Migrating Database Character Sets to Unicode

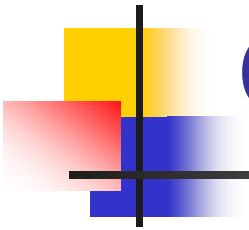
Yan Li

Yanli_us@yahoo.com

3 P + 3 H

- Be Positive
- Be Prepared
- Be Patient

- Be H ?
- Be H ?
- Be H ?



OUTLINE

- What is Unicode?
- How to choose Unicode character set?
- How to install and use CSSCAN?
- What are the issues, options and resolutions?
- What are the migration methods?

Assumptions

- Database Version: 10g and up
- Source CharacterSet: WE8ISO8859P1
- Target CharacterSet: AL32UTF8
- CharacterSet Scope: NLS_CHARACTERSET

What is Unicode?

- Universal encoded character set
- Store information in ANY language
- Unique code point for each character, regardless of platform, program, or language
- Version 5.1.0 contains over 100,000 characters
- Essential to support global business

Major Unicode Formats

- UTF-8
  8-bit variable-width 1-4 bytes/character
- UCS-2
  16-bit fixed-width 2 bytes/character
- UTF-16 (extension of UCS-2)
  16-bit fixed–width 2 or 4 bytes/character
- UTF-4, UTF-7, UTF-32

Standard Unicode vs. Oracle

Standard Unicode	Oracle Unicode
UTF - 8	UTF8 AL32UTF8
UTF - 16	AL16UTF16

Oracle - UTF8 vs. AL32UTF8

UTF8

- 1-3 bytes per character
- Support Unicode standard to Version 3.0 only

AL32UTF8

- 1-4 bytes per character
- Continue supporting future Unicode standard
- Best choice of a Unicode character set
- Not recognized by pre-9i database

Note: Steps of migrating to UTF8 or AL32UTF8 are the same

Oracle - AL16UTF16

- 2 or 4 bytes per character
- Not used for normal database character set
- Used for national character set on NCHAR, NVARCHAR2 and NCLOB columns
- Continue supporting future Unicode standard

?

Install CSSCAN

- **What is CSSCAN ? (Character Set SCANer)**
  - Mandatory for character set migration
  - CSALTER based on CSSCAN result
- **Install CSSCAN**
  - Run $ORACLE_HOME/rdbms/admin/csminst.sql
  - Modify the script not to use SYSTEM tablespace
- **Schema owner and objects**
  - Schema owner: CSMIG
  - Tables: CSM$...
  - Views: CSMV$...

Run CSSCAN

- **Scan levels**

  full database, owner, table

- **Get help**

  csscan help=y

- **Command line**

  csscan full=Y fromchar=WE8ISO8859P1 tochar=AL32UTF8

- **Use parfile**

  csscan parfile=…

CSSCAN OUTPUT (1)

CSSCAN output files:

- scan.txt - summary of exceptions
- scan.err - detailed rows with ROWID
- scan.out - log of the scan process

CSSCAN OUTPUT (2)

SAMPLE scan.txt file:

[Database Size]

Tablespace	Used	Free	Total	Expansion
SYSTEM	12,586.25M	5,413.75M	18,000.00M	343.00K
TSABCD	9,807.38M	2,192.63M	12,000.00M	13.38M
TSXXX	267.44M	1,732.56M	2,000.00M	.00K
TSYYY	64.00K	1,999.94M	2,000.00M	.00K
TSZZZ	7,285.88M	714.13M	8,000.00M	.00K
TSINDEX	640.06M	1,359.94M	2,000.00M	.00K
TSUNDO1	2,141.13M	93,858.88M	96,000.00M	.00K
TSUNDO2	24,519.00M	35,481.00M	60,000.00M	.00K
… …				
Total	3,581,509.00M	586,521.98M	4,168,030.98M	586.55M

CSSCAN OUTPUT (3)

[Data Dictionary Conversion Summary]

Datatype	Changeless	Convertible	Truncation	Lossy
VARCHAR2	136,913,559	10	0	154
CHAR	7	0	0	0
LONG	6,614,799	0	0	0
CLOB	58	71	0	0
Total	143,528,423	81	0	154
Total in percentage	100%	0%	0%	0%

[Application Data Conversion Summary]

Datatype	Changeless	Convertible	Truncation	Lossy
VARCHAR2	189,317,244,480	113,000	14,326	7,455,330
CHAR	977,410,022	0	0	0
LONG	36,450,616	0	0	0
CLOB	585,086,356	100,702	0	523
Total	190,916,191,474	213,702	14,326	7,455,853
Total in percentage	100%	0%	0%	0%

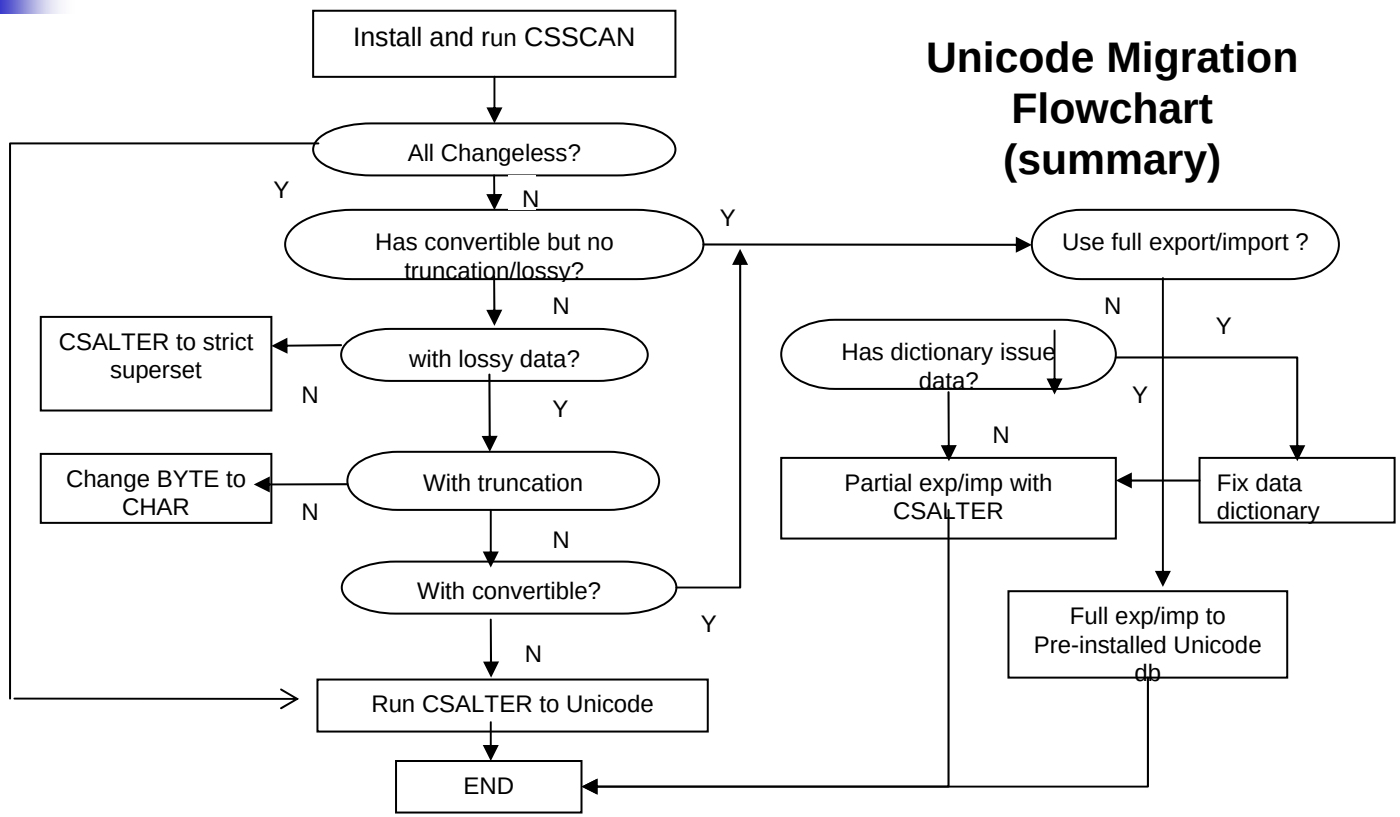
CSSCAN OUTPUT (4)

CSSCAN results in 4 types of data:

1. Changeless
2. Lossy data
3. Truncation
4. Convertible

Migration tasks depend on the scan output.

MIGRATION FLOWCHART



MIGRATION TASKS

- If all “changeless”
  - run CSALTER to Unicode. No more steps
- If have “lossy” data
  - convert to its “strict” superset using CSALTER
- If have “truncation”
  - changing length semantics from BYTE to CHAR on database level or column level
- If have “convertible”
  - full export/import
  - or partial export/import with CSALTER

Note: “lossy” and “truncation” must be handled before “convertible”

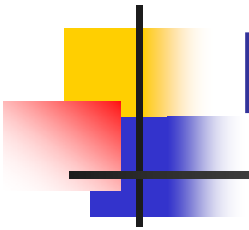
LOSSY DATA (1)

To find “lossy”, run CSSCAN with fromchar / tochar both to current character set

csscan full=Y fromchar=WE8ISO8859P1 tochar=WE8ISO8859P1

Sample “lossy” data in scan.err file:

```
ROWID       Exception Type     Size   Cell Data(first30 bytes)
--------    ---------- ----- ------------------------------
AACQtnAC3AAAp2XAAF lossy conversion     XT¬Q QVRLTQ
AACQtnAC3AAAqHsAAG lossy conversion     VO¬C KTLUOC
...
```



LOSSY DATA (2)

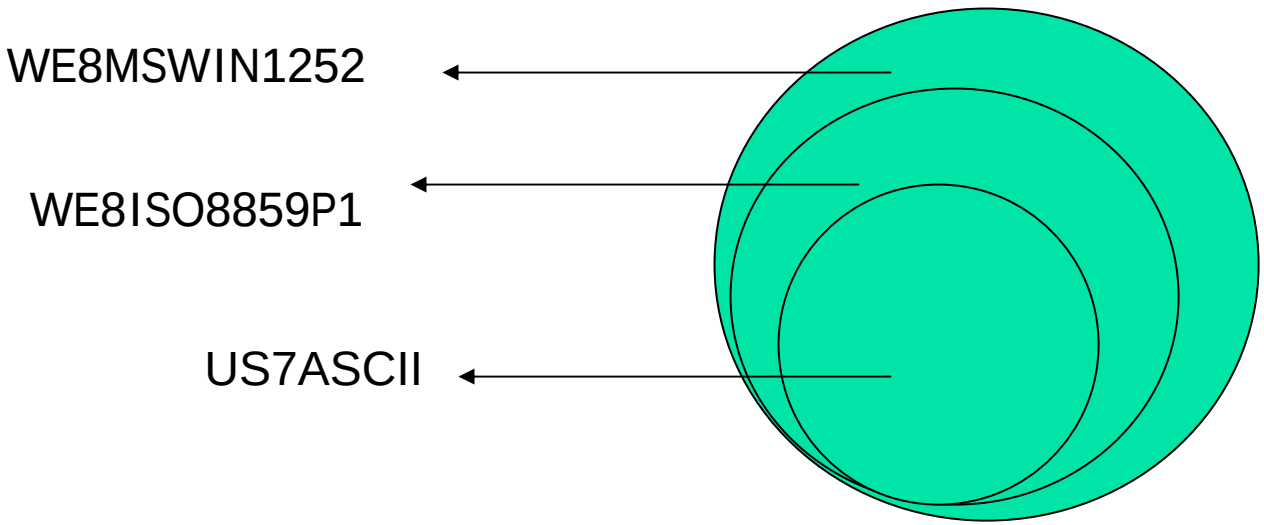
What is “lossy” conversion?

- Not a valid value for database character set

  Example: € can NOT be stored in WE8ISO8859P1
- Maybe caused by incorrect client NLS_LANG setting
- If no action taken, data will be “lost” in conversion
- Lossy character will be converted to “default replacement character”, usually “?” or “¿”
- Can be resolved by converting to its “strict” superset

LOSSY DATA (3)

“strict” superset - if and only if each and every character in the source character set is available in the target character set, with the same corresponding character value.



LOSSY DATA (4)

Run CSSCAN with fromchar / tochar both to “strict” superset

```
csscan full=Y fromchar=WE8MSWIN1252 tochar=WE8MSWIN1252
```

Must have 3 lines in scan.txt in order to run CSALTER

All character type data in the data dictionary remain the same in the new character set
All character type application data remain the same in the new character set
The data dictionary can be safely migrated using the CSALTER script

LOSSY DATA (5)

Convert WE8ISO8859P1 to WE8MSWIN1252 using CSALTER

- Shutdown listener and connected applications
- Shutdown db and startup restrict
- Run $ORACLE_HOME/rdbms/admin/csalter.plb
- Query nls_database_parameters to verify new character set

?

TRUNCATION (1)

CSSCAN from current character set to Unicode:

csscan full=Y fromchar=WE8MSWIN1252 tochar=AL32UTF8

Output: no “lossy” but may have the following:

“truncation” and “convertible”

What is “truncation”?

data resulting from conversion does not fit within the column’s maximum length

TRUNCATION (2)

- Column length typically expressed in bytes

  CHAR(2) = CHAR(2 BYTE)

  VARCHAR2(10) = VARCHAR2(10 BYTE)
- Single byte encoding

  1 byte = 1 char
- Multi-byte encoding

  1 char = 1-4 bytes (AL32UTF8)

TRUNCATION (3)

Example: Euro symbol: €

WE8MSWIN1252	1 byte
AL32UTF8	3 byte

e.g.

Table MONEY, Column SYMBOL CHAR(1):

'$' –	ok	(still 1 byte)
'€' –	"truncation"	(3 bytes now)

TRUNCATION (4)

How to handle “truncation”?

1. Enlarge column using more bytes

```
alter table MONEY modify (SYMBOL CHAR(3));
```

2. Change length semantics to CHAR

- at database level

```
alter system SET nls_length_semantics=CHAR;
```

- at session level

```
alter session SET nls_length_semantics=CHAR;
```

- change column from BYTE to CHAR

```
alter table MONEY modify (SYMBOL CHAR(1 CHAR));
```

TRUNCATION (5)

What if BYTE to CHAR exceeds maximum bytes?

Example:

User : ABC
Table : RECORDS
Column : DESC
Type : VARCHAR2(4000)
Number of Exceptions: 1
Max Post Conversion Data Size: 4082

And the ROWID is ‘AACQtnAC3AAAp2XAAF’

TRUNCATION (6)

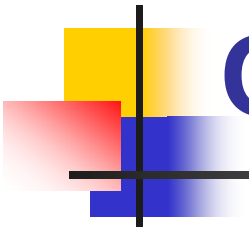
- **Shorten violated data**

```
UPDATE ABC.RECORDS
SET  DESC = 'Abcdefg….'  -- a value <= 4000 bytes
WHERE ROWID = 'AACQtnAC3AAAp2XAAF';
```

- **Change VARCHAR2 column to CLOB**

```
ALTER TABLE ABC.RECORDS ADD (tmp CLOB);
UPDATE ABC.RECORDS SET tmp=TO_CLOB(DESC);
COMMIT;
ALTER TABLE ABC.RECORDS DROP COLUMN DESC;
ALTER TABLE ABC.RECORDS RENAME COLUMN tmp to DESC;
```

?



CONVERTIBLE (1)

What is “convertible” ?

- Valid data
- Different code point from source to target

How to handle it?

1. Full export/import
2. Partial export/import with CSALTER

(Note: use exp/imp; expdp/impdp needs patch)

CONVERTIBLE (2)

Method 1: Full export / import
(Don't need to handle source data dictionary issues)

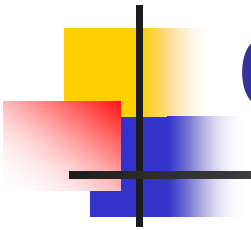
1. Pre create Unicode database (AL32UTF8)

From source database:

2. Set NLS_LANG to source character set and run full export
   NLS_LANG=AMERICAN_AMERICA.WE8MSWIN1252
   exp FULL=Y…

From Unicode database:

3. Set NLS_LANG to source character set and run full import
   NLS_LANG=AMERICAN_AMERICA.WE8MSWIN1252
   imp FULL=Y …



CONVERTIBLE (3)

Method 2: Partial export / import with CSALTER (Need to handle data dictionary issues)

To run CSALTER, data dictionary can have only:

- Changeless
- Convertible CLOB

All other issues must be addressed manually

How? - Follow Oracle documents and workarounds

CONVERTIBLE (4)

Method 2: Partial export / import with CSALTER (handle application data)

- 1. Export convertible tables with correct NLS_LANG
- 2. Truncate convertible tables
- 3. Run CSSCAN after truncating tables
  csscan full=y fromchar=WE8MSWIN1252 tochar=AL32UTF8
- 4. Convert to AL32UTF8 using CSALTER
  $ORACLE_HOME/rdbms/admin/csalter.plb
- 5. Import the convertible tables with correct NLS_LANG

Database and data are Unicode now!

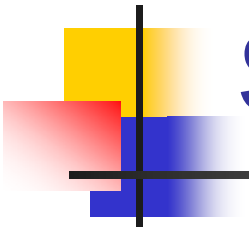
?

Unicode Client Tool

Use Unicode Client tool to validate

- **Oracle SQL Developer**
  Free fully supported GUI tool
  To download:
  http://www.oralce.com/technology/products/database/sql_developer/index.html

- **iSQL*plus**
  $ORACLE_HOME/bin/isqlplusctl start
  http://machine_name:5560/isqlplus/



SUMMARY

- Unicode and Oracle Character Sets
- Install and use CSSCAN
- Issues and resolutions

  “lossy” data, truncation, convertible
- Migration methods

  Full export/import

  Partial export/import with CSALTER

3 P + 3 H

- Be Positive
- Be Prepared
- Be Patient
- Be Happy
- Be Healthy
- Be Helpful

THANK YOU!

Yan Li

Yanli_us@yahoo.com