

IBM Software Group

DB2 for z/OS V8
Unicode: preparing today for tomorrow

Cécile Benhamou cecile_benhamou@fr.ibm.com

Based on a presentation by John J. Campbell and Florence Dubois

DB2 Information Management Software



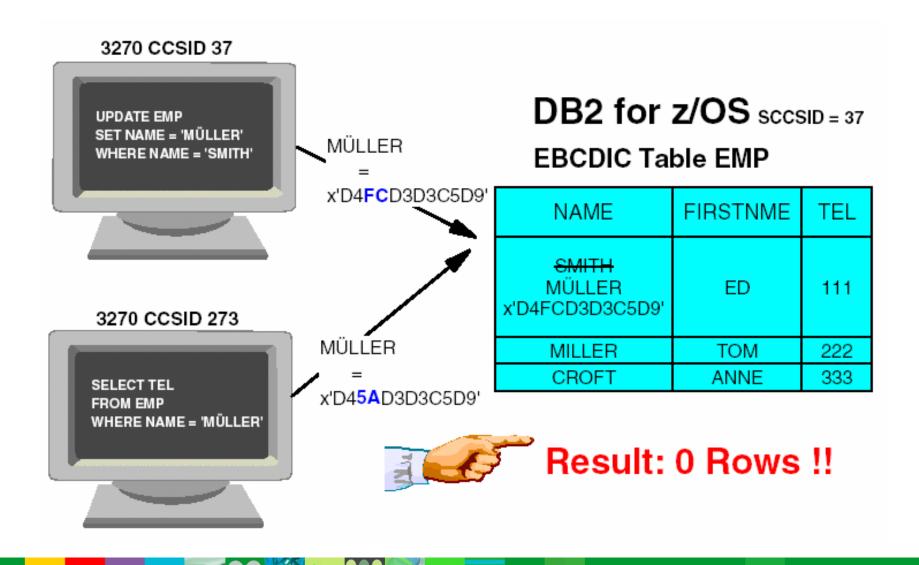


CCSID – Is it a problem?

- DB2 uses a set of three CCSIDs to describe data stored in DB2
 - EBCDIC, ASCII and Unicode
- DB2 supports specification of CCSIDs at DB2 subsystem level
 - **DSNHDFCP**
 - Once set they should not be changed!
- Local applications: General case, no data conversion
 - Terminal emulators should have a compatible code page
- Local applications: sometimes, conversion occurs
 - Dealing with ASCII/Unicode tables
 - Specified by application:
 - ✓ Application ENCODING bind option (V7)
 - ✓ CURRENT APPLICATION ENCODING SCHEME special register (V7)
- Remote applications: Automatic conversion for remote access when needed
 - Done by DRDA receiver

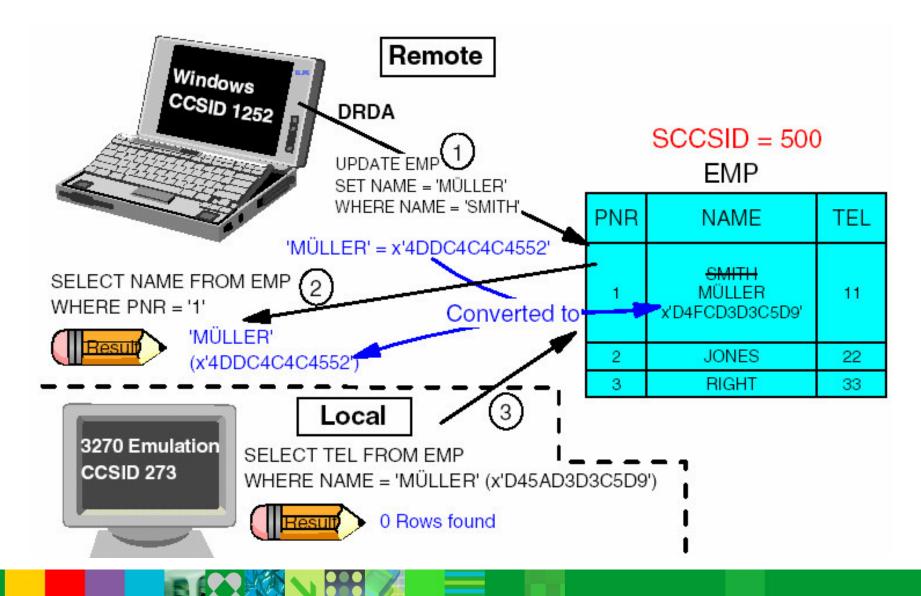


Storing Characters by Local Applications





Remote (DRDA) Applications





CCSID – Is it a problem? ...

- Potential for incorrect character interpretation (i.e., data corruption) if multiple CCSIDs within one encoding scheme
 - Terminal emulator using incompatible code page
 - Incorrect CCSID specification
 - Change of CCSID specification
- Must be fixed and residual corruption cleaned up ahead of V8
- V7 APAR PQ56697 introduces checks for non-zero valid CCSIDs and issues warning when trouble found
- V7 APAR PQ89018 will provide CCSID detection if DECP value does not match the emulator value





Conversion methods

- Native DB2 SYSIBM.SYSSTRINGS
- z/OS support for Unicode
 - Must configure Unicode Conversion Services for z/OS

```
COMMAND INPUT ===> /d uni,all
```

```
RESPONSE=SC63
 CUN3000I 02.37.19 UNI DISPLAY 752
  ENVIRONMENT: CREATED
                             04/17/2004 AT 10.08.19
              MODIFIED
                           04/17/2004 AT 10.08.20
               IMAGE CREATED 12/08/2003 AT 17.02.33
      SERVICE: CHARACTER
                              CASE
                                             NORMALIZATION COLLATION
      STORAGE: ACTIVE
                       424 PAGES
              LIMIT
                     51200 PAGES
     CASECONV: NORMAL
   NORMALIZE: DISABLED
      COLLATE: DISABLED
  CONVERSION: 00850-01047-ER
                                           01047-00850-ER
               00037-01200(13488)-ER
                                           01200 (13488) -00037-ER
               00037-01208-ER
                                           01208-00037-ER
               00437-01208-ER
                                           01208-00437-ER
               00037-00367-ER
                                           01252-00037-ER
               00037-01252-ER
                                           00367-00037-ER
               00500-01200(13488)-ER
                                           01200 (13488) -00500-ER
               01047-01200 (13488) -ER
                                           01200 (13488) -01047-ER
               01047-01208-ER
                                           01208-01047-ER
               01208-01200-ER
                                           01200-01208-ER
               01383-01200-ER
                                           01200-01383-ER
               00932-01200-ER
                                           01200-00932-ER
```

Brochure 'Support for Unicode: Using Conversion Services' (SA22-7649)





Prepare Unicode for DB2 V8

- Run DSNTIJP8 (V7) or DSNTIJPM (V8) ahead of migration to check that there are not multiple CCSIDs within one encoding
- V8 start up checks for:
 - Non-zero, valid CCSIDs
 - EBCDIC<->Unicode round trip conversion
- Must configure Unicode Conversion Services for z/OS
 - Unicode CCSIDs (367,1208,1200) <-> ASCII/EBCDIC CCSIDs
 - Client CCSIDs <-> Unicode CCSIDs (367,1208,1200)
 - Must also add to/from Unicode CCSIDs (367,1208,1200)
 - CCSID 37 for DBRMs provided with DB2 install
 - CCSID 500 for DRDA
 - CCSID 1047 for Unix System Services (USS)
 - Rebuild conversion image -> New image picked up by DB2 'on the fly'
- Configure for best conversion performance
 - z890 and z990 zSeries (hardware instructions instead of mcode/ucode) + z/OS R4



Unicode Conversion

- Use zparm UIFCIDS option to have IFCID trace data in Unicode UTF-8 format
 - Default = NO
 - Use YES to move conversion overhead off-line e.g., DB2PE
- Conversion of SQL statements and derived metadata to Unicode
 - Minor Conversion (fast)
 - Optimization for most common single byte CCSIDs (MIXED=NO) i.e., simple alphanumeric characters
 - ✓ Most common case ASCII/EBCDIC->UTF-8, can go other way
 - ✓ Internal 256 byte translation tables used by DB2 (TR/TRT)
 - Major Conversion (slower)
 - ✓ Offloaded to Unicode Conversion Services for z/OS
 - ✓ Used if EBCDIC/ASCII->UTF-8 cannot be performed inline
 - ✓ Always for ASCII/EBCDIC->UTF-16 (no optimization)
- Particular conversion not available -> SQLCODE -332



Unicode Catalog

- SQL parsed and metadata derived from SQL in Unicode UTF-8 format
- Most character columns converted to Unicode VARCHAR(128)
- Trailing blanks not stripped
- Special characters expand beyond single byte (e.g., #)
- Increase in space: CHAR -> VARCHAR (length bytes)
- Decrease in space: DB2 defined indexes converted to NOT PADDED
- Consider converting user defined indexes to NOT PADDED
- Compatibility (COMPAT) mode
 - SQL parsed in Unicode UTF-8 format
 - Metadata derived from SQL converted back to EBCDIC
- New Function Mode (NFM) mode
 - Catalog converted to Unicode UTF-8 format
 - Metadata derived from SQL no longer converted back to EBCDIC



Unicode Catalog ...

- If precompile application with NEWFUN(NO)
 - Disables use of new SQL function for static SQL
 - Go back through SQL Parser during BIND processing
 - ✓ To get Unicode UTF-8 format
 - ✓ Incur extra overhead
- After entry to NFM, default for NEWFUN changes to YES
 - DBRMs switch to Unicode UTF-8 format
 - Set NEWFUN to NO
 - ✓ Get DBRMs in EBCDIC format.
- At BIND time
 - Anything from DBRM in Unicode UTF-8 format goes into SYSSTMT and SYSPACKSTMT in Unicode UTF-8 format
 - Anything from DBRM in EBCDIC format goes into SYSSTMT and SYSPACKSTMT in **EBCDIC** format







Unicode Catalog ...

- ORDER: How to influence it?
 - SELECT NAME FROM SYSIBM.SYSTABLES WHERE NAME LIKE 'T%' ORDER BY NAME
 - ✓ V7 (EBCDIC): returns TA, TB, T1, T2
 - √ V8 (UNICODE): returns T1, T2, TA, TB
 - SELECT CAST(NAME AS CCSID EBCDIC) AS E_NAME FROM SYSIBM.SYSTABLES WHERE NAME LIKE 'T%' ORDER BY E NAME
 - ✓ returns TA, TB, T1, T2
 - ✓ Column NAME is converted into EBCDIC







Migration Highlights

- 18 Tablespaces converted to Unicode
- Some Tablespaces which must interface with MVS external names stay in **EBCDIC**
 - SYSCOPY
 - DBD01, SCT02, SYSLGRNX, SYSUTILX
- New tablespace called SYSEBCDC contains a single row/byte to support **SYSDUMMY**
- 2 Tables dropped
 - SYSLINKS
 - SYSPROCEDURES





Migration Highlights ...

- Run DSNTIJP8 asap under V7 to check for rogue conditions in V7 Catalog prior to V8
 - Mode (COMPAT, ENFM, NFM) is stored away in Catalog and Directory
- DSNTIJNE job (ENFM) can be restarted from the beginning without change
 - Objects reorganized in a specific order
- Up to 10% increase in disk space requirement for Cat/Dir
 - Some objects shrunk by 50% because they had never been reorganized



Some Examples if using Unicode in V8

- Select rows from ET1, UT1 where C1 = 'ABC'
 - Application encoding scheme: EBCDIC

Coded SQL statement:

SELECT ET1.C1, UT1.C1 FROM ET1,UT1 WHERE ET1.C1 = X'C1C2C3'AND UT1.C1 = X'414243'

Interpreted as:

SELECT ET1.C1, UT1.C1 FROM ET1,UT1 WHERE ET1.C1 = 'ABC' AND UT1.C1 = ' âä'



Correct encoding:

SELECT ET1.C1, UT1.C1 FROM ET1,UT1 WHERE ET1.C1 = X'C1C2C3'AND UT1.C1 = X'C1C2C3'

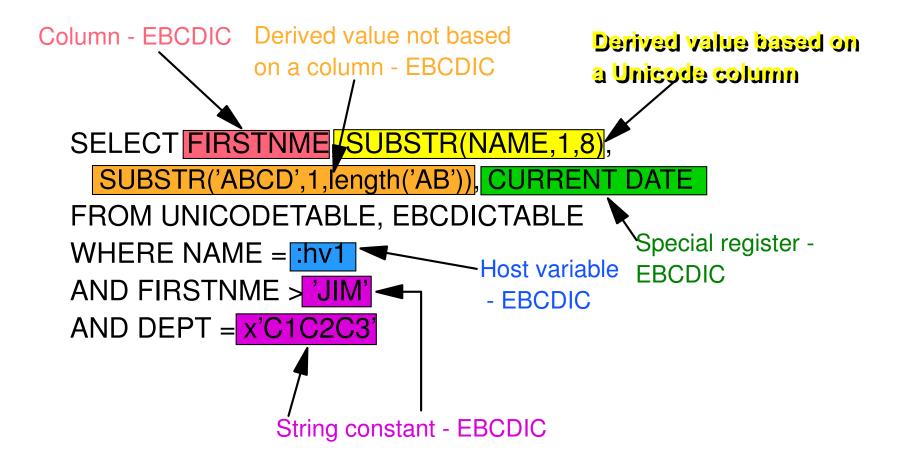




Some Examples if using Unicode in V8



Application encoding scheme: EBCDIC







Some Examples if using Unicode in V8

UTILITIES

- Utility Control Statements can be specified in EBCDIC or UNICODE
 - ✓ DB2 detects which encoding scheme is being used
 - ✓ Objects names in messages will be in EBCDIC.

EBCDIC

```
//SYSIN DD *
   COPY TABLESPACE A.B.
```

UNICODE

 //SYSIN DD * "à|&\$"èà<àë&äà""â"""""""







Conclusion

Don't be afraid

Cyrillic

X 046A	O47A	Й ,	K
X 046B	O47B	Й ,	К 049В

Mongolian

.2	\sim
1874	1884
7	3
1875	1885
4	333
1876	1886

CJK

路	揺	綾 Ps1
露	路	菱 P958
魯	雷	陵 1959

But be prepared

Basic Latin

1	A 0041	Q 0051
2	B 0042	R 0052
3	C 0043	S 0053
4	D44	T



Don't wait until V8 migration: Start now