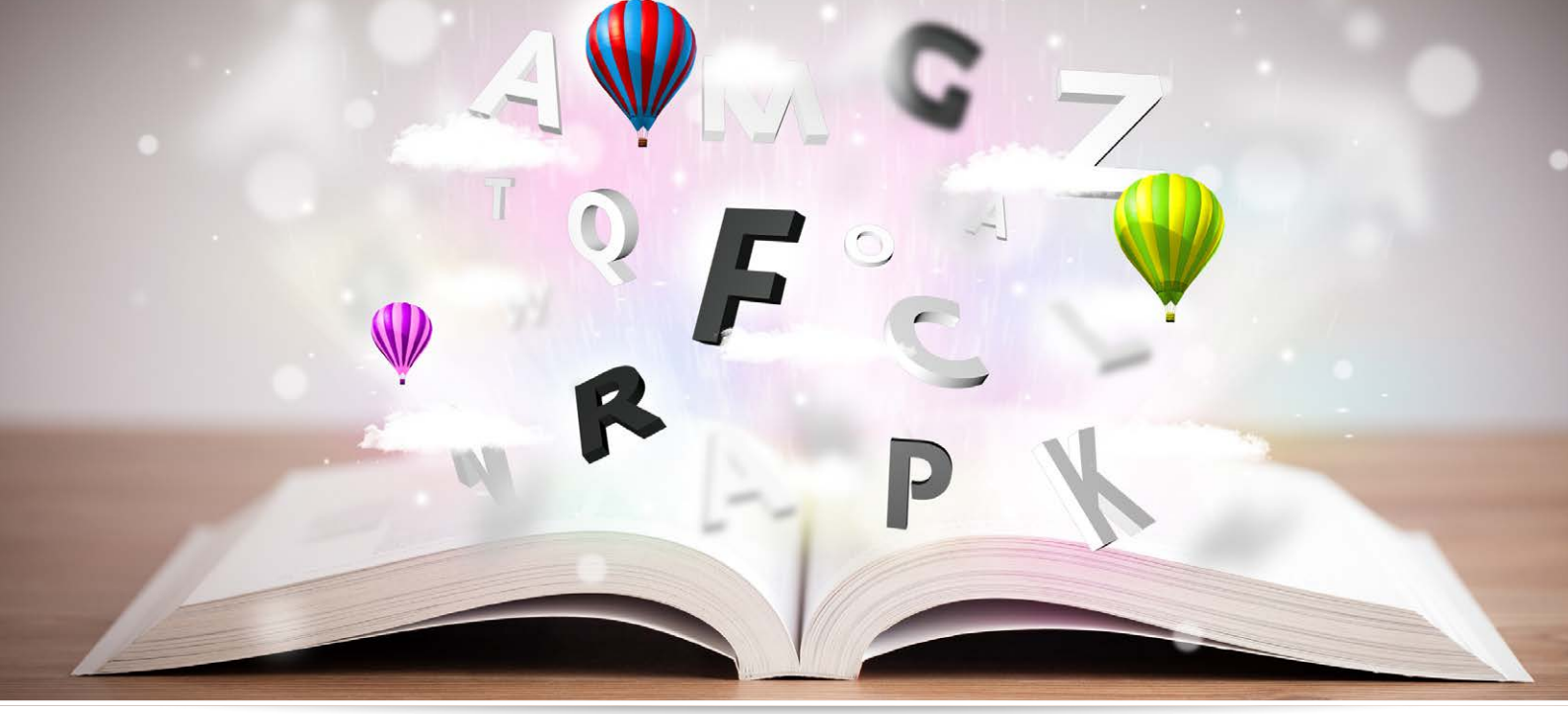

Character Sets – die Welt ist nicht genug

Martin Hoermann, Ordix AG

Die Auswahl des Character Sets beim Anlegen einer Datenbank entscheidet über die zur Verfügung stehenden Zeichen der Datenbank. Die richtige Verwendung und die Änderung des Zeichensatzes, etwa auf einen Unicode-Zeichensatz, stellen den Administrator und Anwendungsentwickler vor verschiedene Herausforderungen. Sollten Anwender im Geheimdienst arbeiten, müssen natürlich Nachrichten aus allen Sprachen dieser Welt gespeichert werden können. Der Artikel beleuchtet die Grundlagen von Character Sets, deren Migration sowie die verschiedenen Ausprägungen der unter Oracle verfügbaren Unicode-Zeichensätze.

Der Character Set einer Datenbank legt prinzipiell die zur Verfügung stehenden Zeichen einer Oracle Datenbank-Anwendung fest. In Westeuropa waren früher die Datenbank-Zeichensätze „WE8ISO8859P1", „WE8ISO8859P15" und „WE8MSWIN1252" sehr weit verbreitet. Diese Zeichensätze haben höchstens 256 Zeichen, sodass jedes Zeichen mit einem Byte kodiert werden kann *(siehe Abbildung 1)*.

Diese sogenannten Single-Byte-Zeichensätze haben den Nachteil, dass nicht definierte Zeichen – etwa die Buchstaben des griechischen Alphabets – damit nicht definiert gespeichert werden können. Es ist zwar prinzipiell möglich, die Bytes anders als definiert zu interpretieren, jedoch führt dies zu einer sehr unübersichtlichen Datenspeicherung. Davon ist dringend abzuraten.

Um das Problem des eingeschränkten Zeichensatzes zu beheben, wurde mit Unicode ein sehr mächtiger Standard definiert, der die Zeichen der meisten lebenden und vieler ausgestorbener Sprachen definiert. Die am meisten verwendeten Kodierungen der Datenbank sind „UTF8" (veraltet), „AL32UTF8" und „AL16UTF16". Mit dem Unicode-Standard 6.1 stehen 110.181 Zeichen in der Version 12*c* zur Verfügung. Die Unicode-Zeichensätze sind Multibyte-Zeichensätze, die je nach Kodierung und Zeichen zwischen einem und vier Byte Speicherplatz benötigen.

Vom Client zur Datenbank und zurück

Dieses Kapitel beschreibt den Weg eines Zeichens von der Tastatur bis hin zur Datenbank und von der Datenbank wieder zurück zum Bildschirm des Anwenders: Wenn eine Datei auf einem Client erzeugt wird, so hat diese Datei eine bestimmte Kodierung (engl. „Encoding") oder auch einen bestimmten Zeichensatz. Idealerweise sollte die „NLS_LANG"-Variable genau diesen Zeichensatz benennen, beispielsweise „NLS_LANG=. WE8MSWIN1252", wenn es um eine auf Windows mit deutscher Spracheinstellung und mit einem Standard-Editor erzeugte Datei geht *(siehe Abbildung 2)*.

Unterscheidet sich der Zeichensatz in der Definition von „NLS_LANG" und dem Datenbank-Zeichensatz, so wird über einen festgelegten Algorithmus eine Konvertierung durchgeführt. Existiert dasselbe Zeichen in beiden Zeichensätzen, wird es entsprechend auf die Kodierung des Ziels – also der Datenbank – konvertiert. Existiert das Zeichen nicht, gibt es jedoch ein äquivalentes Zeichen, so wird das Zeichen ersetzt. So könnte beispielsweise aus einem „ä" ein „a" werden.

Gibt es weder das genaue noch ein äquivalentes Zeichen, so wird ein Ersatzzeichen („Replacement Character") verwendet. Äqui-

valente Zeichen und Ersatzzeichen können nicht wieder zurückgewandelt werden. Idealerweise sollten die Zeichensätze zwischen Client und Datenbank also entweder identisch sein oder die Datenbank sollte eine Obermenge („Superset“) von Zeichen zur Verfügung stellen. Dies ist dann in der Regel ein Unicode-Zeichensatz. Sind der Client-Zeichensatz und der Datenbank-Zeichensatz identisch, gehen weder Zeichen verloren noch werden Zeichen ersetzt. Dies ist der Idealzustand einer jeden Datenbank-Anwendung.

Wird nun ein Zeichen aus der Datenbank zurückgelesen, passiert prinzipiell genau das zuvor Beschriebene, nur in umgekehrter Richtung. Unterscheiden sich also „NLS_LANG“ des Client und der Zeichensatz der Datenbank, können Zeichen entweder in ein äquivalentes Zeichen oder ein Ersatzzeichen konvertiert werden. Alle Daten, die verlustfrei vom Client in die Datenbank gelangt sind, kommen bei gleicher Konfiguration auch verlustfrei wieder zurück.

Ob die Zeichen allerdings korrekt dargestellt werden können, hängt vom Zeichensatz des Frontend und dem zur Verfügung stehenden Font zur Darstellung ab. In einer DOS-Box („Codepage 850“) gelingt es meist nicht, trotz korrekter Einstellung die deutschen Umlaute darzustellen. Schreibt man hingegen aus einer DOS-Box eine Spool-Datei und öffnet diese wiederum mit einem Standard-Editor („Codepage 1252“), so sind die Zeichen dort sichtbar.

Code	…0	…1	…2	…3	…4	…5	…6	…7	…8	…9	…A	…B	…C	…D	…E	…F
0…	nicht belegt															
1…																
2…	SP	!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
3…	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4…	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5…	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6…	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7…	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	
8…	nicht belegt															
9…																
A…	NBSP	¡	¢	£	¤	¥	¦	§	¨	©	ª	«	¬	SHY	®	¯
B…	°	±	²	³	´	µ	¶	·	¸	¹	º	»	¼	½	¾	¿
C…	À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï
D…	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß
E…	à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï
F…	ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ

Abbildung 1: ISO-8859-1, Quelle: http://de.wikipedia.org/wiki/ISO_8859-1

Invalid Data & Conversion Errors

Bei der oben beschriebenen Verbindung kann es zu zwei Arten von Problemen kommen, die Oracle als „Invalid Data“ beziehungsweise „Pass Through“-Konfiguration und „Conversion Errors“ bezeichnet. Beide Phänomene wurden bereits zuvor erklärt und werden nun an einem Beispiel veranschaulicht.

Ist die Kodierung einer Datei beispielsweise in „WE8ISO8859P7“ und enthält Zeichen des griechischen Alphabets und sind die „NLS_LANG“-Variable sowie der Datenbank-Zeichensatz auf WE8ISO8859P1 eingestellt, werden die Zeichen an die entsprechenden Positionen „0xA0..0xFF“ geschrieben. Werden die Daten mit derselben Konfiguration ausgelesen und als „P7“-kodiert interpretiert, sind die griechischen Zeichen wieder sichtbar.

Ohne dieses Wissen sind die Daten jedoch bedeutungslos, da es sich im „P1“-Zeichensatz um Sonderzeichen verschiedener Sprachen handelt. Wird die Datenbank nun beispielsweise in einen anderen Zeichensatz konvertiert, werden natürlich semantisch die „P1“-Daten konvertiert, was ein Auslesen praktisch unmöglich macht. Oracle spricht hier von „Invalid Data“. *Abbildung 3* zeigt, wie eine solche „Pass Through“-Konfiguration mittels DMU (siehe unten) behoben werden kann.

Ist die Kodierung einer Datei beispielsweise „WE8ISO8859P7“ und enthält Zeichen des griechischen Alphabets und sind die „NLS_LANG“-Variable auf „P7“ und der Datenbank-Zeichensatz auf „P1“ eingestellt, so konvertiert die Datenbank die Zeichen. Dabei werden bis auf das kleine Beta alle Zeichen durch den Replacement-Character des „P1“ – das umgekehrte Fragezeichen – ersetzt. Das kleine Beta wird zum deutschen „ß“. Die ursprünglichen Daten sind damit nicht mehr reproduzierbar. Oracle spricht von „Conversion Errors“.

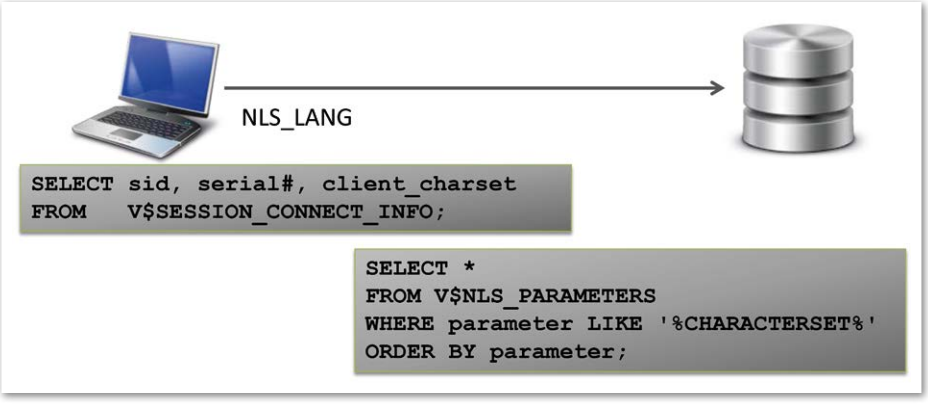

Abbildung 2: Informationen zum Character Set des Client und der Datenbank

Encoding I

Bei Single-Byte-Datensätzen ist die Kodierung relativ einfach. Jedem Buchstaben wird ein Wert zwischen „0x00“ und „0xFF“ zugewiesen. Die Zuweisung erfolgt über Character-Set-Tabellen, die im Internet, etwa bei Wikipedia, einfach zu recherchieren sind.

Bei Unicode-Zeichensätzen gestaltet sich die Kodierung etwas schwieriger. Jedes Unicode-Zeichen erhält eine Nummer.

Diese Nummer wird in der Regel Hex-kodiert mit führenden „U+“ geschrieben. So ist ein „Ä“ beispielsweise ein „U+00C4“. Im Internet stehen umfangreiche Unicode-Tabellen zur Verfügung.

Jetzt gibt es für verschiedene Anwendungszwecke unterschiedliche Kodierungen. Für den westeuropäischen Raum eignet sich die Kodierung „AL32UTF8“. Dabei werden die 128 ASCII-Zeichen identisch wie in der ASCII-Tabelle kodiert und kommen daher mit 7 Bit plus einem führenden „0“-Bit aus. Alle anderen Zeichen werden mit zwei bis vier Byte kodiert. Hilfreich kann der Algorithmus zum Auslesen sein, um etwa festzustellen, ob ein „Ä“ tatsächlich ein „Ä“ ist.

Mithilfe der Dump-Funktion auf ein einzelnes Zeichen ergibt sich beispielsweise „SELECT dump(zuklaerendeszeichen) FROM tabelle“ und „Typ=1 Len=2 195, 132“. Im ersten Schritt bietet es sich an, die Zahlen in Binärschreibweise als „11000011 10000100“ aufzustellen. Die führende „110“ des ersten Byte beschreibt, dass es sich um einen Zwei-Byte-Zeichen handelt. Die führende „10“ des zweiten Byte besagt, dass es ein Folgebyte ist. Alle anderen Bits werden nun zusammengefasst, in Hex-Code umgewandelt und aus einer Unicode-Tabelle ausgelesen: „00011000100 = 19610 = U+00C4 = Ä“. Bei dem zu klärenden Zeichen handelt es sich also tatsächlich um ein „Ä“.

Encoding II

Wie bereits erläutert, sind vielfältige Komponenten auf dem Weg von der Tatstatur bis zur Datenbank beteiligt. Der Datenbank-Administrator legt beim Anlegen der Oracle-Datenbank in der Regel nach Vorgabe des Applikationsverantwortlichen den Zeichensatz der Datenbank fest. Dieser ist im Nachhinein nur mit einigem Aufwand zu ändern.

Auf Seite des Client sind die Oracle-NLS-Einstellungen und die Kodierung der Anwendung relevant. Im Falle einer Datei kann diese von verschiedenen Werkzeugen gelesen und manipuliert werden. SQL*Developer, TOAD und Ultra-Compare verfügen zwar über Einstellungen für die Default-Kodierung, interpretieren aber häufig zur Laufzeit anhand des Daten-Inhalts die Kodierung der Datei um. Gelangt zum Beispiel ein Schmierzeichen in eine Datei, so wird von vielen Tools eine Unicode-Kodierung angenommen. Dies kann fatale Auswirkungen bei der Arbeit mit den Dateien haben. Für die Arbeit mit einer Anwendung gilt prinzipiell dasselbe.

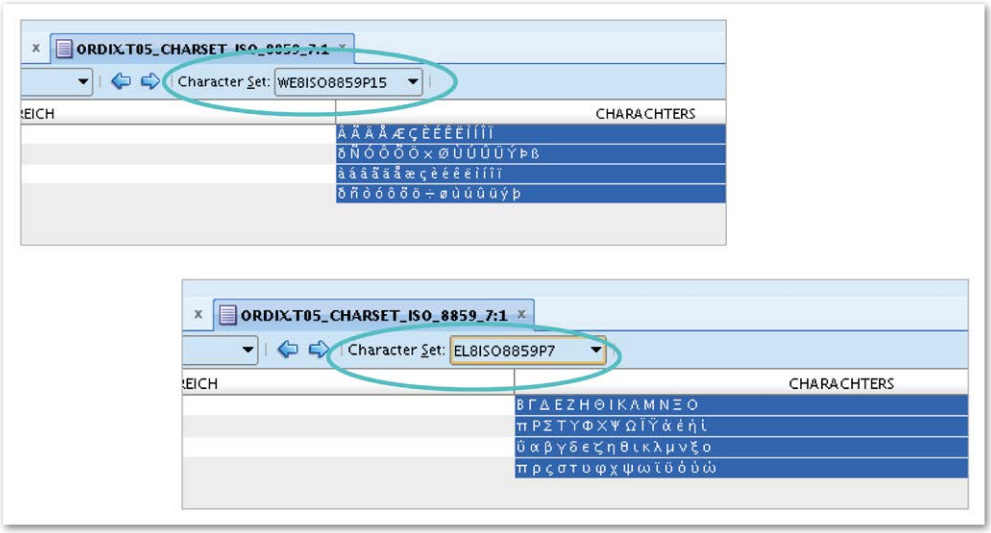

Abbildung 3: Behebung einer „Pass Through“-Konfiguration mittels DMU

Unicode in der Datenbank

Die Datenbank verfügt über zwei Zeichensätze („v$nls_parameters“). Der Datenbank-Zeichensatz („NLS_CHARACTERSET“) legt die Kodierung für die Spalten mit den Datentypen „VARCHAR2“, „CHAR“ und „CLOB“ fest. Der NLS-Datenbank-Zeichensatz („NLS_NCHAR_CHARACTERSET“) bestimmt die Kodierung für die Spalten mit den Datentypen „NVARCHAR2“, „NCHAR“ und „NCLOB“.

Bis zu der Oracle-Datenbank-Version 8 lautete der Name des „UTF-8“-Zeichensatzes „UTF8“, ab der Version 9 heißt er „AL32UTF8“. Der Präfix „AL2 steht dabei für „All Languages“. Abhängig von der Oracle-Version wird mit diesem Zeichensatz eine jeweils aktuelle Unicode-Spezifikation unterstützt, beispielsweise in Version 11 die Spezifikation 5.0 und in der Version 12*c* R1 die Spezifikation 6.1 *(siehe Abbildung 4)*.

In früheren Versionen war es noch möglich, sowohl für den Datenbank-Zeichensatz als auch für den NLS-Datenbank-Zeichensatz einheitlich „UTF8“ zu verwenden. Bei Migrationen stehen beim Datenbank-Zeichensatz „AL32UTF8“ für den NLS-Datenbank-Zeichensatz ausschließlich die Zeichensätze „AL16UTF16“ und „UTF8“ zur Verfügung. Der Zeichensatz „AL16UTF16“ ist unter anderem optimiert auf chinesische, koreanische und japanische Zeichen, von denen die meisten nur zwei Byte benötigen. In „AL32UTF8“ benötigen diese Zeichen in der Regel drei Byte.

Bei der Migration des Zeichensatzes kann es aufgrund der unterschiedlichen Kodierung natürlich zu einem Datenverlust kommen, da die Kodierungen für ein und dasselbe Zeichen unterschiedlich viele Bytes benötigen. Insbesondere die Einstellung von „NLS_LENGTH_SEMANTIC“ auf Bytes statt Character verursacht hier potenziell Probleme.

Aber auch mit der Einstellung auf Character kann das Problem entstehen, wenn die maximale Größe der Datentypen „CHAR“ und „VARCHAR2“ bei der Konvertierung 4.000 Bytes überschreitet. Unter Oracle 12*c* lässt sich diese Grenze auf 32.767 Byte erweitern. Hierzu ist der Initialisierungsparameter „MAX_STRING_SIZE“ auf „EXTENDED“ zu setzen. Diese Einstellung ist unumkehrbar.

Abbildung 4: Das wunderschöne Zeichen „U+2A6A5“ besteht aus 64 Grundstrichen und stellt vier Mal das Wort „Drachen“ dar

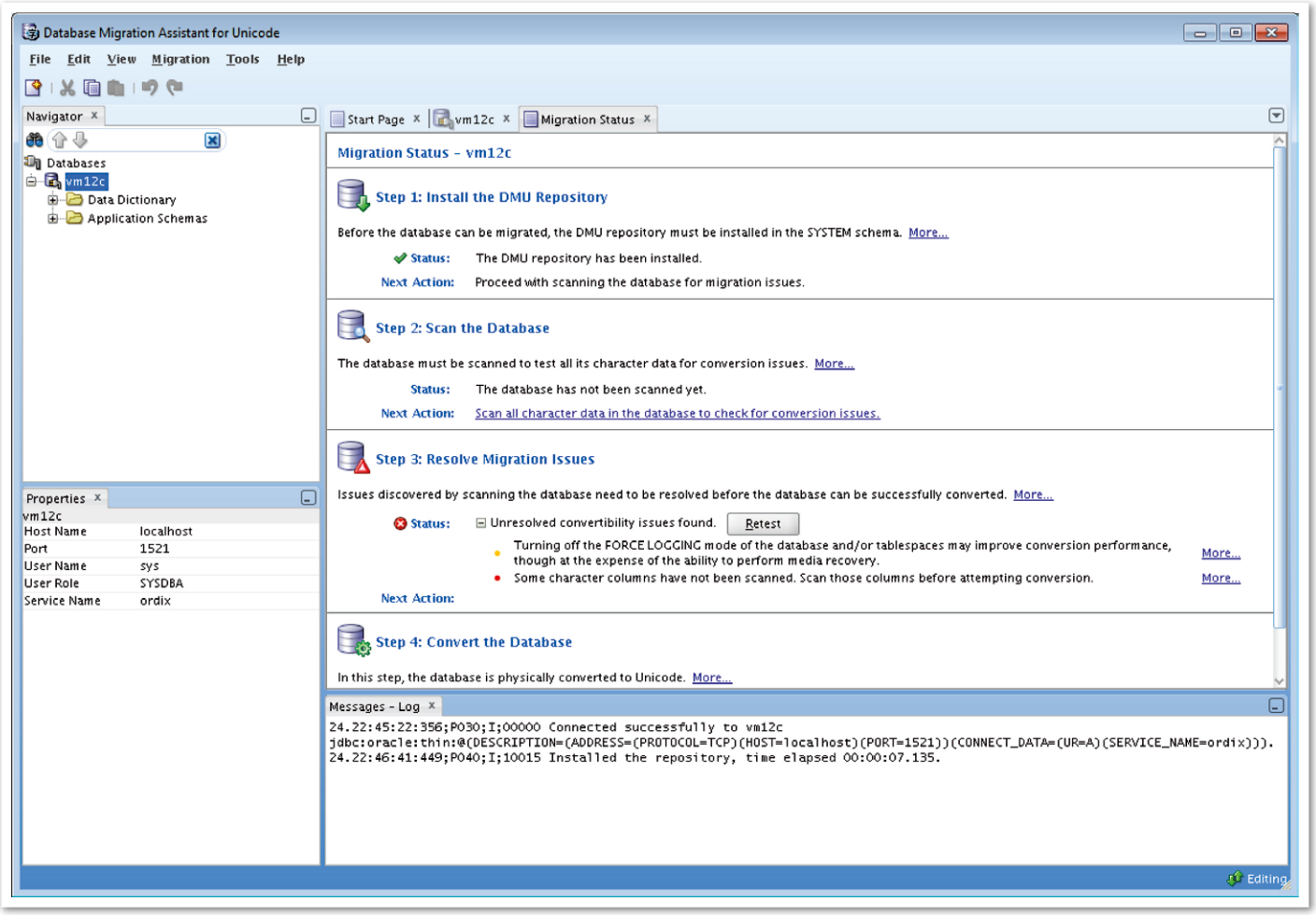

Abbildung 5: Startmaske des DMU

Character-Set-Migration

Oracle bietet verschiedene Möglichkeiten, den Zeichensatz einer Datenbank zu migrieren oder die Migration zu prüfen: „Export/Import“, „CSSCAN“ und „DMU“. Das Database Migration Utility (DMU) steht seit der Version 12.1 zur Verfügung und soll die Arbeit bei der Migration des Zeichensatzes vereinfachen *(siehe Abbildung 5)*.

Bei der Migration auf einen anderen Zeichensatz besteht aufgrund der Kodierung immer die Möglichkeit des Datenverlusts („Data Truncation“), weil Character-Felder eine für die neue Kodierung nicht ausreichende Länge haben (siehe oben). Zudem kann es zu Konvertierungsfehlern kommen, wenn ein Zeichen des ursprünglichen Zeichensatzes im neuen Zeichensatz nicht zur Verfügung steht („Replacement Errors“). In diesem Fall werden die Zeichen durch ein sogenanntes „Replacement Character“ ersetzt (siehe oben). Sowohl „DMU“ als auch der Vorgänger „CLSCAN“ prüfen die Daten vor der Migration auf potenzielle Fehler.

Sortierung

Der Datenbank-Zeichensatz legt die zur Verfügung stehenden Zeichen fest. Dies ist jedoch nur die Basis der Globalisierung einer Datenbank-Anwendung. Im nächsten Schritt muss für die Anwendung eine Sortierung festgelegt werden. Dies geschieht in der Regel über den Parameter „NLS_SORT“, der wiederum abhängig vom Parameter „NLS_LANGUAGE“ ist. Für die gängigen Sprachen stehen die dort typisch verwendeten Sortiermöglichkeiten zur Verfügung. Diese sind für den deutschen Sprachraum beispielsweise „GERMAN“, „XGERMAN“, „GERMAN_DIN“ und „XGERMAN_DIN“. Ab der Version 12 stellt Oracle eine Implementierung des Unicode Collation Algorithm (UCA) zur Verfügung. Über dieses Verfahren können auch mehrsprachige Sortierungen eingestellt und sogar selbst implementiert werden.

Fazit

Dieser Artikel zeigt die zahlreichen Facetten des Datenbank-Zeichensatzes. Mit einem Unicode-Zeichensatz stehen zwar alle gängigen Zeichen dieser Welt zur Verfügung, nichtsdestotrotz kann es bei der Anwendung zahlreiche Hürden geben. Vielleicht kann ja die NSA bei der Konvertierung kyrillischer Geheimbotschaften helfen.

Martin Hoermann
mh@ordix.de