiness



Database Properties: Converting

Properties on this subtab allow you to control certain aspects of the conversion processing. See Figure 3–4 for an illustration. The following properties can be set:

- Degree of CTAS 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.

- Parallel Execution Threshold

  If a table is smaller than this threshold, it will be converted serially, without using the parallel execution feature of the database.

- 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.

- 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.

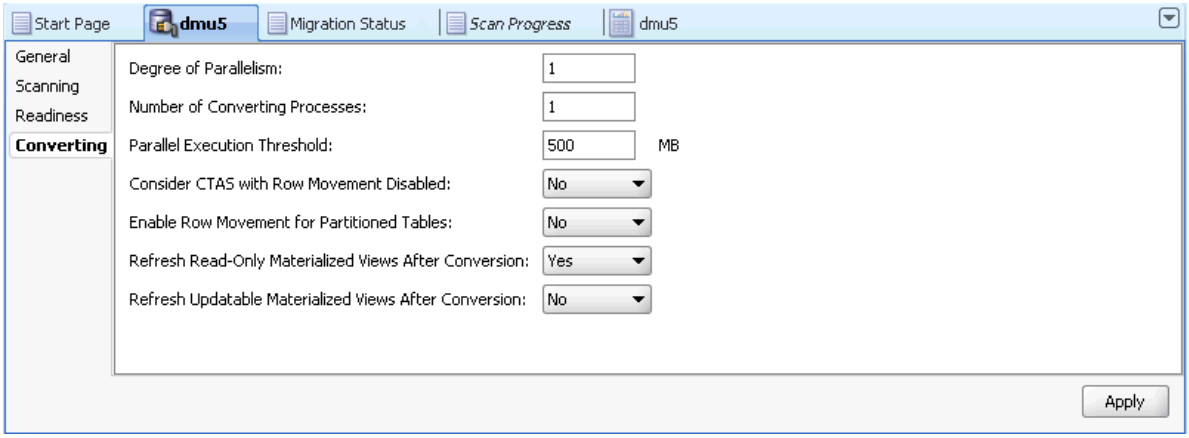
- Refresh Read-Only Materialized Views after Conversion

  Set this property to Yes to let the DMU automatically refresh read-only materialized views after their master tables have been converted.

- Refresh Updatable Materialized Views after Conversion

  Set this property to Yes to let the DMU automatically refresh updatable materialized views after their master tables have been converted.

Figure 3–4 Database Properties: Converting



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

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.

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. These properties are read-only.

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:

- 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 migration 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" on page 4-4.

All the preceding properties are read-only.

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.

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.

Table Properties: General

The General subtab of the Table Properties tab 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.

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:

- 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 Chapter 6, "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" on page 4-4.

- 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 migration 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" on page 4-4.

  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.

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.

Table Properties: Converting

Properties on this subtab allow you to control certain aspects of the conversion of the table. The following properties can be set:

- 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.

  - 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 rows changing in conversion

    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.

  The DMU automatically assigns one of the preceding conversion methods to each scanned table. 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.

- 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.

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.

Column Properties: General

The General subtab of the Column Properties tab shows the following properties of a table:

- 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.

These properties are read-only.

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 Chapter 6, "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 migration 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" on page 3-1.

The properties on this subtab are read-only.

Column Properties: Readiness

The Readiness subtab of the Column Properties tab contains the following properties:

- Scheduled Cleansing Action

  This property shows the scheduled cleansing action defined for the column in the Cleansing Editor. See Chapter 6, "Using the DMU to Cleanse Data".

- Data Readiness for Conversion

  This property tells if 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," additional information is displayed to explain the problem.

- Allow Conversion of Exceptional Data

  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.

Column Properties: Converting

Properties on this subtab allow 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 true, the DMU will not update the data in this column.

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 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.

This chapter contains the following sections:

Initializing the Database

The initialization of a database for use with the DMU includes the following tasks:

Installing Required Patches

To install required patches:

1. In the downloaded, newest DMU release notes, check if any database patches are required for your database release to work with the DMU. Then use the `opatch` utility to check if the required patches are already installed. Finally, 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 issue the following statement:

   ```
   opatch lsinventory -patch
   ```

If the `opatch` utility identifies the lack of a valid `oraInst.loc` file, 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_22-50-28PM.log

Lsinventory Output file location : C:\oracle\product\10.2.0\db_
1\cfgtoollogs\opatch\lsinv\lsinventory2010-05-17_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.
```

2. Check if the required database patches are listed among interim patches. Download any missing patches from the My Oracle Support site and install them. Then review the documentation that is part of the patches for installation instructions. These patches can be found at:

   http://support.oracle.com

Installing Supporting Packages

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

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.
```

In addition, you should perform the preceding steps if installation documentation from any installed database patches requires them.

Creating a Tablespace for the Migration Repository

For ease of maintenance and to avoid fragmentation of other production tablespaces, Oracle recommends that a separate tablespace be used for database objects (tables and indexes) forming the migration repository of the DMU. In order to follow this recommendation, create a suitable tablespace now, so that you can select it for the migration repository in the following steps. See *Oracle Database 2 Day DBA* or *Oracle Database Administrator's Guide* for instructions about creating a tablespace.

To create a tablespace for the Migration Repository:

Oracle recommends that the tablespace be locally managed with the default extent allocation policy and with the autoextension feature switched on.

1. Use the result of the following SQL query (run as SYSDBA) as the initial size of the tablespace:

```
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 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 autoextend feature be switched on to allow the tablespace to grow as necessary.

Creating a Database Connection

See "Creating a Database Connection" on page 2-6.

Refreshing the Migration Repository

After a repository has been installed, it is automatically refreshed at predefined points in the DMU workflow to account for any objects containing character data that may have been added to, altered in, or removed from the database.

To refresh the Migration Repository:

1. You can explicitly refresh the repository to immediately include any recent changes to the database objects by selecting **Refresh Migration Repository** from the Migration menu or from the context menu of the database node in the Navigator pane. After the refresh has finished, a confirmation message appears announcing that the repository has been successfully refreshed.

Uninstalling the Migration Repository

To uninstall the Migration Repository:

1. To remove a repository, from the Migration menu or the context menu of the database node in the Navigator pane, select **Uninstall Migration Repository**. A warning dialog box will appear that asks you to confirm that you really want to uninstall the repository. When the repository has been uninstalled, a confirmation message appears to confirm the success of the operation.

Scanning the Database

The next step is to scan your current data. In general, what you want to achieve in this step is to analyze what is causing problems and to choose a cleansing strategy to resolve the data issues.

Scanning is the process of reading character values from the database, converting them to the target character set, and counting 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. The DMU stores the calculated counts and statistics in the migration 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 character data in the `VARCHAR2`, `CHAR`, `LONG`, and `CLOB` table columns must be analyzed 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:

- 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

This section discusses:

- Setting Database Properties
- Scanning the Database with the Scan Wizard
- Viewing the Database Scan Report

Setting Database Properties

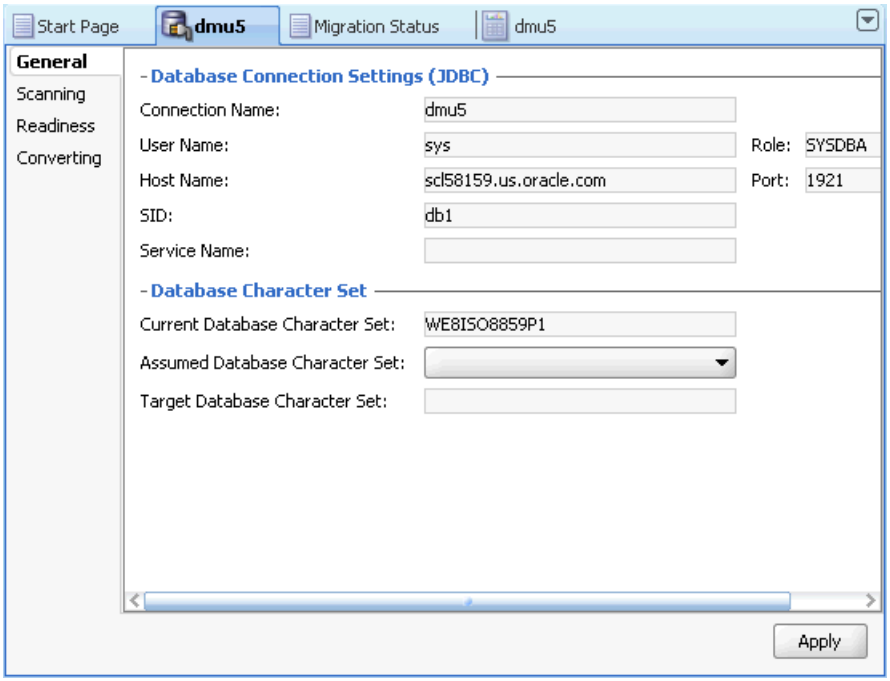
Before you start scanning the database, you should provide the DMU with the language and character set information that you collected while preparing the migration process, as described in "Review Your Preparations for Migration" on page 2-4.

To set database properties:

1. Open the Database Properties tab, which is illustrated in Figure 4–1. This tab is automatically opened in the client pane of the DMU window when you connect to a database. The title of the tab is the name of the connection.

   You can also view this tab by right-clicking the database node in the Navigator pane and selecting **Properties** from the context menu. From the left sidebar of the tab, choose the General subtab. 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.

Figure 4–1 Database Properties Tab - General



See Also: Chapter 3, "Viewing and Setting Object Properties in the DMU" for more details about database and other properties

Scanning the Database with the Scan Wizard

You are now ready to run the first scan of the database contents. This process is started through the Scan Wizard. The scan results inform you if any 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.

Three elements of the DMU user interface let you control the scanning process and view the scan results. You use the Scan Wizard to start the scanning process, the Scan Progress tab to monitor this process (See "Monitoring the Progress of a Scan" on page 4-12), and the Database Scan Report tab to view the scan results (See "Viewing the Database Scan Report" on page 4-14).

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.

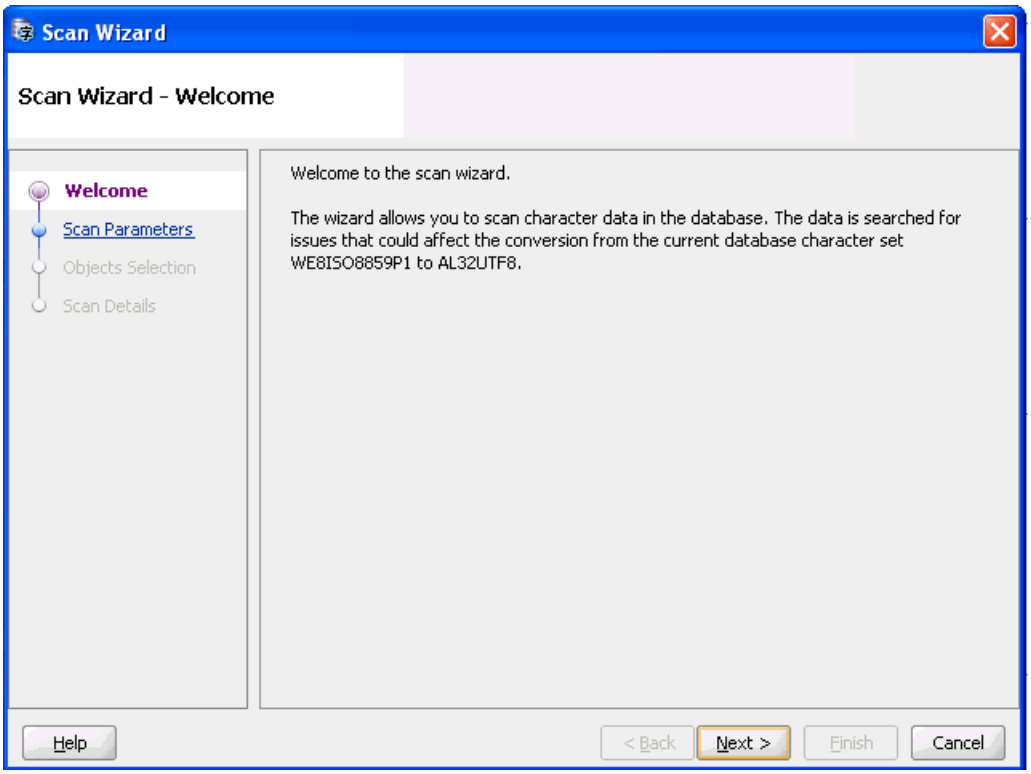
`CLOB` columns are not scanned or even listed in the Navigator pane in a multibyte database. Any multibyte database, including databases in AL32UTF8 and UTF8 character sets, store `CLOB` values in the same internal storage character set (Unicode UTF-16), and so `CLOB` values need no conversion when the 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" on page 4-21.

To start the Scan Wizard:

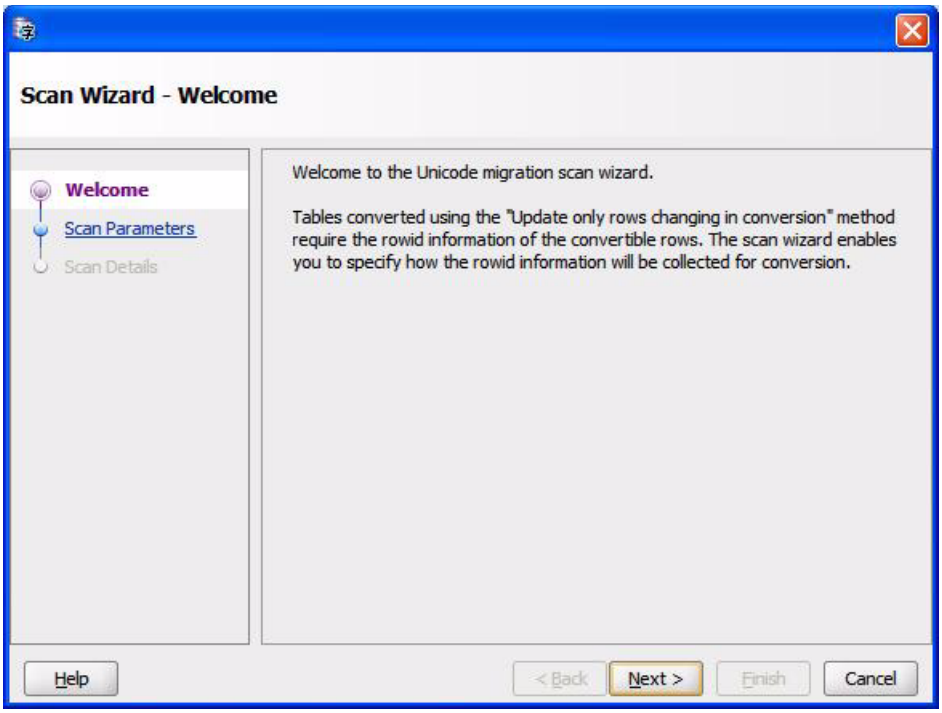
1. To open the Scan Wizard, select **Scan Database** from the Migration menu or **Scan Database**, **Scan Schema**, **Scan Table**, or **Scan Column** from the context menu of the corresponding object node in the Navigator pane. The Scan Wizard is displayed, as in Figure 4–2, "Scan Wizard: Welcome".

Figure 4–2 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 4–3, "Scan Wizard - Welcome: Rowid Collection". It also lacks the object selection page. See "Converting the Database" on page 4-21 for more information about the conversion phase.

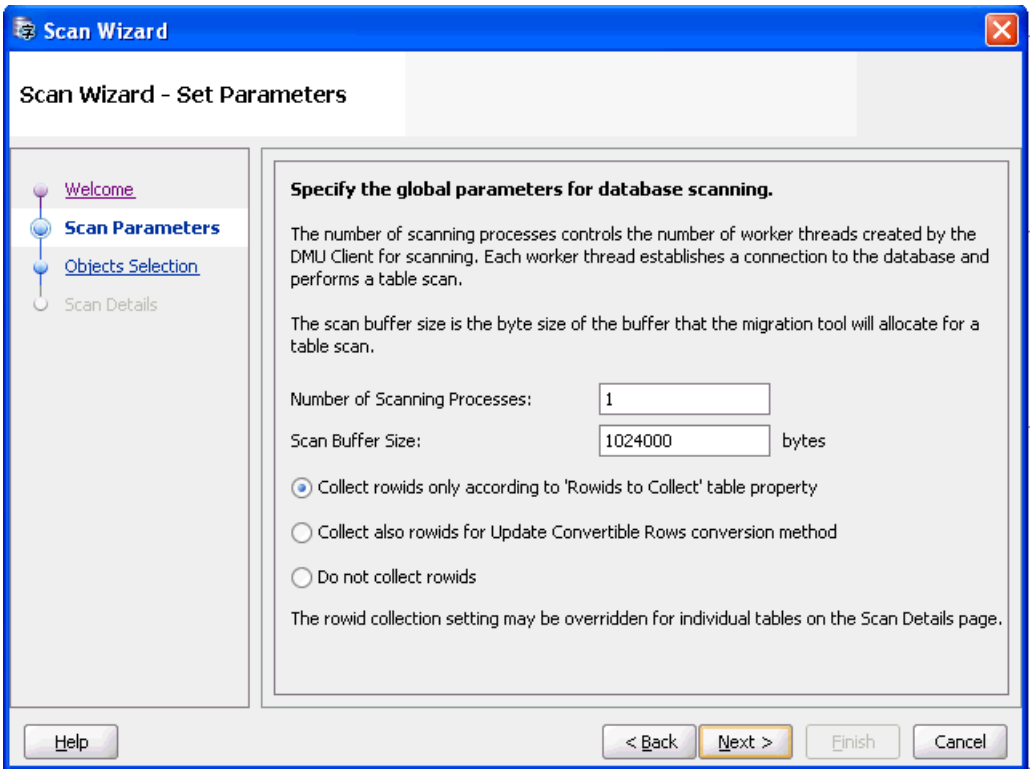
Figure 4–3 Scan Wizard - Welcome: Rowid Collection



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 4–4, "Scan Wizard: Set Parameters".

   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 4–4 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" on page 3-9 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.

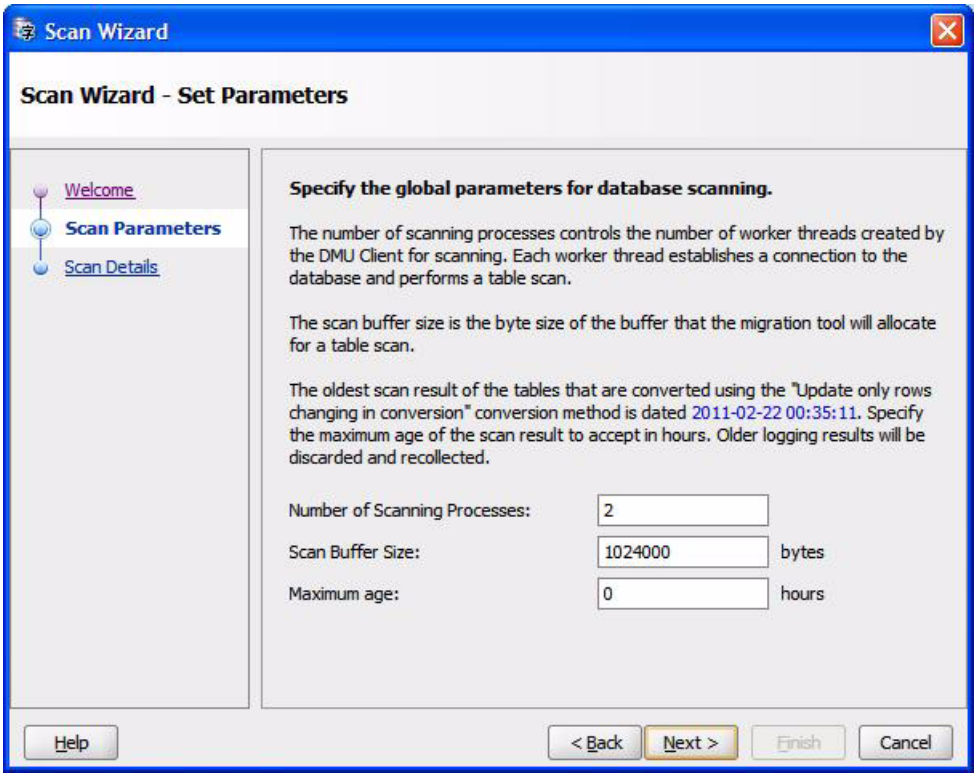
Pre-collection of rowids removes the need for this additional scan, shortening the required downtime window and enabling the database to be closed for business for a shorter time. However, if any additional convertible rows are added to a table after the last scan that pre-collected rowids for the table 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".

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 4–5, "Scan Wizard - Set Parameters: Rowid Collection". 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 4–5 Scan Wizard - Set Parameters: Rowid Collection



See "Converting the Database" on page 4-21 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

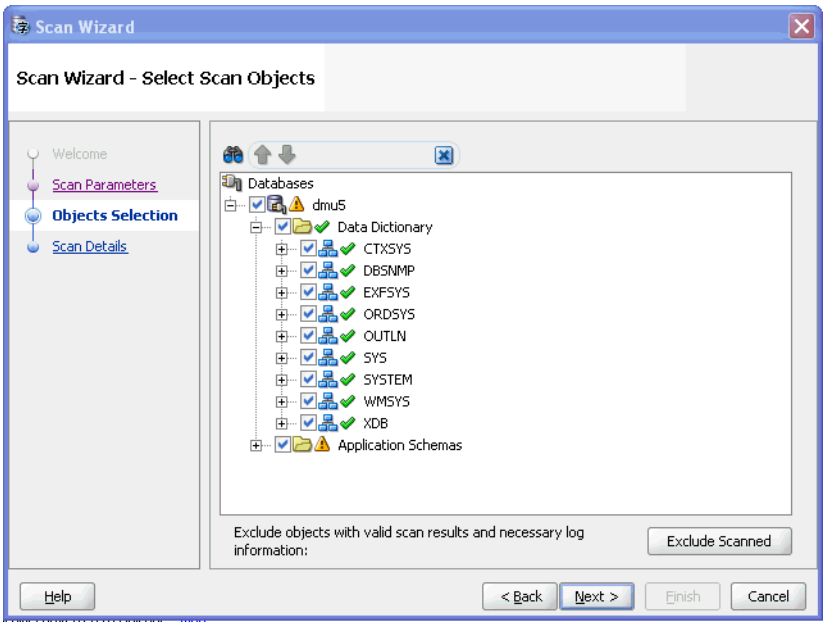
recent DML activity on the objects to be analyzed in the context of possibly introduced conversion issues.

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 4–6 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.

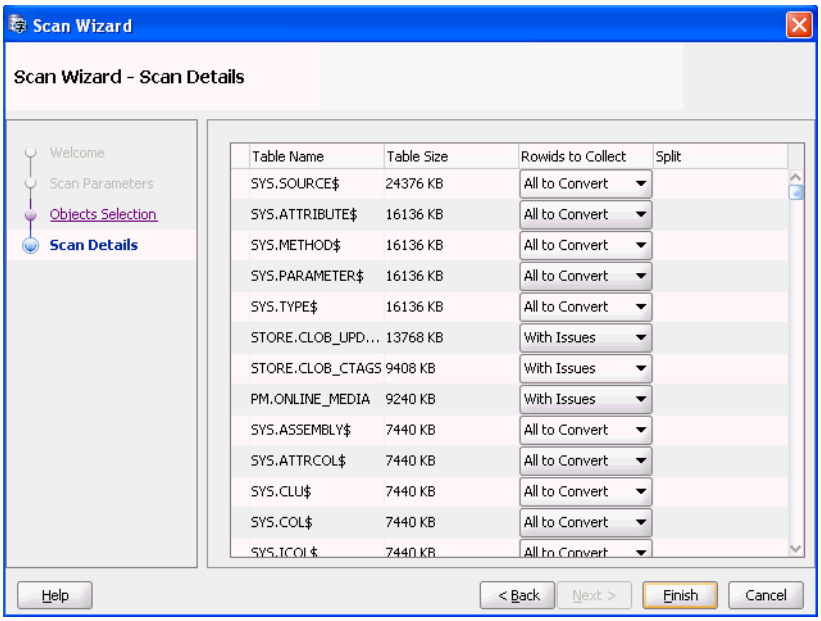
5. On the Scan Details page, shown in Figure 4–7, "Scan Wizard: Scan Details", you can verify that all objects you wanted to be scanned are included. You can also set the rowid collection level in the drop-down lists that can be opened by selecting fields in the Rowids to Collect column. The choices 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 Chapter 6, "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.

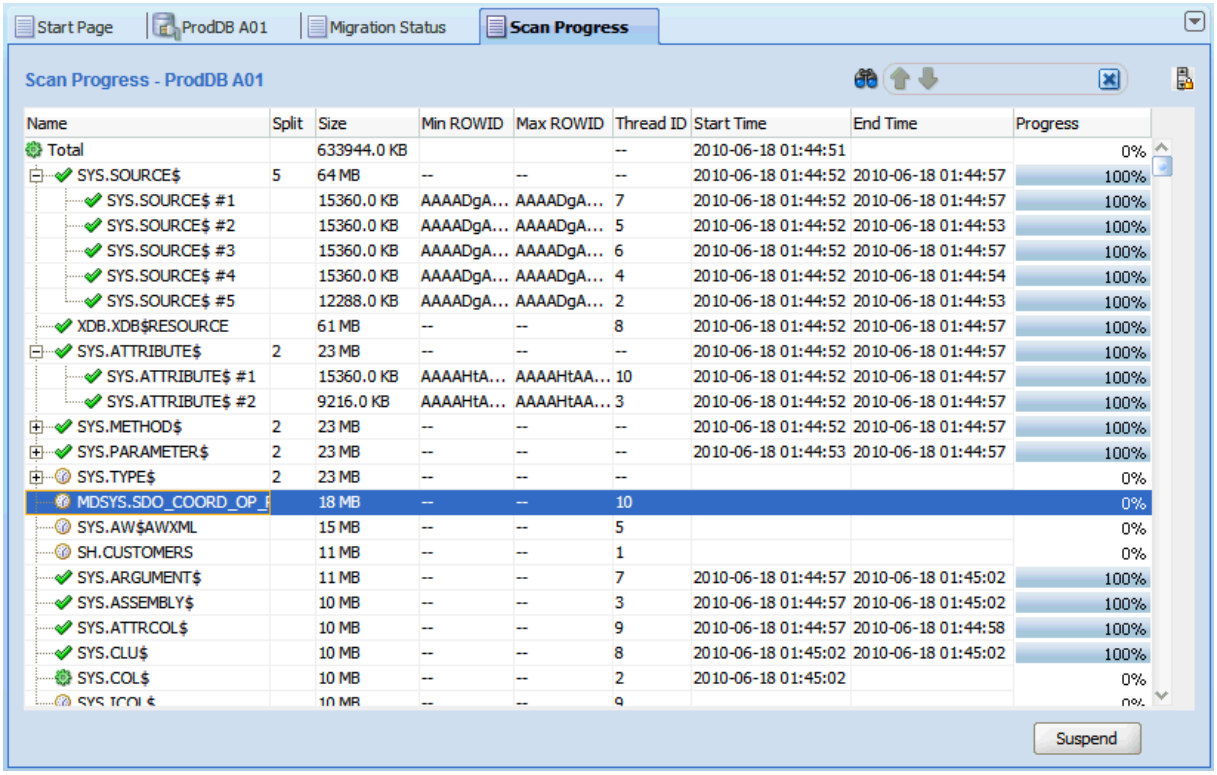
Figure 4–7 Scan Wizard: Scan Details



Monitoring the Progress of a Scan

When the scanning starts, the DMU automatically opens the Scan Progress tab, shown in Figure 4–8. You can use this tab to monitor the progress of the scanning task.

Figure 4–8 Scan Progress Tab



Name	Split	Size	Min ROWID	Max ROWID	Thread ID	Start Time	End Time	Progress
Total		633944.0 KB			--	2010-06-18 01:44:51		0%
SYS.SOURCE$	5	64 MB	--	--	--	2010-06-18 01:44:52	2010-06-18 01:44:57	100%
SYS.SOURCE$ #1		15360.0 KB	AAAADgA...	AAAADgA...	7	2010-06-18 01:44:52	2010-06-18 01:44:57	100%
SYS.SOURCE$ #2		15360.0 KB	AAAADgA...	AAAADgA...	5	2010-06-18 01:44:52	2010-06-18 01:44:53	100%
SYS.SOURCE$ #3		15360.0 KB	AAAADgA...	AAAADgA...	6	2010-06-18 01:44:52	2010-06-18 01:44:57	100%
SYS.SOURCE$ #4		15360.0 KB	AAAADgA...	AAAADgA...	4	2010-06-18 01:44:52	2010-06-18 01:44:54	100%
SYS.SOURCE$ #5		12288.0 KB	AAAADgA...	AAAADgA...	2	2010-06-18 01:44:52	2010-06-18 01:44:53	100%
XDB.XDB$RESOURCE		61 MB	--	--	8	2010-06-18 01:44:52	2010-06-18 01:44:57	100%
SYS.ATTRIBUTE$	2	23 MB	--	--	--	2010-06-18 01:44:52	2010-06-18 01:44:57	100%
SYS.ATTRIBUTE$ #1		15360.0 KB	AAAAHtA...	AAAAHtAA...	10	2010-06-18 01:44:52	2010-06-18 01:44:57	100%
SYS.ATTRIBUTE$ #2		9216.0 KB	AAAAHtA...	AAAAHtAA...	3	2010-06-18 01:44:52	2010-06-18 01:44:57	100%
SYS.METHOD$	2	23 MB	--	--	--	2010-06-18 01:44:52	2010-06-18 01:44:57	100%
SYS.PARAMETER$	2	23 MB	--	--	--	2010-06-18 01:44:53	2010-06-18 01:44:57	100%
SYS.TYPE$	2	23 MB	--	--	--			0%
MDSYS.SDO_COORD_OP_I		18 MB	--	--	10			0%
SYS.AW$AWXML		15 MB	--	--	5			0%
SH.CUSTOMERS		11 MB	--	--	1			0%
SYS.ARGUMENT$		11 MB	--	--	7	2010-06-18 01:44:57	2010-06-18 01:45:02	100%
SYS.ASSEMBLY$		10 MB	--	--	3	2010-06-18 01:44:57	2010-06-18 01:45:02	100%
SYS.ATTRCOL$		10 MB	--	--	9	2010-06-18 01:44:57	2010-06-18 01:44:58	100%
SYS.CLU$		10 MB	--	--	8	2010-06-18 01:45:02	2010-06-18 01:45:02	100%
SYS.COL$		10 MB	--	--	2	2010-06-18 01:45:02		0%
SYS.ICOL$		10 MB	--	--	9			0%

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.

- 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.

You can suspend the scanning process by clicking **Suspend** on the bottom of the Scan Progress tab. The button will change to Resume. Click **Resume** to resume scanning.

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

Viewing the Database Scan Report

After the scan is completed, you can look at the Database Scan Report to view the scan results.

To view the Database Scan Report:

1. Open the Database Scan Report for the whole database by selecting **Database Scan Report** from the Migration menu or from the context menu of the database node in the Navigator pane. The Database Scan Report tab is opened in the client pane of the DMU window. See Figure 4–9, "Database Scan Report".

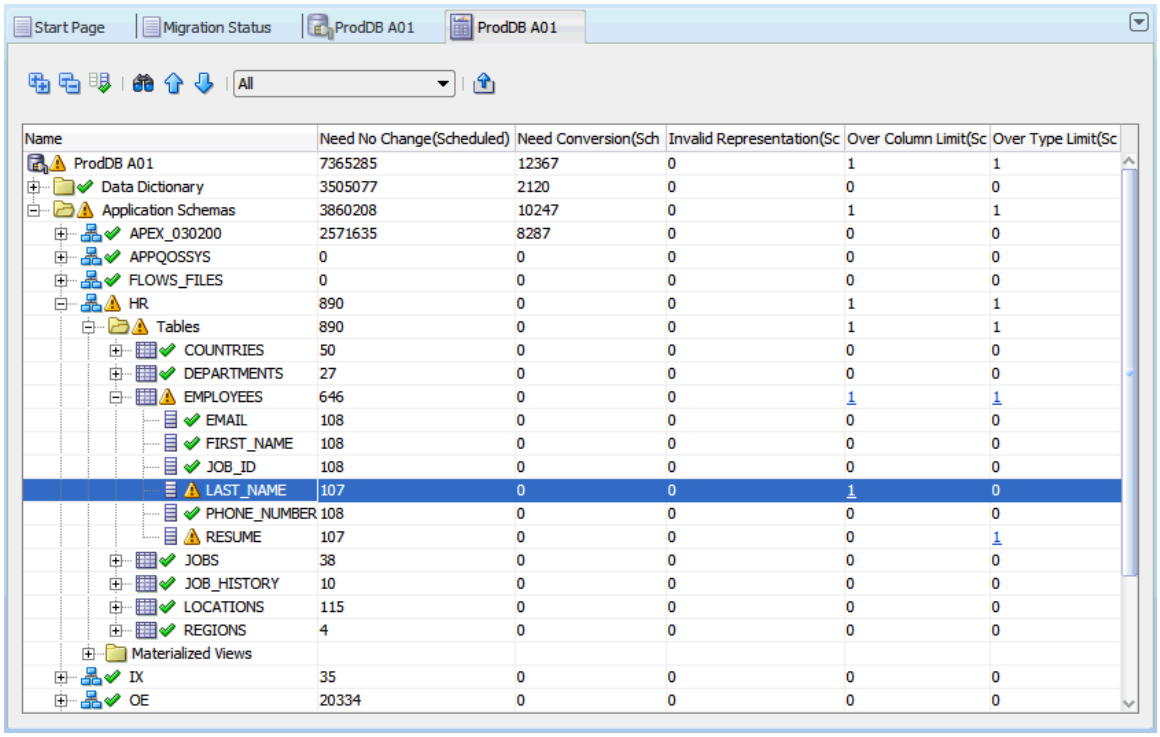
   Depending on what the scan results are, you should follow one the migration scenarios described in "Cleansing Scenario 1: A Database with No Issues" on page 6-15, "Cleansing Scenario 2: Cleansing Expansion Issues" on page 6-16, or "Cleansing Scenario 3: Cleansing Invalid Representation Issues" on page 6-25.

Overview of the Database Scan Report

You can view the most current scan results either on the Scanning subtabs of various Properties tabs, described in Chapter 3, "Viewing and Setting Object Properties in the DMU", or, in a much more convenient way, on the Database Scan Report tab.

You can open a Database Scan Report tab with scan results for the whole database by selecting **Database Scan Report** under the Migration menu. 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. A Database Scan Report tab for the whole database is shown in Figure 4–9.

Figure 4–9 Database Scan Report



Name	Need No Change(Scheduled)	Need Conversion(Sch	Invalid Representation(Sc	Over Column Limit(Sc	Over Type Limit(Sc
ProdDB A01	7365285	12367	0	1	1
Data Dictionary	3505077	2120	0	0	0
Application Schemas	3860208	10247	0	1	1
APEX_030200	2571635	8287	0	0	0
APPQOSSYS	0	0	0	0	0
FLOWS_FILES	0	0	0	0	0
HR	890	0	0	1	1
Tables	890	0	0	1	1
COUNTRIES	50	0	0	0	0
DEPARTMENTS	27	0	0	0	0
EMPLOYEES	646	0	0	1	1
EMAIL	108	0	0	0	0
FIRST_NAME	108	0	0	0	0
JOB_ID	108	0	0	0	0
LAST_NAME	107	0	0	1	0
PHONE_NUMBER	108	0	0	0	0
RESUME	107	0	0	0	1
JOBS	38	0	0	0	0
JOB_HISTORY	10	0	0	0	0
LOCATIONS	115	0	0	0	0
REGIONS	4	0	0	0	0
Materialized Views					
IX	35	0	0	0	0
OE	20334	0	0	0	0

The Database Scan Report tab contains a toolbar on the top and a result grid under the toolbar.

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" on page 2-13). 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, " Database Migration Assistant for Unicode Icons" on page 2-14.

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 first icon on the toolbar (see Table 2–1, " Database Migration Assistant for Unicode Icons"). You can collapse all descendants by selecting the node and clicking **Collapse All**, the second 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 Chapter 3, "Viewing and Setting Object Properties in the DMU".

By clicking on **Customize Report**, the third 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 Chapter 3, "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 Chapter 6, "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 Chapter 3, "Viewing and Setting Object Properties in the DMU").

The context menu for data dictionary tables 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.

You can use three methods to quickly locate scan results in the Database Scan Report in which you are interested:

- Database Scan Report: Navigating by Status Icons
- Database Scan Report: Filtering
- Database Scan Report: Searching
- Database Scan Report: Exporting to HTML

Database Scan Report: Navigating by Status Icons

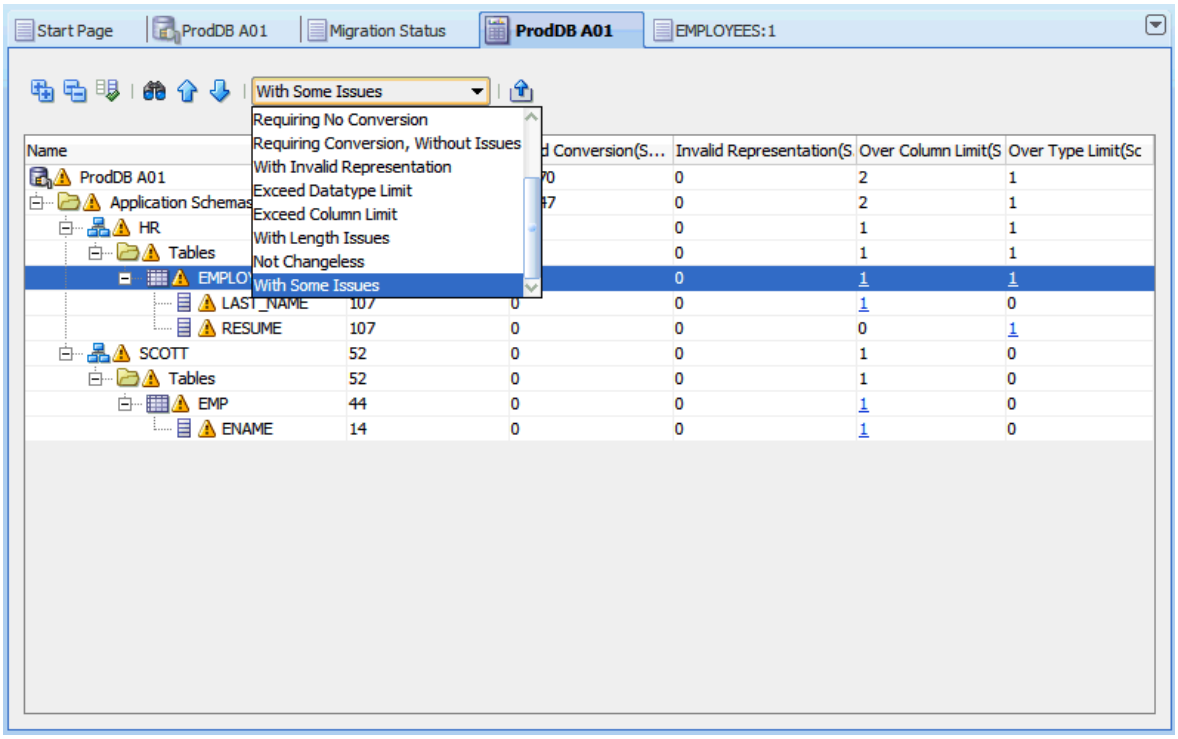
You can locate database columns that contain data with convertibility issues by expanding Database Scan Report nodes that are marked with the yellow triangle icon. This icon signals that one or more columns among descendants of the node have some convertibility issues. Expand a node marked with such an icon, then look through its children to locate child nodes marked with the same icon, expand those nodes, and so on, until you reach the column nodes. The column nodes marked with a yellow

triangle icon describe database 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. The 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.

Database Scan Report: Filtering

A useful feature of the Database Scan Report tab is the filtering function. You can open the drop-down filter list on the toolbar of the tab, as shown in Figure 4–10, "Database Scan Report: Filtering", and select which type of results you would like to see in the report.

Figure 4–10 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.

- 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 Changless

  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.

Database Scan Report: Searching

Another feature of the Database Scan Report tab that makes analyzing scan results easier in large scan reports is the search feature. You initiate a search by clicking one of the following toolbar buttons:

- Find

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

- Find Previous

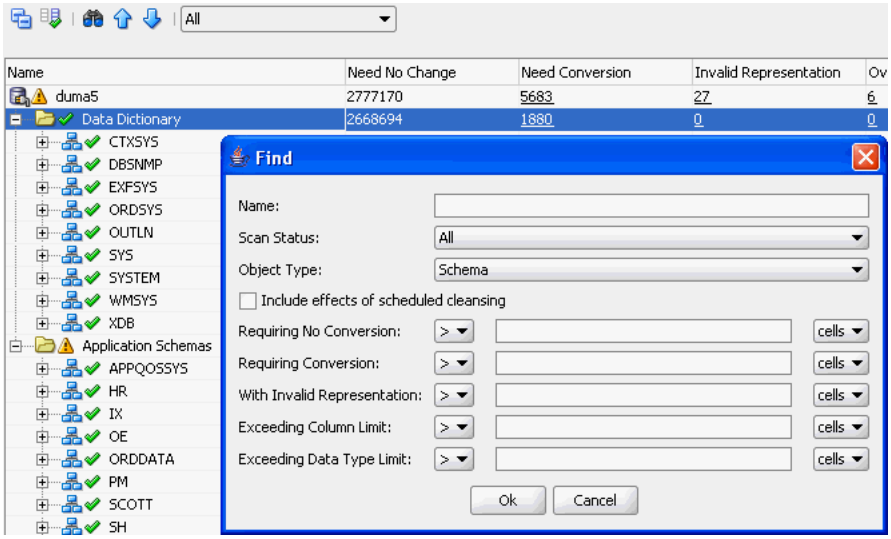
The fifth button on the toolbar highlights the previous instance of the originally searched item.

- Find Next

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

If you select **Find**, a dialog box such as that shown in Figure 4–11, "Database Scan Report: Searching" is displayed.

Figure 4–11 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

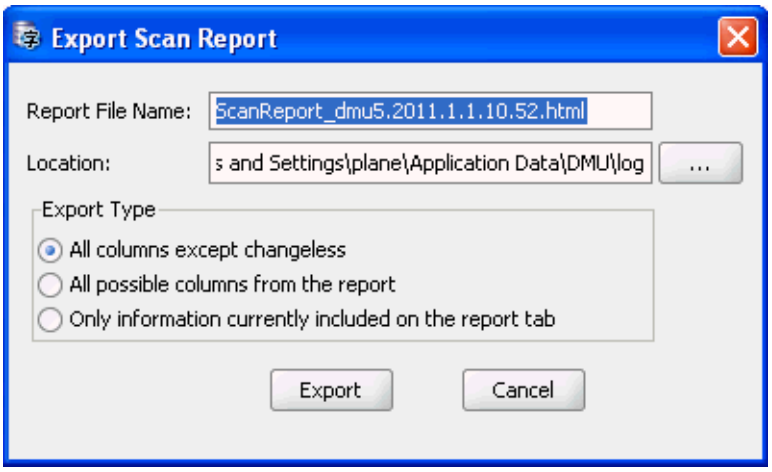
  Select the comparison operator and the value to compare with the Over Type Limit results.

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.

Database Scan Report: Exporting to HTML

If you want to save the scan results for future reference, you can 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 4–12, "Export Scan Report".

Figure 4–12 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.

Cleansing the Data

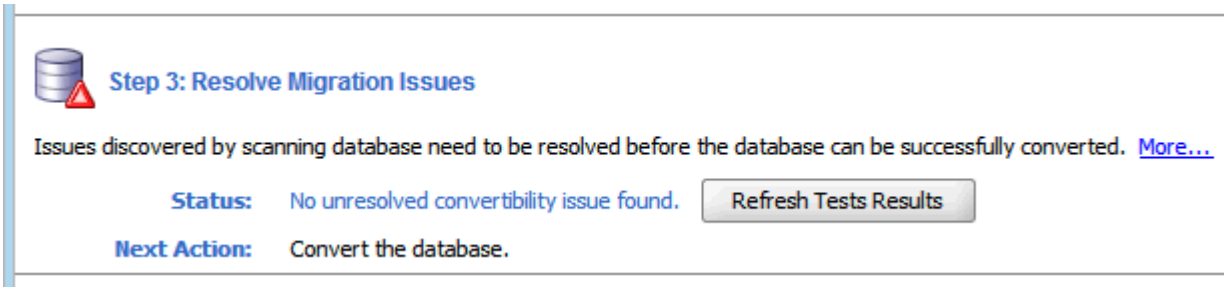
Data issues that have been identified in the scanning process must be resolved before the database can be converted. The DMU does not allow the conversion process to start until all issues have been resolved or explicitly marked as ignorable. The actions that you can take to resolve the convertibility issues are called cleansing actions.

See Chapter 6, "Using the DMU to Cleanse Data" for details regarding cleansing.

Converting the Database

The final step is the actual conversion of your database character data, which you can perform after all issues in the database have been resolved, and the Migration Status tab resembles what is shown in Figure 4–13, "Migration Status Tab - No Unresolved Issues".

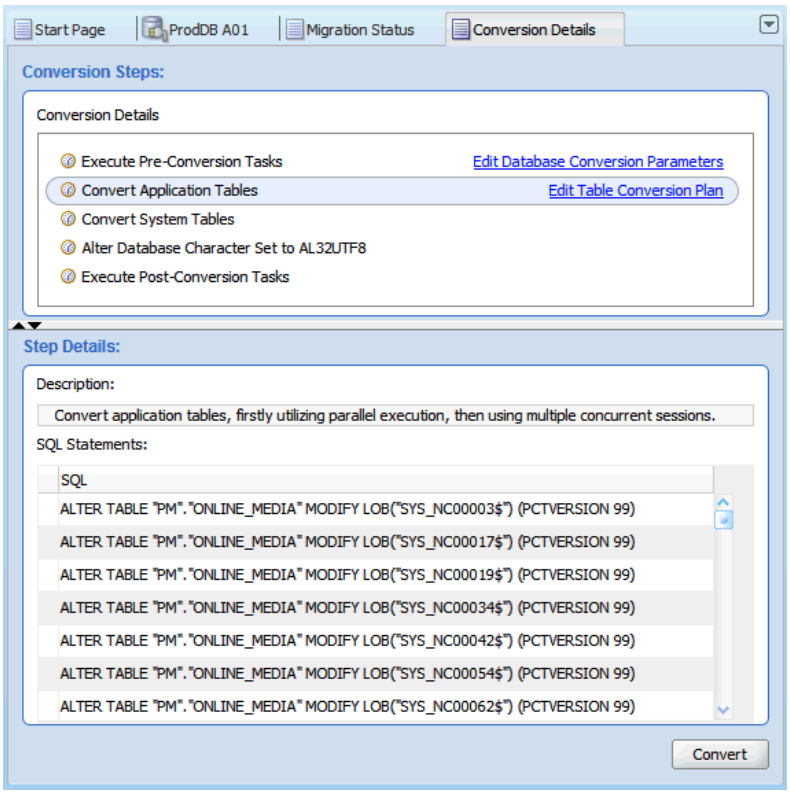
Figure 4–13 Migration Status Tab - No Unresolved Issues



To convert the database:

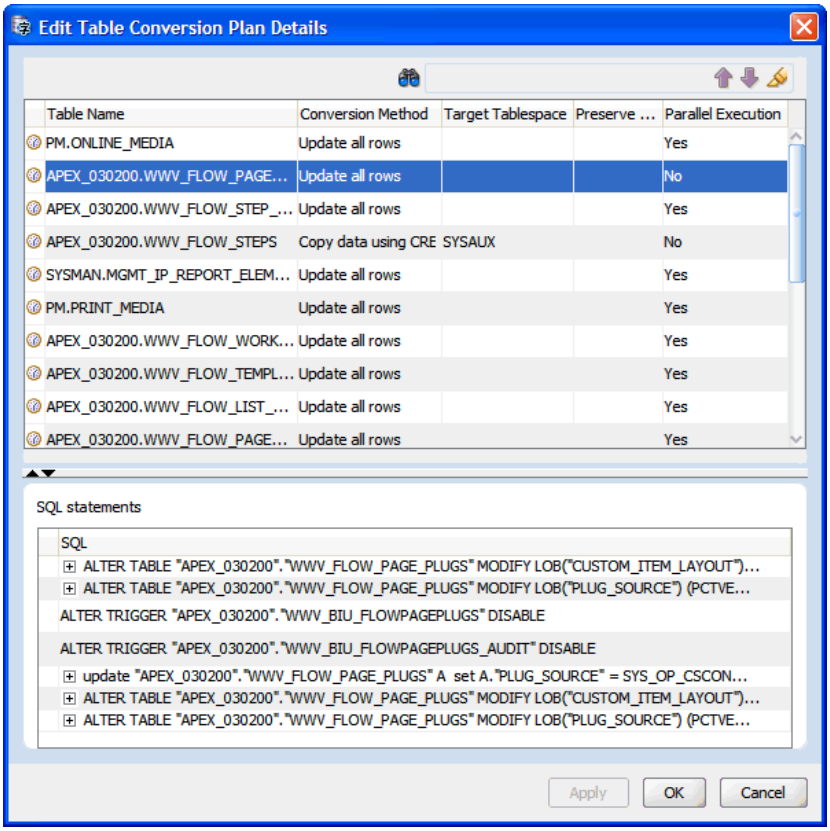
1. To begin the conversion process, select **Convert Database** from the Migration menu or the context menu of the database node in the Navigator pane. The DMU generates a conversion plan, which is a list of SQL statements to run in the conversion phase, and shows it on the Conversion Details tab in the client pane of the DMU window, as shown in Figure 4–14, "Conversion Details Tab". 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 "Conversion Details Tab" on page 4-24.

Figure 4–14 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" on page 3-1 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" on page 3-9 for a description of these parameters.

Figure 4–15 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$SESSION.PROGRAM` in a row returned by the above query contains the name of the Oracle Database executable, in an operating system dependent format, followed by "(J*nnn*)", where *nnn* are three decimal digits, for example "ORACLE.EXE (J000)", then the user session described by the row has been created by the database server itself to execute a job submitted through the PL/SQL package `DBMS_JOB` or `DBMS_SCHEDULER`. You can query the Data Dictionary views `DBA_JOBS_RUNNING` and `DBA_SCHEDULER_RUNNING_JOBS` to identify jobs currently running in the database. Before continuing with conversion, make sure that all the jobs have finished their work and that their sessions have disappeared from `V$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 make sure 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.

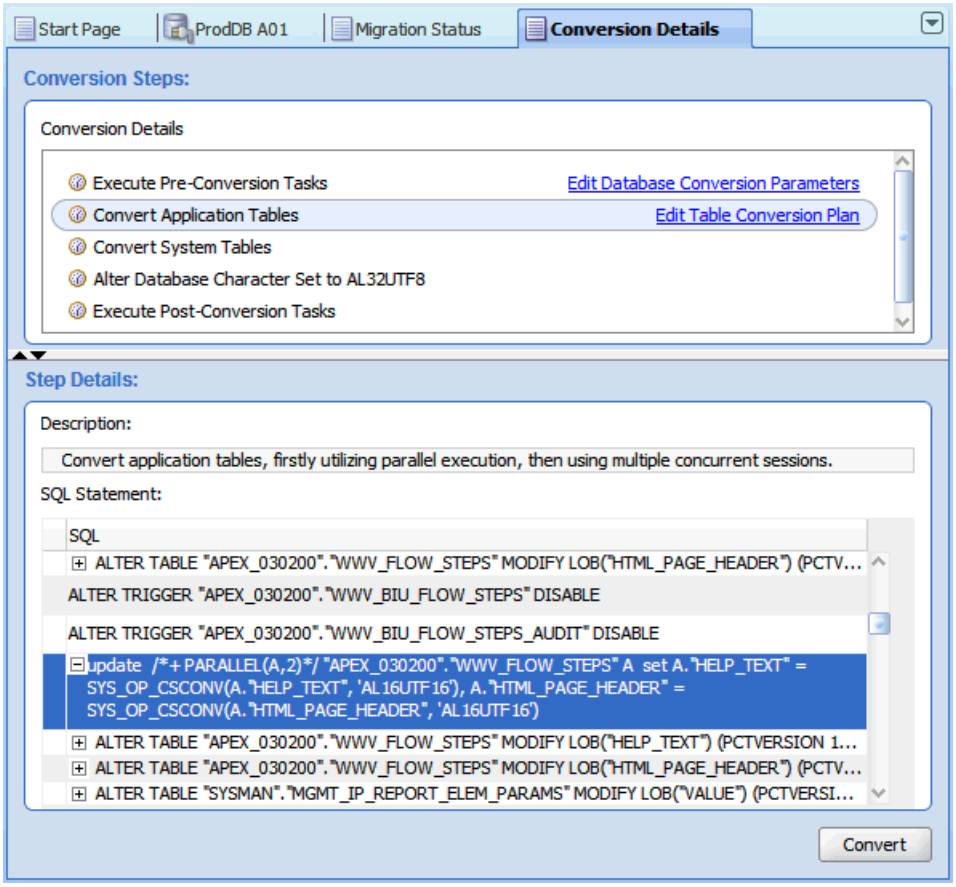
When the conversion starts, the Conversion Details tab changes its form and begins to display progress information for the conversion process. The link Edit Table Conversion Plan becomes View Table Conversion Progress. Clicking **View Table Conversion Progress** opens a table conversion progress dialog 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.

Conversion Details Tab

The Conversion Details tab works in two modes. 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 4–16 Conversion Details Tab: Planning Mode

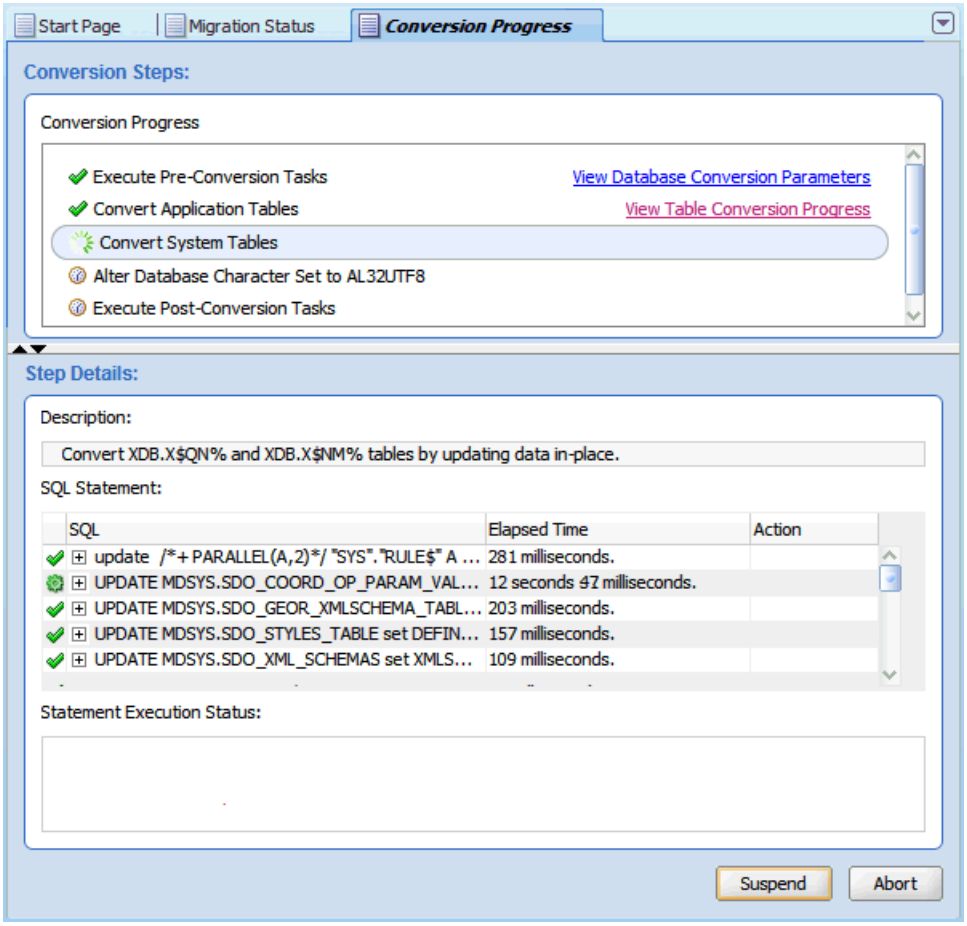


If you click the link **Edit Database Conversion Parameters**, the Converting subtab of the Database Properties tab is opened in a dialog box. See "Column Properties: Converting" on page 3-14 for a description of available parameters.

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" on page 4-27.

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 4–17, "Conversion Details Tab: Monitoring Mode".

Figure 4–17 Conversion Details Tab: Monitoring Mode



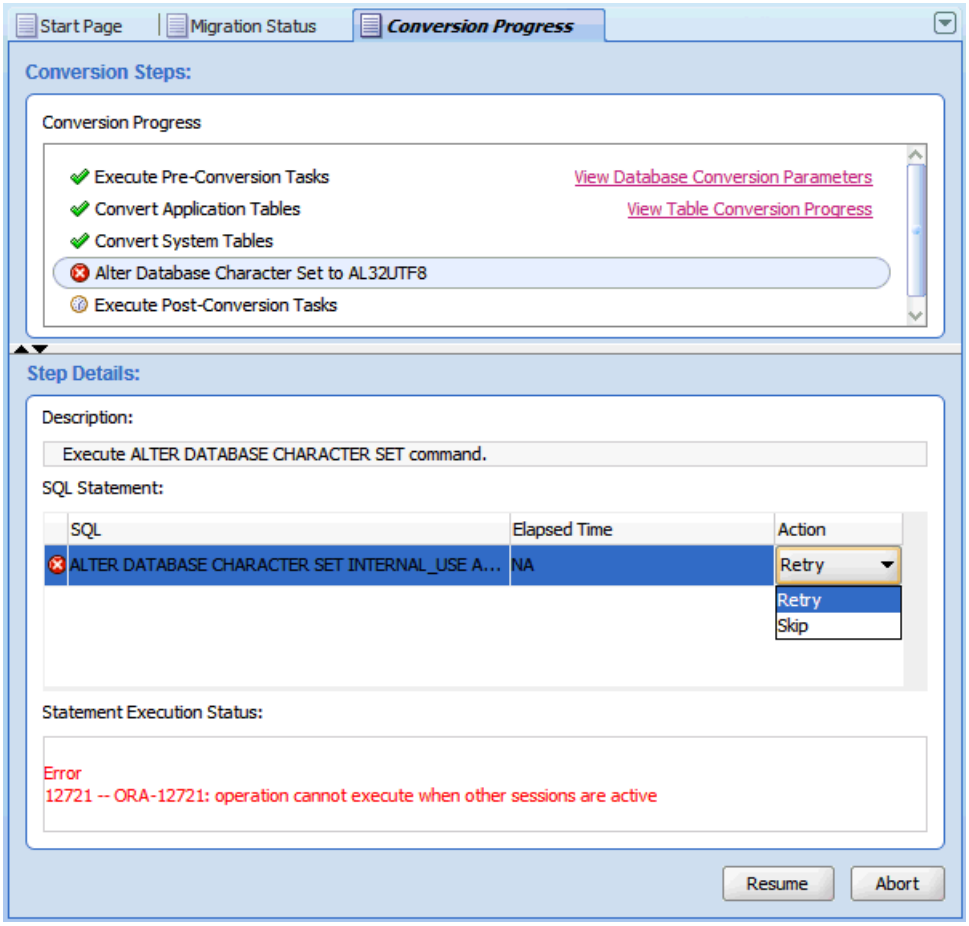
In the monitoring mode, the Convert button is replaced with two buttons Suspend and Abort. If you click **Suspend**, the DMU suspends the conversion process. The Suspend button changes to Resume and the DMU waits for further input. Clicking **Resume** resumes the conversion process. If you click **Abort**, the conversion process is aborted. The database is left in inconsistent state and you have to restore it from a backup database.

In the monitoring mode, the link Edit Table Conversion Plan changes to View Table Conversion Progress. The link opens the Edit Table Conversion Plan Details 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 4–18, "Conversion Errors Tab". The conversion process is suspended.

Figure 4–18 Conversion Errors Tab



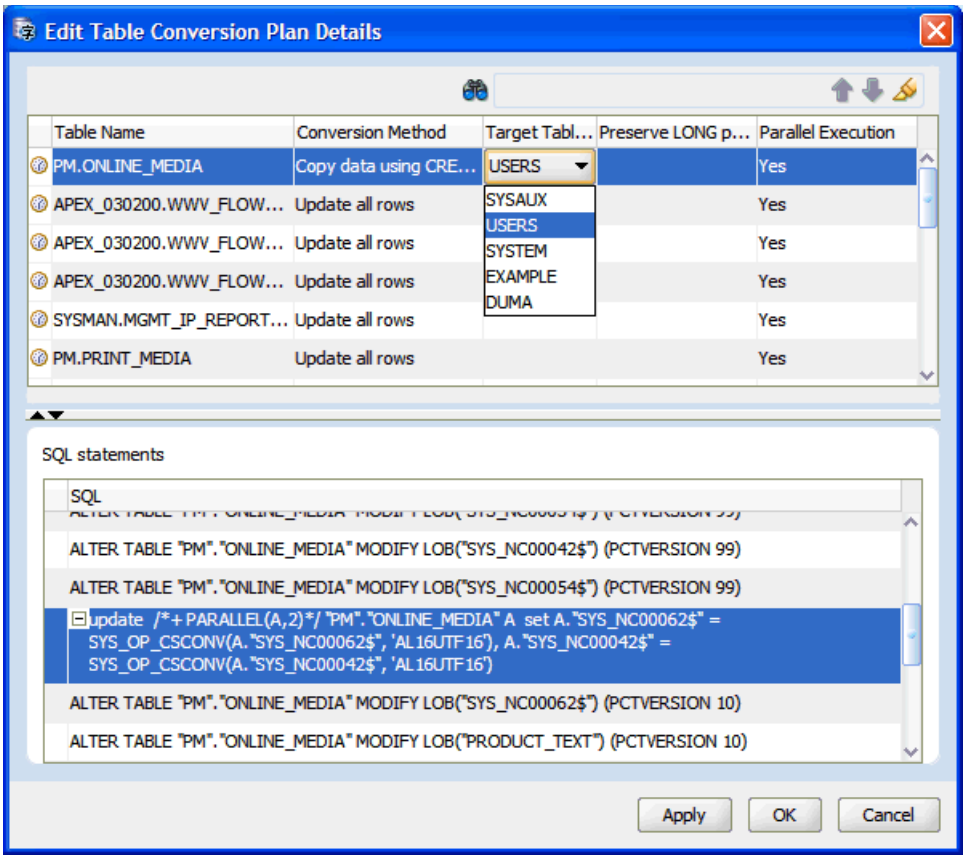
If you click the failing statement, the reported database error messages are displayed in the Statement Execution Status field. You can tell the DMU to retry or skip the statement by selecting the corresponding option from the drop-down list in the Action column of the SQL Statement area. Use the Skip 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 **Resume** 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 migration repository in Unicode Validation Mode by selecting **Convert Repository for Validation Mode** from the Migration menu or you can uninstall the repository by selecting **Uninstall Migration Repository**.

Edit Table Conversion Plan Details Dialog

The Edit Table Conversion Plan Details dialog box, show in Figure 4–19, enables you to modify conversion parameters for individual tables. 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 4–19 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" on page 3-11. 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 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**.

Validating Data as Unicode

In addition to migrating a database to Unicode, the DMU can validate the contents of an existing AL32UTF8 or UTF8 database. Such a database 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 data you have now.

The Database Migration Assistant for Unicode can be used 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. The DMU activates the migration mode automatically, if the current database character set is neither AL32UTF8 nor UTF8. It activates the validation mode, if the current database character set is AL32UTF8. If the current database character set is UTF8, the first page of the Repository Installation Wizard (see "Installing the Migration Repository" on page 2-8) gives you the choice between migration to AL32UTF8, the recommended Unicode character set, and activation of the validation mode. The relevant page of the wizard is shown in Figure 4–21, "Selection of the Operation Mode". To activate the validation mode after you have migrated your database to Unicode, reinstall the migration 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.

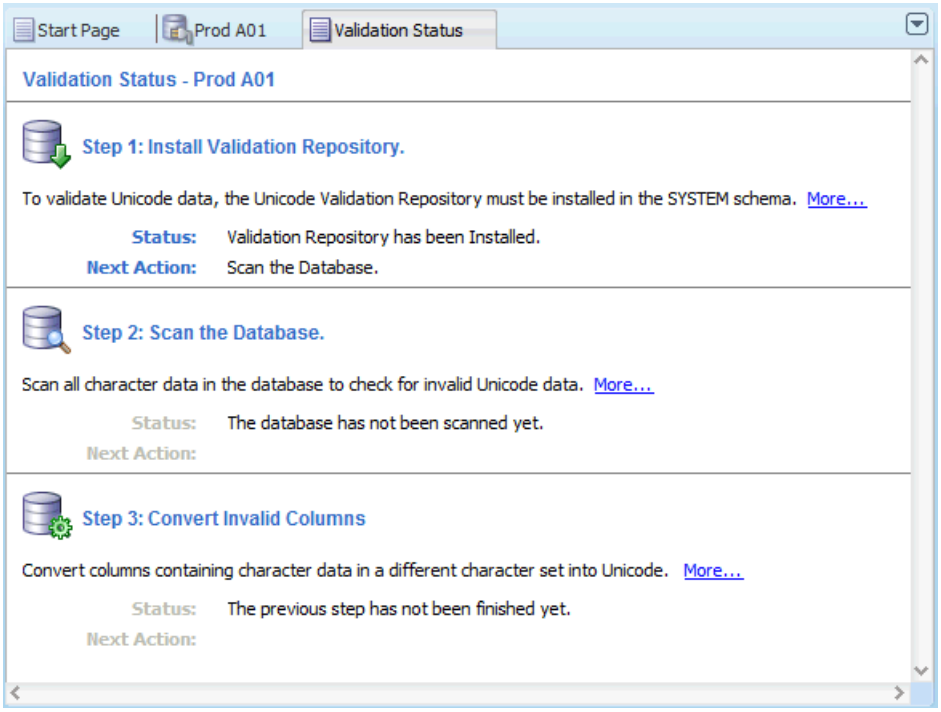
Fixing the data should usually be accompanied by fixing an application configuration problem that caused the invalid data to be stored in the database. Client configuration issues, usually a pass-through configuration one, are the most common reason for incorrectly encoded character data being stored 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.

Introduction to the User Interface in Validation Mode

The DMU user interfaces in validation and migration modes are similar. The following are the main differences:

- The Migration Status tab becomes the Validation Status tab, shown in Figure 4–20.
- 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 4–20 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.

How to Validate 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 needs to 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".

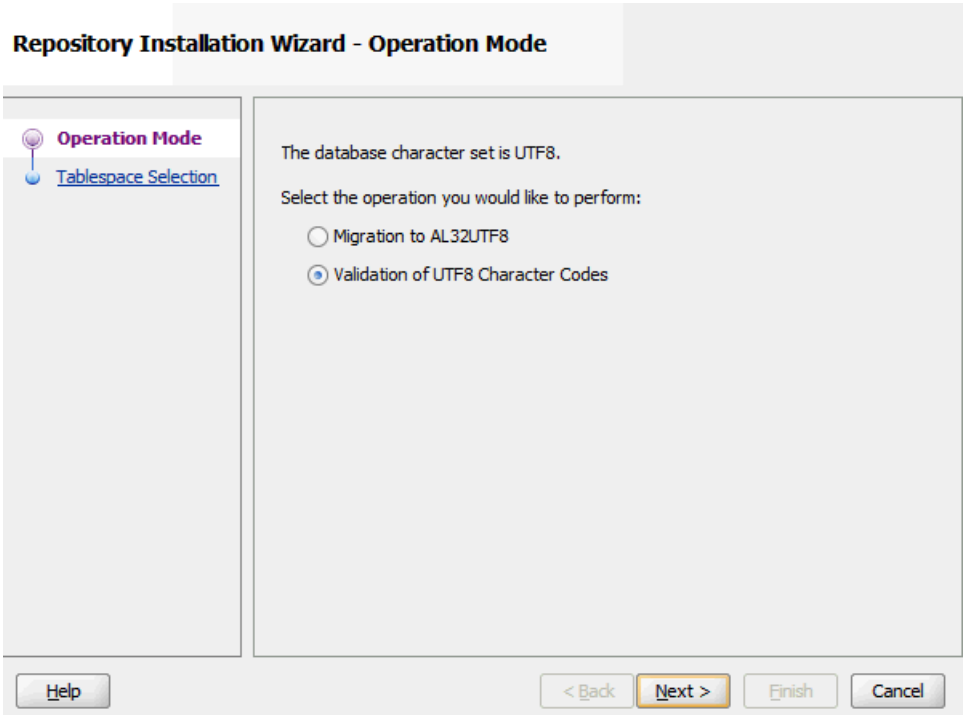
You initiate scanning and view the results as they were in the migration mode. See "Scanning the Database" on page 4-4 for more details.

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" on page 6-25 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" on page 6-25 and "Setting the Assumed Character Set" on page 6-7 for more information.

Figure 4–21 Selection of the Operation Mode



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 Chapter 6, "Using the DMU to Cleanse Data" and in particular "Modifying Columns" on page 6-11 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.

5

Advanced Topics in the DMU

This chapter illustrates various advanced topics that you should consider when working with the Database Migration Assistant for Unicode. It includes:

- Excluding Columns and Tables From Migration
- Handling Non-Accessible Data
- Migrating Data Dictionary Contents
- Working with Multilingual Columns
- Advanced Convertibility Issues
- Adapting Applications for Unicode Migration

Excluding Columns and Tables From Migration

In certain situations, you may want to exclude selected columns or tables from scanning or conversion steps of the migration process. Situations in which this may be justified are the following:

- You need to solve a problem of multiple character sets in a single column (see "Working with Multilingual Columns" on page 5-12).
- 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" on page 5-2 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" on page 3-12) 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.

Handling Non-Accessible Data

Certain types of data in a database might be inaccessible to applications. The data may be non-updatable or it may be 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:

- Read-Only Tables Considerations
- Read-Only Tablespaces Considerations
- Offline Tablespaces and Data Files Considerations
- Working With External Tables

Read-Only Tables Considerations

All Oracle Databases since release 11.1 support the read-only mode for database tables. Thus, users who want table contents to be protected against updates can alter a table to read-only mode. A read-only table can be queried, its constraints modified, its segment moved, shrunk, or expanded, but none of its column values can be modified in any way and no new rows can be added. Users 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, thus, their contents could not be converted if required 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.

Read-Only Tablespaces Considerations

A tablespace can be put into read-only mode. This prevents any data stored in it from being updated. Data files of read-only tablespaces could be put on read-only media, such as DVD-ROM, and moved to jukeboxes, which are used as cheaper storage for seldom accessed archived data. The data files might also remain 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. As read-only tablespaces cannot change, no further backups are required. This significantly reduces backup time of the very large databases.

If any segment of a table, including partition, `CLOB`, `VARRAY`, and IOT overflow segments, contains data that requires conversion, and 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 the data is still on standard disk devices, put the tablespaces back into read/write mode, convert them with the rest of the database, put into read-only mode again and 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.

Offline Tablespaces and Data Files Considerations

A tablespace or selected data files in a tablespace may be put into offline mode. Any 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.

Working With External Tables

An external table is a table whose data resides in files outside of the database. 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.
- The character set of an `ORACLE_LOADER` file may be specified in the `CHARACTERSET` parameter in the access parameters clause of the external table definition. If the parameter is not specified, 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, 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, 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, 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, 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.

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" on page 1-10 and "Cleansing Scenario 3: Cleansing Invalid Representation Issues" on page 6-25.

The following sections describe actions that you can perform to cleanse the external tables from various reported issues.

Cleansing Length Issues

If the scan report shows length issues in external table contents, 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.

Correcting Character Set Declaration of ORACLE_LOADER Files

If character values of an external table are read by the `ORACLE_LOADER` driver in a pass-through configuration, that is, the declared character set of data files and the database character set are the same, but the real character set of data contents 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 – as described in "Setting the Assumed Character Set" on page 6-7 – 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.

Correcting Character Set Declaration of ORACLE_DATAPUMP Files

If character values of an external table are read by the `ORACLE_DATAPUMP` driver in a pass-through configuration, that is, in the character set configuration described in the previous section, you should make sure 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;
```

Fixing Corrupted Character Codes

If analysis of invalid binary representation issues in an external table shows that there are only some corrupted character codes in some character values – usually due to a user error, an application defect, or a temporary configuration problem – the values should be corrected in the source database and the affected files re-exported or unloaded 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.

Handling Binary Data

Invalid binary representation issues in an external table may also 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.

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.

Migrating Data Dictionary Contents

The DMU classifies tables as belonging to the data dictionary based on the schema that owns them. Schemas that the DMU considers to be in the data dictionary are those displayed under the Data Dictionary node in the Navigator panel – see "Introduction to the DMU Interface and Navigation" on page 2-2. If you create your own table in a data dictionary schema, such as SYS or SYSTEM, the DMU will treat 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. Therefore, the DMU handles data dictionary tables differently from other tables in the database, as described in the following sections.

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" on page 4-17.

The View Data tab, described in "Viewing Data" on page 6-9, 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 "Converting Data Dictionary Tables" on page 5-10, 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 "Converting Data Dictionary Tables" on page 5-10 for more details.

Cleansing Data Dictionary Tables

As Oracle neither supports altering structure of data dictionary tables nor updating their contents, the DMU does not allow cleansing actions on such tables. You can set the assumed character set of columns of the data dictionary tables but 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.

Cleansing Data Length Issues

If data length issues are reported for data dictionary contents, that is, the metadata, 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, so you must cleanse the issues even if you plan to use alternative conversion methods, such as moving data with the Data Pump utilities.

The most common length issues are object names becoming longer than allowed after conversion. As the usual length limit for an identifier is only 30 bytes, 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.

To shorten an identifier, you must rename the corresponding database object with an appropriate SQL statement or a PL/SQL package call. Some objects can be just renamed but most have to be dropped and re-created under the new name. When you drop an object, some dependent objects may be dropped along. You must re-create them as well.

For example, you can rename a table, a view, a sequence, or a private synonym using the SQL statement `RENAME`. You can rename a table column using the SQL statement `ALTER TABLE RENAME COLUMN`. But you cannot simply rename a 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. As 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 a lot of objects. The Data Pump utilities and the Metadata API may be helpful in such case.

> **See Also:** *Oracle Database Utilities* for more information about the Data Pump utilities and the Metadata API

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 "Identifying Metadata" on page 5-9 for suggestions about identifying right statements to cleanse data dictionary issues.

Cleansing Invalid Binary Representation Issues

A common reason for invalid binary representation of data is the pass-through scenario, described in "Invalid Binary Storage Representation of Data" on page 1-10. If SQL statements or PL/SQL calls are issued in such a configuration, it is possible to create and use a database object that is named using or contains characters not valid or not having the expected meaning in the database character set. For example, a seemingly senseless table name in a WE8MSWIN1252 database might be interpreted as appearing correct on a Japanese JA16SJIS client, or, 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 is a seldom encountered issue as 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 – see "Database Properties: General" on page 3-1. Otherwise, use the same approach that is described in "Cleansing Data Dictionary Tables" on page 5-8 to 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.

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 – see "Viewing Data" on page 6-9. 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. These scripts define data dictionary views that are documented in *Oracle Database Reference*. 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 Web site.

Once you identified the metadata that has the reported convertibility issues, refer to Oracle documentation to identify the right procedure to change this metadata.

Converting Data Dictionary Tables

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 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
  - 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 tables and columns listed above 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.

Data Dictionary Tables That Are Ignored

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 Performance Tuning Guide* 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" on page 4-29.

Handling Automatic Workload Repository Tables

The SYS schema contains a number of tables with names beginning with WRI$, WRH$%, and WRR$_, which comprise the Automatic Workload Repository (AWR). In addition to historical object statistics, mentioned in "Data Dictionary Tables That Are Ignored" on page 5-11, this repository stores snapshots of vital system statistics, such as those visible in various fixed views, for example, V$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, recreate 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.

Working with Multilingual Columns

As mentioned in "Cleansing Incorrect Character Set Declaration" on page 6-28, you may find out while analyzing contents of a column in the Cleansing Editor that none of the assumed character sets set for the column makes all values in the column appear correctly at the same time, but each of the values does seem to be correct in one of the selected character sets. This 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" on page 3-14.
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 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.

Advanced Convertibility Issues

This section describes less frequently encountered convertibility issues that are not currently handled automatically by the DMU and might require that you perform additional scanning and cleansing steps outside of the tool.

It contains these topics:

- Convertibility Issues: Uniqueness Validation
- Convertibility Issues: Index Size
- Convertibility Issues: Partition Range Integrity
- Convertibility Issues: Objects in the Recycle Bin
- Convertibility Issues: PL/SQL Local Identifiers Greater Than 30 Bytes

Convertibility Issues: Uniqueness Validation

There are two situations when some rows in a table might no longer satisfy a unique or primary key constraint after database contents have been converted to Unicode:

- 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 *Oracle Database Globalization Support Guide* for more information about this 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.

Convertibility Issues: Index Size

The maximum size of an index key in an Oracle database index – that is, the sum of maximum byte lengths of all key columns plus rowid length plus the number of length bytes – cannot exceed the data block size in the tablespace of the index minus around 25% overhead.

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.

In this release, the DMU does not proactively verify index key lengths that change in the conversion step. Therefore, "ORA-01450: maximum key length (maximum) exceeded" or "ORA-01404: ALTER COLUMN will make an index too large" may be reported during conversion from an `ALTER TABLE`, `CREATE INDEX`, or `ALTER DATABASE CHARACTER SET` statement.

Before attempting conversion, you should review all indexes defined on `VARCHAR2` and `CHAR` columns that have a lengthening cleansing actions scheduled or that use character length semantics to verify that they are not affected by this issue. The easiest approach is to test the migration on a copy of the original database. The copy should include the DMU repository with all planned cleansing actions. The actual application data does not affect the test so you can truncate all large convertible tables to shorten the test time.

Convertibility Issues: Partition Range Integrity

While the DMU does not support converting a database in which any partition bounds require conversion, you may want to migrate such a database using Data Pump

utilities. While preparing for the migration process, you should 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 and, if required, adjust their partition definitions, so that rows are still distributed according to your expectations after the database migration to Unicode.

While very improbable, partition range bounds and partition list values may also suffer from the uniqueness issue described in "Convertibility Issues: Uniqueness Validation" on page 5-14. If such an issue is encountered for a partitioned table, the table cannot be imported successfully without partition definitions being adjusted accordingly.

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. Their 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.

Convertibility Issues: PL/SQL Local Identifiers Greater Than 30 Bytes

While the DMU does not support converting names of PL/SQL stored modules, that is, stored procedures, stored functions and packages, it 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). 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.

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.

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

Changes to SQL and PL/SQL Code

Because SQL and PL/SQL code runs inside the database, its processing character set, that is, the character set in which the processed data is encoded. In this case, the database character set - always changes after a database migration to Unicode. 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.

6

Using the DMU to Cleanse Data

This chapter provides a reference for elements of the user interface of the Database Migration Assistant for Unicode (DMU) used during cleansing and describes how to use them to perform various tasks.

This chapter contains the following sections:

In certain specific cases, two column properties allow you to ignore data issues as described in "Ignoring Convertibility Issues" on page 6-15.

The cleansing actions can be immediate or scheduled. An immediate action is performed directly on the database contents and its effects are immediately visible. A scheduled cleansing action is registered in the migration 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.

Cleansing Data

The Cleansing Editor tab is the starting point of most cleansing actions on a table. You can open it 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.

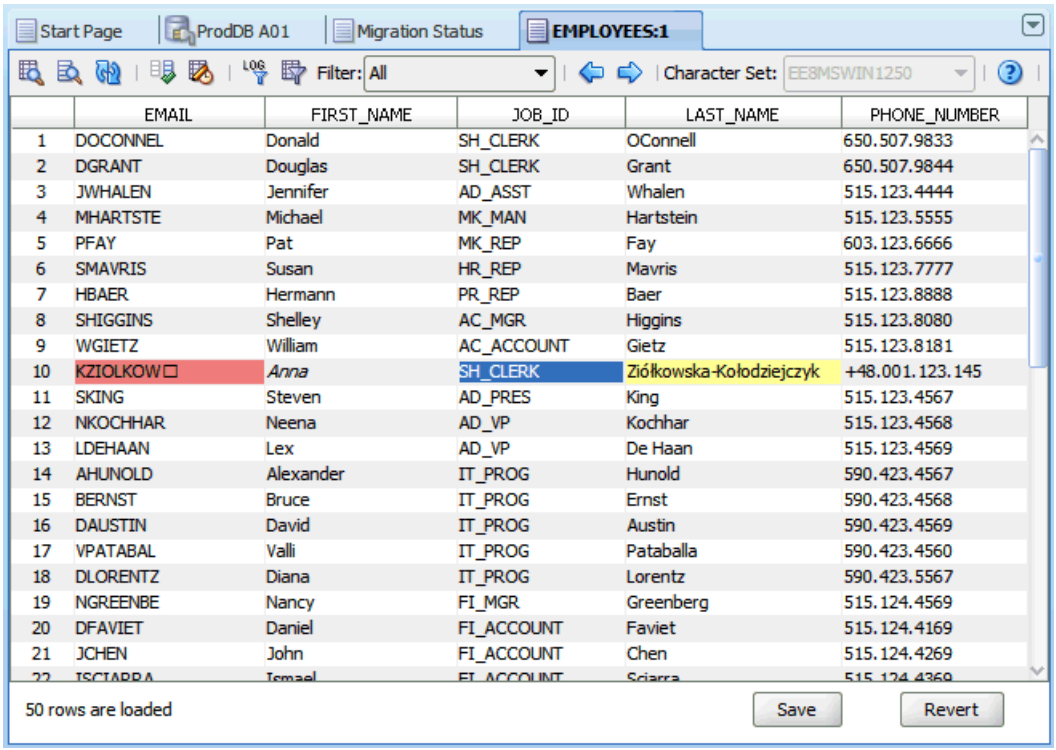
The Cleansing Editor tab contains a toolbar, an editor grid, and two buttons, Save and Revert. The editor grid displays contents of the table for which the Cleansing Editor was opened. This table is called the edited table later in this guide. The columns of the grid correspond to character data type 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, "Cleansing Editor" 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" on page 6-4 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 let the Cleansing Editor 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" on page 4-17 for more information.

Figure 6–1 Cleansing Editor



	EMAIL	FIRST_NAME	JOB_ID	LAST_NAME	PHONE_NUMBER
1	DOCONNEL	Donald	SH_CLERK	OConnell	650.507.9833
2	DGRANT	Douglas	SH_CLERK	Grant	650.507.9844
3	JWHALEN	Jennifer	AD_ASST	Whalen	515.123.4444
4	MHARTSTE	Michael	MK_MAN	Hartstein	515.123.5555
5	PFAY	Pat	MK_REP	Fay	603.123.6666
6	SMAVRIS	Susan	HR_REP	Mavris	515.123.7777
7	HBAER	Hermann	PR_REP	Baer	515.123.8888
8	SHIGGINS	Shelley	AC_MGR	Higgins	515.123.8080
9	WGIETZ	William	AC_ACCOUNT	Gietz	515.123.8181
10	KZIOLKOW□	*Anna*	SH_CLERK	Ziółkowska-Kołodziejczyk	+48.001.123.145
11	SKING	Steven	AD_PRES	King	515.123.4567
12	NKOCHHAR	Neena	AD_VP	Kochhar	515.123.4568
13	LDEHAAN	Lex	AD_VP	De Haan	515.123.4569
14	AHUNOLD	Alexander	IT_PROG	Hunold	590.423.4567
15	BERNST	Bruce	IT_PROG	Ernst	590.423.4568
16	DAUSTIN	David	IT_PROG	Austin	590.423.4569
17	VPATABAL	Valli	IT_PROG	Pataballa	590.423.4560
18	DLORENTZ	Diana	IT_PROG	Lorentz	590.423.5567
19	NGREENBE	Nancy	FI_MGR	Greenberg	515.124.4569
20	DFAVIET	Daniel	FI_ACCOUNT	Faviet	515.124.4169
21	JCHEN	John	FI_ACCOUNT	Chen	515.124.4269
22	ISCIARRA	Ismael	FI_ACCOUNT	Sciarra	515.124.4369

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" on page 6-9. 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" on page 6-11, "Scheduling Column Modification" on page 6-13, "Modifying Attributes" on page 6-13, or "Scheduling Attribute Modification" on page 6-14. 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.

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" on page 6-7.

Cleansing Data: Using the Toolbar

The items on the Cleansing Editor toolbar in the order shown in Figure 6–1, "Cleansing Editor" are:

- Rescan Table

  Opens the Scan Wizard (see "Scanning the Database with the Scan Wizard" on page 4-6) 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" on page 6-4.

- Use Scan Log to Filter Data

  Filters data using rowids collected during scanning.

  See "Filtering Data" on page 6-5.

- Customize Filtering Condition

  Displays a dialog box where you can customize the filtering condition. See "Filtering Data" on page 6-5.

- 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" on page 6-5.

- 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" on page 6-7.

- Cleansing Advice

  Displays a help page.

Cleansing Data: Color Highlighting

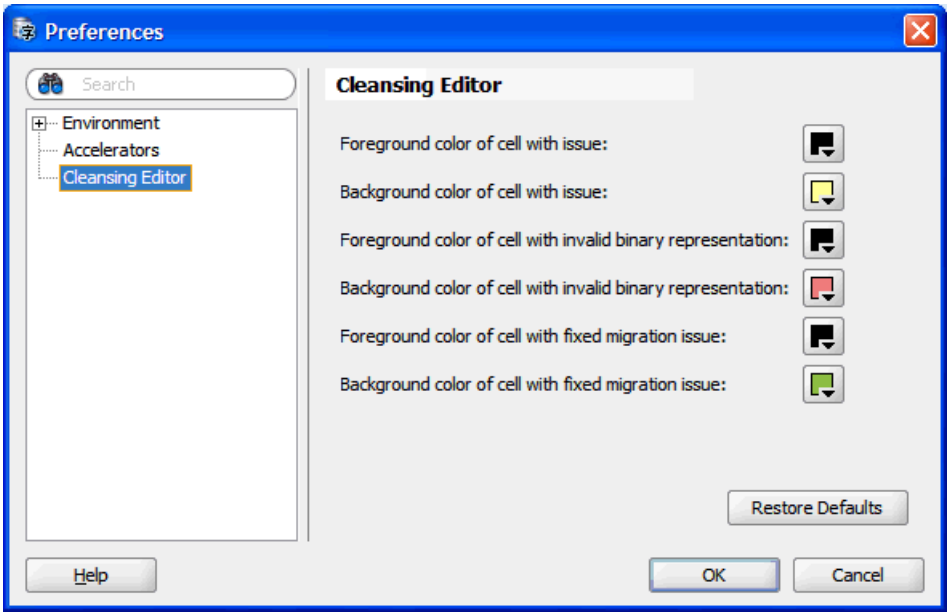
Cells that contain data with convertibility issues are highlighted by using colors different from the standard colors of other cells. By default, cells with length issues, that is, exceeding column limit or exceeding data type limit after conversion, are displayed on yellow background. Cells with invalid 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, "Preferences: Cleansing Editor". You can open the Preferences dialog box by selecting **Preferences** from the Tools menu. The first two colors are for cells with length expansion issues, the next two colors are for cells with invalid binary representation, and the last two colors are for issues that have been cleansed by a scheduled cleansing action.

Figure 6–2 Preferences: Cleansing Editor



Filtering Data

To make working with large tables easier, the Cleansing Editor tab provides a powerful filtering feature. You can filter rows included in the editor grid on the types of the convertibility issues that they contain or on an arbitrary SQL condition.

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 migration 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.

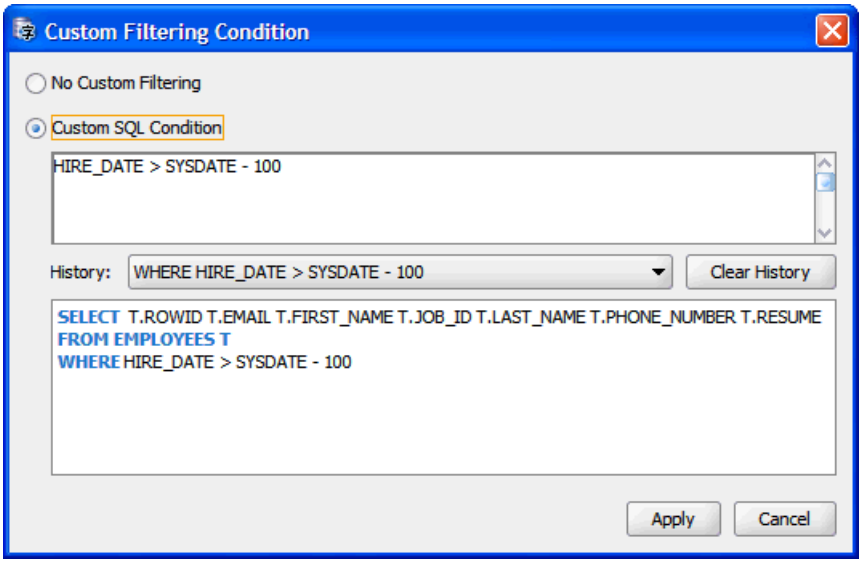
You tell the DMU to use the collected rowids by clicking the **Use Scan Log to Filter Data** button on the toolbar of the Cleansing Editor. This button is enabled only if rowid collection was switched on during the last scan of the table. You can set the rowid collection level either on the Scanning subtab of the Table Properties tab (see "Table Properties: Scanning" on page 3-9) or on the Scan Details page of the Scan Wizard (see "Scanning the Database with the Scan Wizard" on page 4-6). 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.

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, "Custom Filtering Condition". 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



Setting the Assumed Character Set

A common source of data with invalid representation issues, as described in "Invalid Binary Storage Representation of Data" on page 1-10, are applications that use the pass-through configuration to store data directly in the client character set without any conversion and without caring what the database character set is. The DMU helps you to identify such client character sets, the real character sets of the data, and use them as the source character set for conversion, if they are different from the database character set. This allows incorrectly stored data to be correct converted to the target Unicode encoding. The standard conversion from the database character set would corrupt the data and make it illegible. The declared character set of data in a column that you can select in the DMU is called the assumed column character set.

The assumed column character set is used not only for conversion to Unicode but also when column data is interpreted for display in the Cleansing Editor, Edit Data dialog box, or Show Data dialog box. The displayed data looks correctly to a human reader only if its interpretation happens according to its real character set or according to 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.) This fact lets you verify the assumed column character set has been chosen correctly, if all data in the column looks correct.

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.

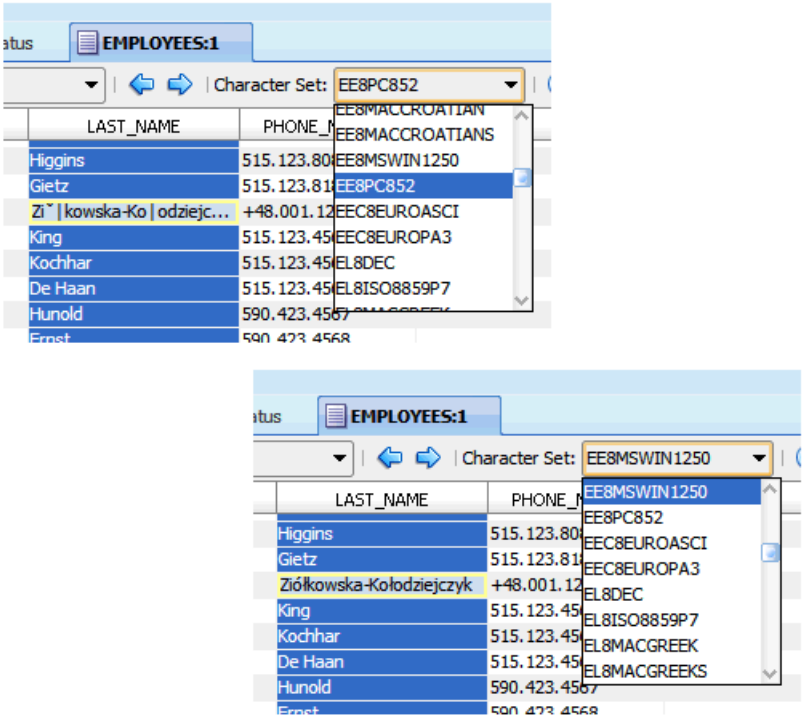
You can identify the real character set of a column by doing either of the following:

- 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).

To set the assumed character set of a column in the Cleansing Editor, select the column by clicking its heading and select the desired character set in the Character Set drop-down list on the toolbar, as shown in Figure 6–4, "Setting Assumed Column Character Set". The contents of the selected column will be redisplayed according to the new interpretation. Try another character set, if the data is illegible. Also, the color highlighting (see "Cleansing Data: Color Highlighting" on page 6-4) will tell you if some bytes in the column cannot be interpreted according to the selected character set.

Figure 6–4 Setting Assumed Column Character Set



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.

To save the selected assumed column character sets permanently in the migration repository for use in any further processing of the columns, click **Save**.

Viewing Data

The Data Viewer is a read-only version of the Cleansing Editor tab, described in "Cleansing Data: Using the Toolbar" on page 6-3. You can open it by selecting **Data Viewer** from the context menu of a data dictionary table node in the Navigator pane or in the Database Scan Report. The Data Viewer is not available for the standard application tables. The Data Viewer looks identical to the Cleansing Editor tab, shown in Figure 6–1, "Cleansing Editor", 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" on page 6-11. 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.

Editing Data

Open the Edit Data dialog box, shown in Figure 6–5, "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 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.

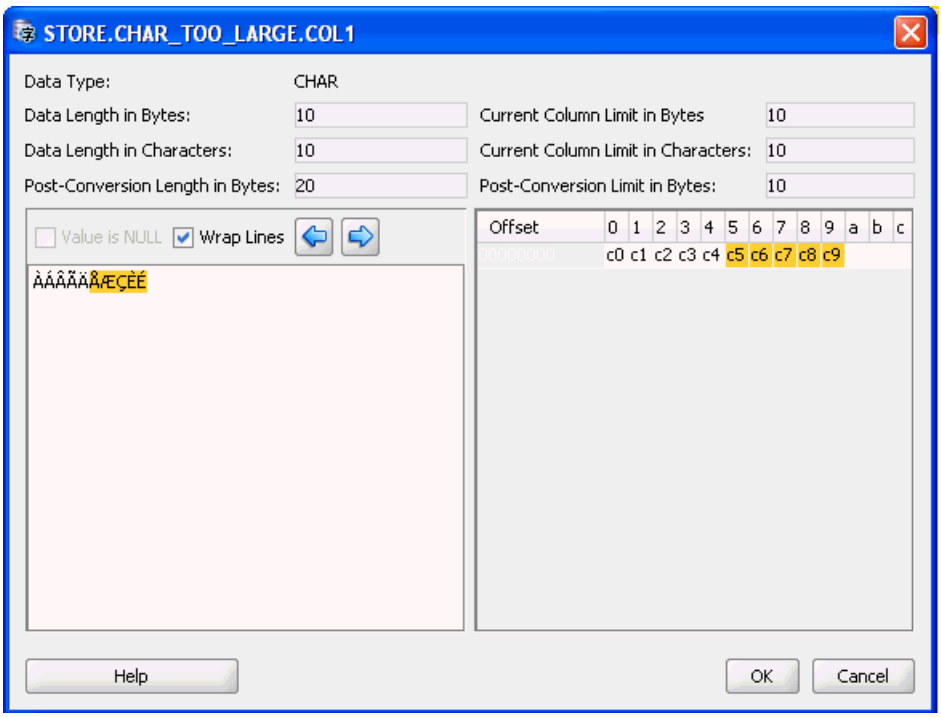
- Post-Conversion Limit in Bytes

  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 6–5 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.

Displaying Data

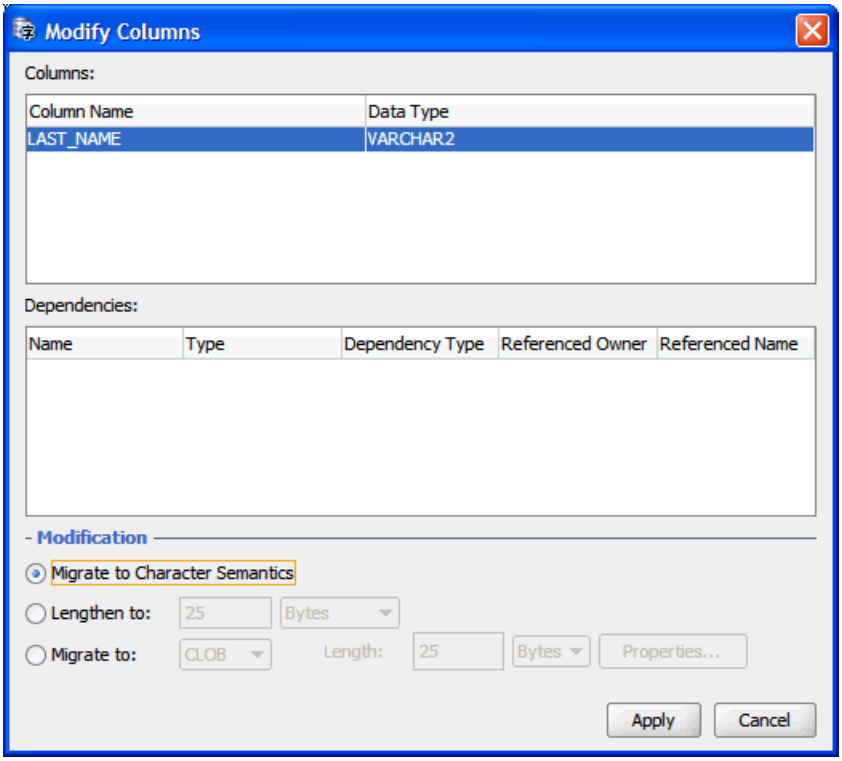
The Show Data dialog box is a read-only version of the Edit Data dialog box. It is used to display values in the data dictionary tables, which you are not allowed to modify directly.

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.

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 6–6. You can open it by selecting **Modify Column** from the context menu of the Cleansing Editor (see "Cleansing Data: Using the Toolbar" on page 6-3).

Figure 6–6 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.

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 migration 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.

You can 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" on page 6-3).

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.

The Modify Attributes dialog box is shown in Figure 6–7. You can open it by selecting **Modify Attribute** from the context menu of the Cleansing Editor (see "Cleansing Data: Using the Toolbar" on page 6-3).

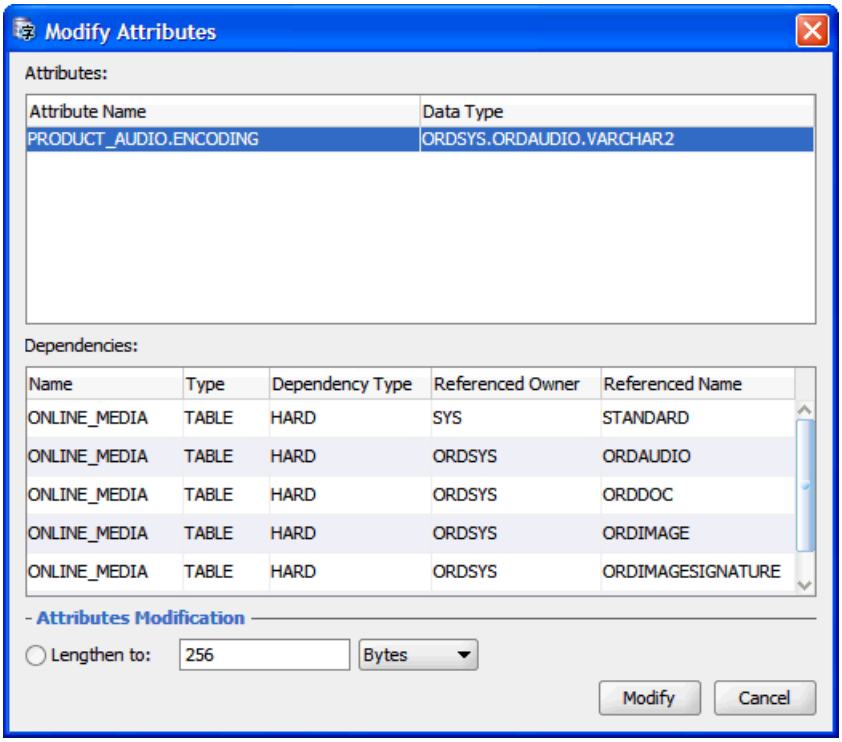
Attributes that will be modified through the Modify Attributes dialog box are displayed in the Attributes list box. The Dependencies list box shows database objects, such as tables, views and PL/SQL procedures, that depend on those attributes and could therefore need updating or recompilation after the 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.

After you have defined the cleansing action, click **Modify**. 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 6–7 Modify Attributes Dialog Box



Scheduling Attribute Modification

The Schedule Attribute Modification dialog box is almost identical to the Modify Attributes dialog box, previously described. 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 migration 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" on page 6-3).

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" on page 3-12.

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:

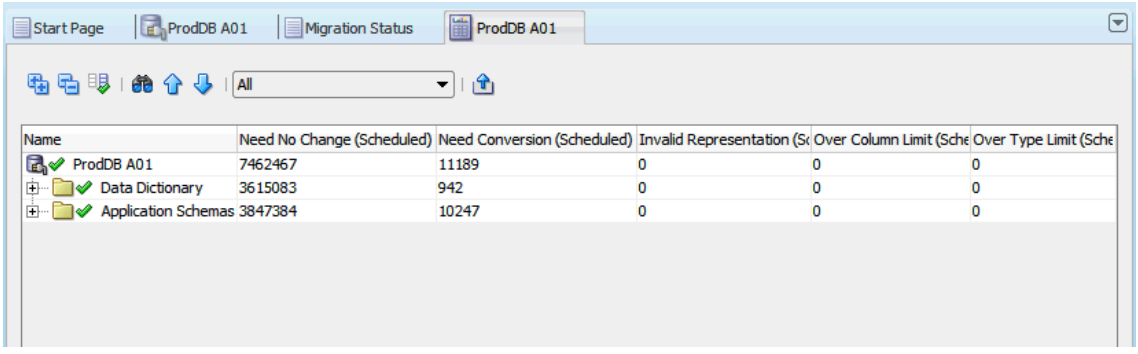
- Contains a mixture of values in multiple character sets, which came in the pass-through configuration from various clients working in different character sets. Such values must be converted manually after the DMU converts the rest of the database and updates the database character set to Unicode.
- 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" on page 5-1 for more information about setting this property.

Cleansing Scenario 1: A Database with No Issues

If there are no issues with data in your database, the Scan Report resemble what is shown in Figure 6–8, "Database Scan Report: No Data Issues".

Figure 6–8 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" on page 2-10) you can proceed to the actual conversion phase of the migration process, which is described in "Converting the Database" on page 4-21.

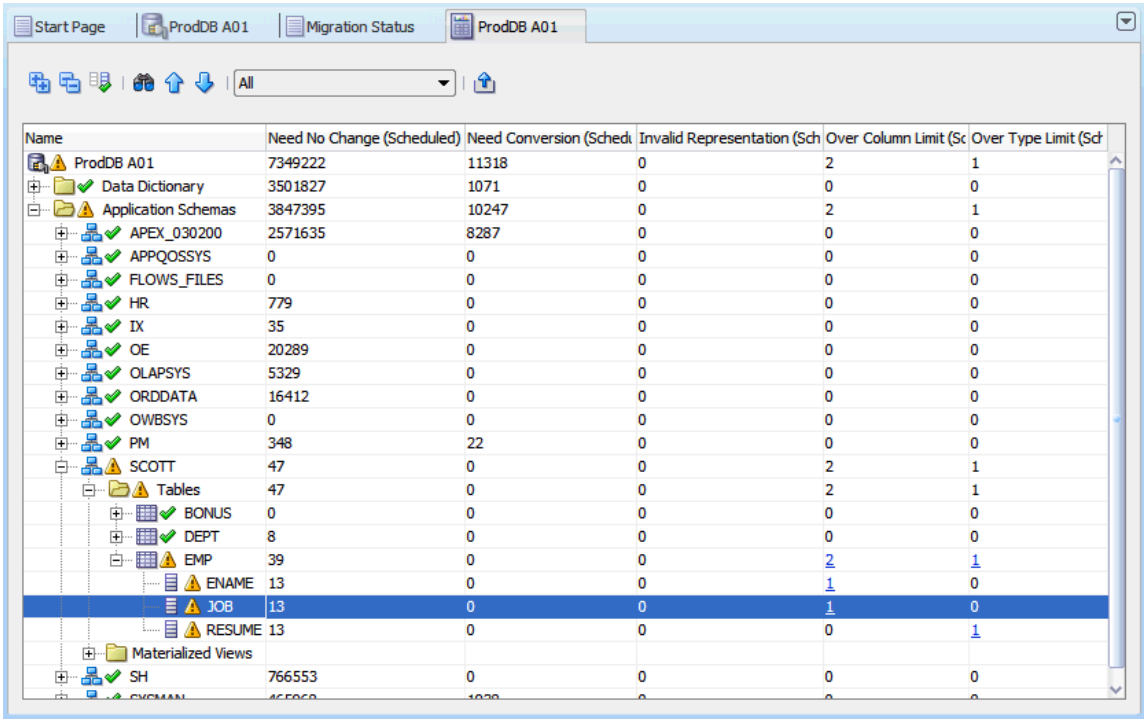
Cleansing Scenario 2: Cleansing Expansion Issues

If there are expansion issues with data in your database, the Scan Report will resemble what is shown in Figure 6–9, "Database Scan Report: Expansion Issues". In addition to the columns Need No Change (Scheduled) and Need Conversion (Scheduled), described in "Cleansing Scenario 1: A Database with No Issues" on page 6-15, 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.

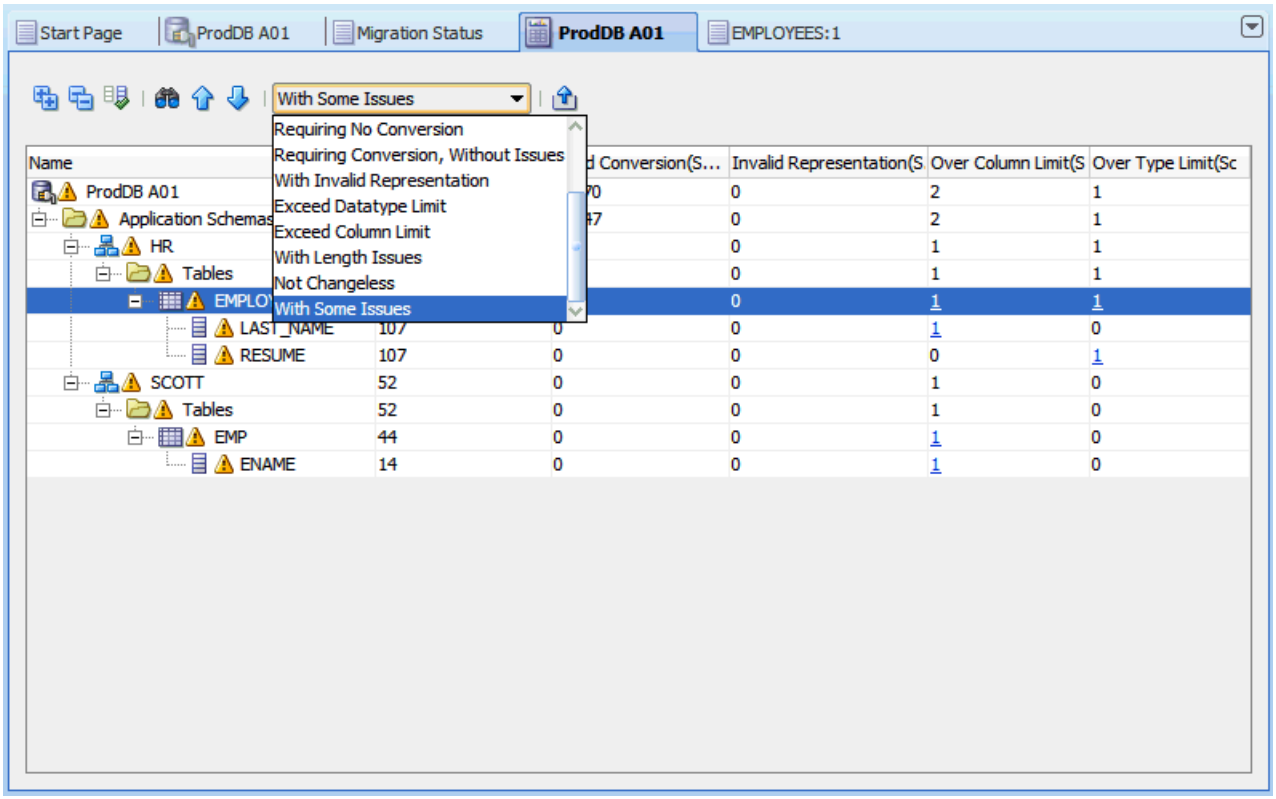
Figure 6–9 Database Scan Report: Expansion Issues



Name	Need No Change (Scheduled)	Need Conversion (Sched	Invalid Representation (Sch	Over Column Limit (Sc	Over Type Limit (Sch
ProdDB A01	7349222	11318	0	2	1
Data Dictionary	3501827	1071	0	0	0
Application Schemas	3847395	10247	0	2	1
APEX_030200	2571635	8287	0	0	0
APPQOSSYS	0	0	0	0	0
FLOWS_FILES	0	0	0	0	0
HR	779	0	0	0	0
IX	35	0	0	0	0
OE	20289	0	0	0	0
OLAPSYS	5329	0	0	0	0
ORDDATA	16412	0	0	0	0
OWBSYS	0	0	0	0	0
PM	348	22	0	0	0
SCOTT	47	0	0	2	1
Tables	47	0	0	2	1
BONUS	0	0	0	0	0
DEPT	8	0	0	0	0
EMP	39	0	0	2	1
ENAME	13	0	0	1	0
JOB	13	0	0	1	0
RESUME	13	0	0	0	1
Materialized Views					
SH	766553	0	0	0	0

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 6–10, "Filtering Scan Results". 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 6–10 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 6–11, "Cleansing Editor Tab: Expansion Issues". 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**.

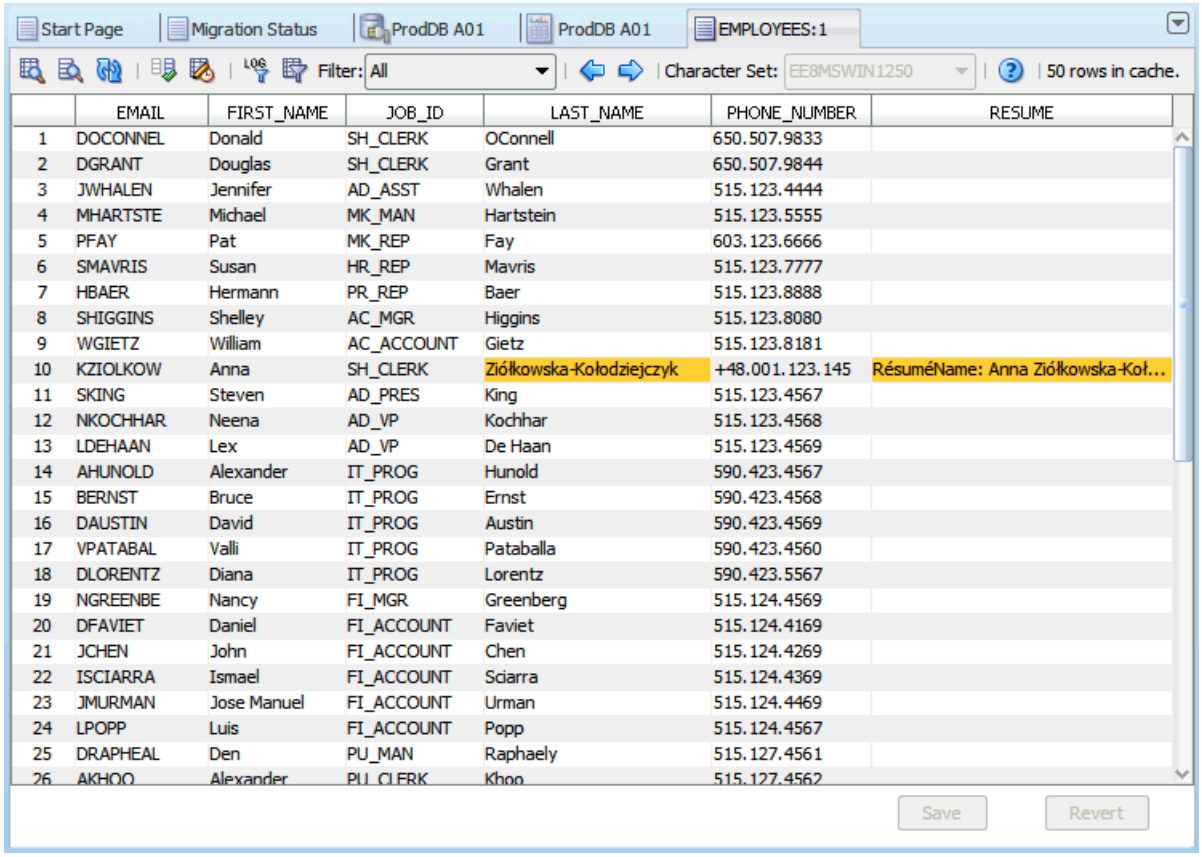
The Cleansing Editor tab shows the contents of the table for which it was opened. By default, the data cells that have length expansion issues are shown on a yellow background. You can change this color in the Preferences dialog box, which can be opened by selecting **Preferences** from the Tools menu.

Various filtering and search features of the Cleansing Editor can help you locate the problematic data cells. See "Filtering Data" on page 6-5 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" on page 4-21.

Figure 6–11 Cleansing Editor Tab: Expansion Issues



	EMAIL	FIRST_NAME	JOB_ID	LAST_NAME	PHONE_NUMBER	RESUME
1	DOCONNEL	Donald	SH_CLERK	OConnell	650.507.9833	
2	DGRANT	Douglas	SH_CLERK	Grant	650.507.9844	
3	JWHALEN	Jennifer	AD_ASST	Whalen	515.123.4444	
4	MHARTSTE	Michael	MK_MAN	Hartstein	515.123.5555	
5	PFAY	Pat	MK_REP	Fay	603.123.6666	
6	SMAVRIS	Susan	HR_REP	Mavris	515.123.7777	
7	HBAER	Hermann	PR_REP	Baer	515.123.8888	
8	SHIGGINS	Shelley	AC_MGR	Higgins	515.123.8080	
9	WGIETZ	William	AC_ACCOUNT	Gietz	515.123.8181	
10	KZIOLKOW	Anna	SH_CLERK	Ziółkowska-Kołodziejczyk	+48.001.123.145	RésuméName: Anna Ziółkowska-Koł...
11	SKING	Steven	AD_PRES	King	515.123.4567	
12	NKOCHHAR	Neena	AD_VP	Kochhar	515.123.4568	
13	LDEHAAN	Lex	AD_VP	De Haan	515.123.4569	
14	AHUNOLD	Alexander	IT_PROG	Hunold	590.423.4567	
15	BERNST	Bruce	IT_PROG	Ernst	590.423.4568	
16	DAUSTIN	David	IT_PROG	Austin	590.423.4569	
17	VPATABAL	Valli	IT_PROG	Pataballa	590.423.4560	
18	DLORENTZ	Diana	IT_PROG	Lorentz	590.423.5567	
19	NGREENBE	Nancy	FI_MGR	Greenberg	515.124.4569	
20	DFAVIET	Daniel	FI_ACCOUNT	Faviet	515.124.4169	
21	JCHEN	John	FI_ACCOUNT	Chen	515.124.4269	
22	ISCIARRA	Ismael	FI_ACCOUNT	Sciarra	515.124.4369	
23	JMURMAN	Jose Manuel	FI_ACCOUNT	Urman	515.124.4469	
24	LPOPP	Luis	FI_ACCOUNT	Popp	515.124.4567	
25	DRAPHEAL	Den	PU_MAN	Raphaely	515.127.4561	
26	AKHOO	Alexander	PU_CLERK	Khoo	515.127.4562	

Over Column Limit Issues

To 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.

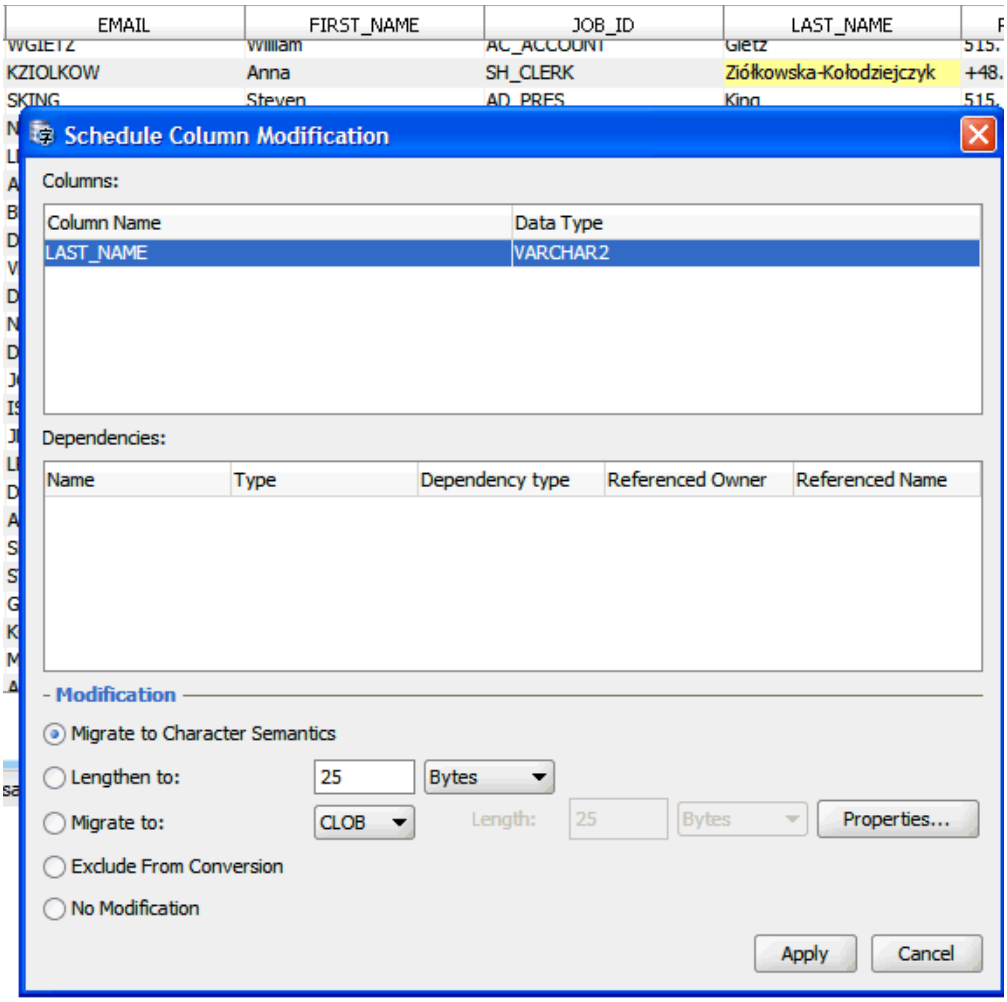
Lengthening a Column

Lengthening a column is the most efficient way to deal with "over column limit" issues, 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 6–12, "Schedule Column Modification: Lengthen the Column Size". 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 6–12 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 migration 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" on page 4-4 in the "Column Properties" section.

When lengthening a column, leave the drop-down list set to "Bytes". Click **Apply**.

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. 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 (with the exception of 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" on page 6-24.

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" on page 6-19 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**.

Shortening Character Values Manually

If the column data types cannot be altered, 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 do this:

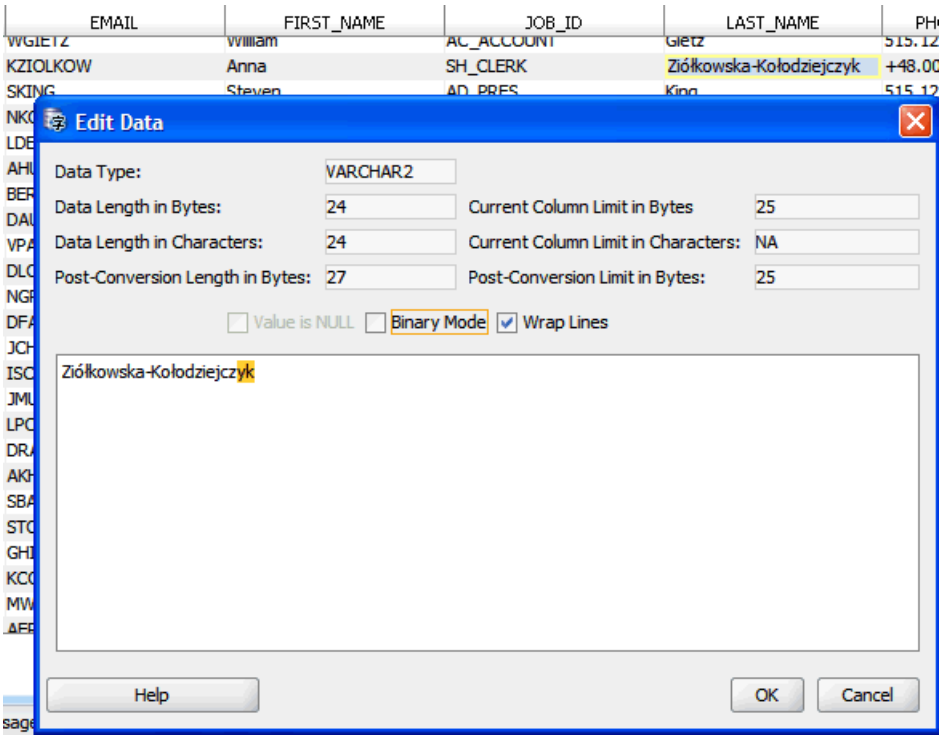
- Shorten the values manually.
- Let the DMU shorten the values during conversion.
- Replace some expanding characters with non-expanding equivalents.

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 6–13, "Edit Data Dialog Box".

Figure 6–13 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.

Truncating Column Values During Conversion

If data in a column is historical and no longer required for business purposes or the essential information is always contained at the very beginning of a cell value, 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 Converison of Exceptional Data.

See Chapter 3, "Viewing and Setting Object Properties in the DMU" for more information about setting column properties.

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" on page 6-21. 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.

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.

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" on page 6-19. 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. See *Oracle Database SQL Reference* for information about the available options. 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.

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" on page 6-19, 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" on page 6-19 for the steps required to perform any of the preceding available cleansing actions.

Refreshing Scan Results

When you define a column cleansing action in the Schedule Column Modification or the Modify Column dialog box, or you edit a value in the Edit Data dialog box, the DMU invalidates the scan results for the column, unless it can otherwise 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" on page 4-4) or in the Current Data column of scan results displayed on the Scanning subtab of the Column Properties tab.

Revoking Scheduled Cleansing Actions

A scheduled cleansing action for a column can be revoked at any time before the database is converted.

To revoke a scheduled action, open the Schedule Column Modification dialog box for the affected column, select **No Modification** and then click **Apply**. The scan results for the column will be invalidated and you will have to rescan the column.

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.

Cleansing Scenario 3: Cleansing Invalid Representation Issues

If there are invalid representation issues with data in your database, the Scan Report resembles what is shown in "Overview of the Database Scan Report" on page 4-15, 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" on page 4-4 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.

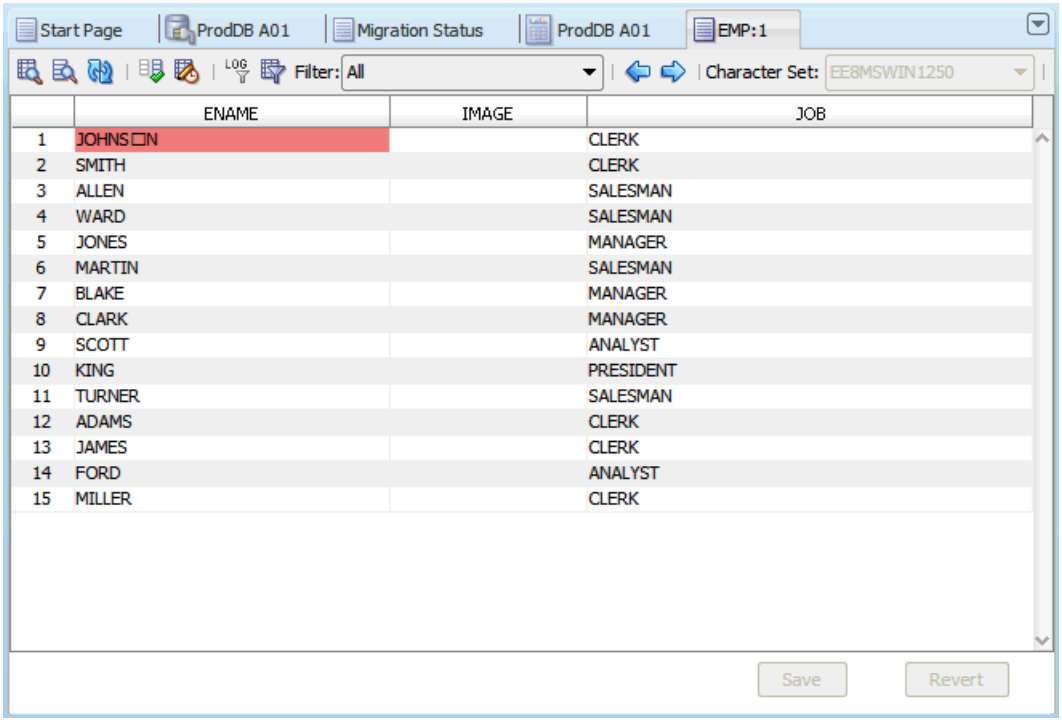
See "Invalid Binary Storage Representation of Data" on page 1-10 for a description of the pass-through configuration.

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 6–14, "Cleansing Editor Tab: Invalid Binary Representation Issues". 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 "Filtering Data" on page 6-5 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.

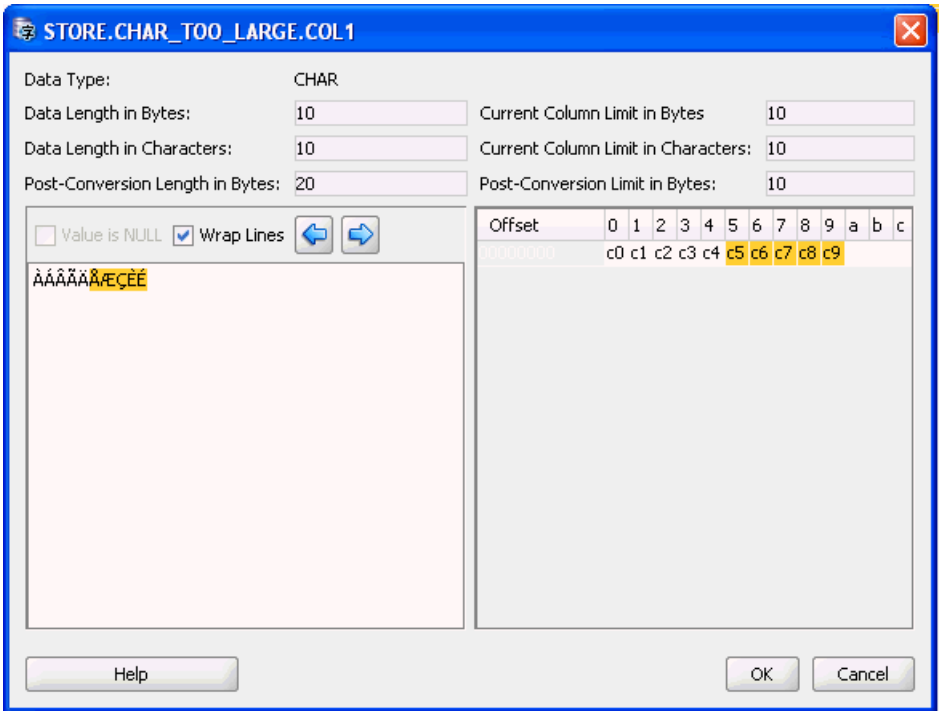
Figure 6–14 Cleansing Editor Tab: Invalid Binary Representation Issues



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 6–13, "Edit Data Dialog Box".

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 6–15, "Edit Data Dialog Box: Viewing Binary Data".

Figure 6–15 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" on page 6-24 and "Revoking Scheduled Cleansing Actions" on page 6-25 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" on page 4-21.

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" on page 6-19.
3. In the opened dialog box (see Figure 6–12, "Schedule Column Modification: Lengthen the Column Size"), 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**.

Cleansing Incorrect Character Set Declaration

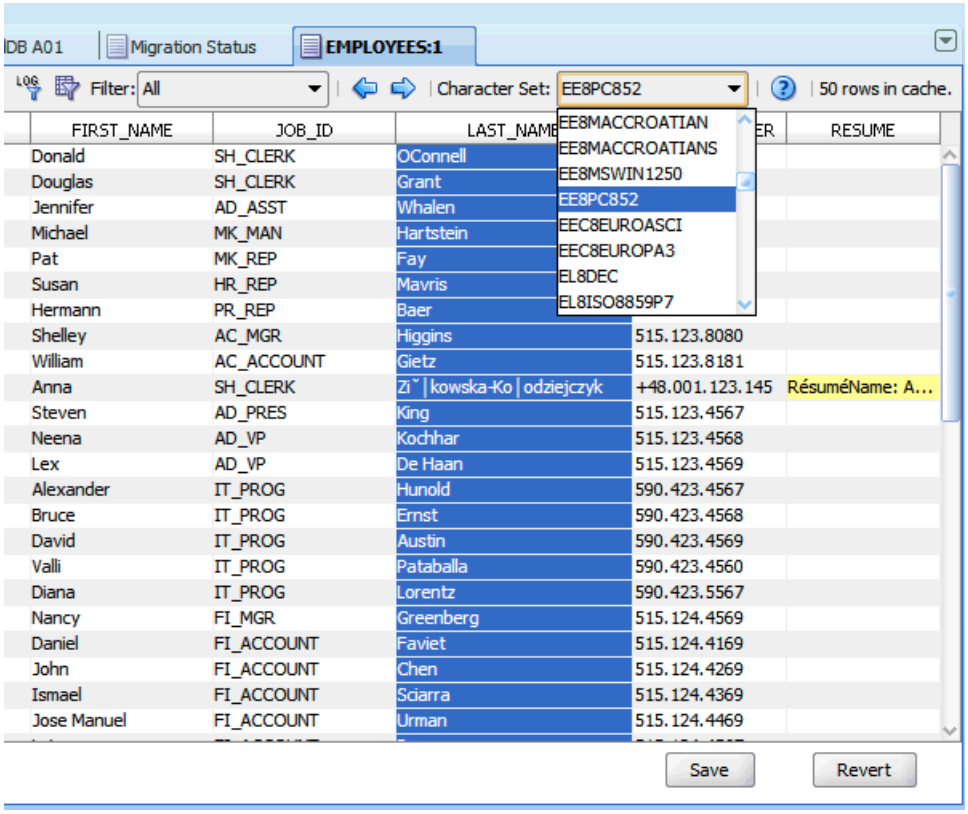
If the binary view in the Edit Data dialog box does not reveal any byte patterns characteristic for binary data, the edited value is most probably character data in an incorrectly used character set. If in doubt, consult application developers or application administrators and identify the source and 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 6–16, "Assumed Column Character Set". To save time, apply the knowledge gathered before starting the migration, as recommended in "Review Your Preparations for Migration" on page 2-4, 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 6–16 Assumed Column Character Set



Cleansing Corrupted Character Values

If only a very small number of values in a column is not legible in the assumed character set of the column and the characters illegible in one value are legible in another, 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" on page 6-21.

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 inline conversion.

character set migration

The process of moving from one database character set to another. In this guide, we discuss moving to Unicode.

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.

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.

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 A to B conversion.

initialization

The process of installing required patches and supporting packages, as well as creating a tablespace for the repository, a database connection, and then installing the migration 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.

Migration 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.

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.

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

F

H

I

K

M

O

P

R

S

T

U

V