Navigating the pitfalls of cross platform copies

Kai Stroh, UBS Hainer GmbH

Overview

Motivation

- Some people are looking for a way to copy data from Db2 for z/OS to other platforms
- Reasons include:
  - Number crunching / reporting on other platforms may be cheaper
  - Build test environments on development servers / local machines
  - Db2 for z/OS DBAs are becoming scarce

UBS HAINER

Motivation

- Not always feasible
- Legacy applications: difficult
  - Host batch jobs
  - Static SQL with DBRMs
  - CICS and other middleware
- “Modern” applications: easier
  - Web applications
  - JDBC-only applications

UBS HAINER

Why are cross platform copies difficult?

- Different DDL dialects
- Different data types
  - Some not available at all
  - Some have different ranges of valid values
- File formats
- Lossless data transfer to another operating system

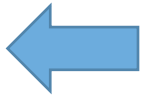
UBS HAINER

Different DDL dialects

Perceived difficulties

- This was our estimation how difficult it would be:

Target system	Difficulty
Other DB2 for z/OS	Easy
Db2 for LUW	Fairly easy
MSSQL Server	Manageable
Oracle	God help us all



This is what the customer wanted

Data type differences

Db2	Oracle
CHAR / VARCHAR	CHAR / VARCHAR2
SMALLINT / INTEGER / BIGINT	NUMBER
DECIMAL	NUMBER
REAL / DOUBLE	BINARY_FLOAT / BINARY_DOUBLE
DECFLOAT	- (Approximate with DOUBLE)
DATE	DATE
TIME	- (Can use DATE or INTERVAL DAY TO SECOND)
TIMESTAMP / TIMESTAMP WITH TIME ZONE	TIMESTAMP/ TIMESTAMP WITH TIME ZONE
BLOB, CLOB, DBCLOB	BLOB, CLOB
BINARY / VARBINARY / FOR BIT DATA	RAW (max. 2,000 bytes, use BLOB if required)
XML	XMLType

DDL generation

- Columns with NOT NULL:
  - Most database systems have a catalog column to reflect this
  - Oracle does not have such a column
  - Instead, an implicit check constraint is used
  - When copying form Oracle to a different system, this can affect eligibility of a column in foreign keys

UBS HAINER

DDL generation

- Default values:
  - Db2 for z/OS: `TIMESTAMP WITH DEFAULT`
  - Db2 for LUW: `TIMESTAMP WITH DEFAULT CURRENT TIMESTAMP`
  - Oracle: `DATETIME DEFAULT CURRENT_TIMESTAMP`
  - MSSQL Server: `DATETIME DEFAULT GETDATE()`
- Similar situations with other default values, such as special registers

DDL generation

- Identity columns:
  - Db2 for z/OS, Db2 for LUW:

    ```
    INTEGER NOT NULL GENERATED BY DEFAULT AS IDENTITY
    ```

  - Oracle (starting with Oracle 12c):

    ```
    NUMBER GENERATED BY DEFAULT ON NULL AS IDENTITY
    ```

  - MSSQL Server:

    ```
    INTEGER IDENTITY(1, 1)
    ```

DDL generation

- Identity columns in Oracle (up to Oracle 11g):

```
CREATE TABLE departments (
  ID              NUMBER(10)    NOT NULL,
  ...);

CREATE SEQUENCE dept_seq START WITH 1;

CREATE OR REPLACE TRIGGER dept_bir
BEFORE INSERT ON departments FOR EACH ROW
BEGIN SELECT dept_seq.NEXTVAL INTO :new.id FROM dual;
END;
```

Integer quirks

- Oracle only uses NUMBER (=DECIMAL) and floating point data types
- No explicit 16, 32 or 64 bit integer types
- SMALLINT = NUMBER(6, 0)
- INTEGER = NUMBER(11, 0)
- BIGINT = NUMBER(19, 0)

UBS HAINER

String quirks

- Empty strings are treated as NULL values
- This is potentially a huge problem for application logic
  - `WHERE description = ''` is never true
  - Columns that are NOT NULL cannot be set to an empty string
  - Allowing the columns to contain NULL values requires changes in the applications (must use wasNull() to avoid NullPointerExceptions)
- Oracle will not change VARCHAR2 semantics, but might VARCHAR in the future

UBS HAINER

Other string problems

- Maximum length of VARCHAR2 or NVARCHAR2 columns:
  - Oracle 11g: limit is 4,000 bytes
  - Oracle 12c: limit is 32,767 bytes
  - Initialization parameter MAX_STRING_SIZE must be EXTENDED
- Maximum length of CHAR columns:
  - Always 2,000 characters

UBS HAINER

Oracle quirks

- Oracle is very strict with timestamp values
  - Example: During switch to DST, one hour is “missing”
  - Oracle rejects timestamp values that are in this missing hour
- Symptom: Individual rows cannot be inserted into an Oracle table for no apparent reason

UBS HAINER

Date and time types

- Db2 and MSSQL have a pure TIME data type
- MSSQL Server and Oracle have data types that represent time intervals
- To copy, convert to character data type
- Disadvantage: can break queries (however, in this scenario, the application is tied to one specific data base anyway)

UBS HAINER

DB2 z/OS quirks

- Db2 accepts 24:00:00 as time
- A timestamp can be 9999-12-32-24.00.00.000000, which technically is in the year 10,000
- These rows were rejected by Oracle and required manual intervention

UBS HAINER

XML data

- Easier than anticipated
- Always use external (human-readable) representation
- XML documents have a DTD and can be validated on import
- Slow

UBS HAINER

LOBs

- Transfer via file reference
  - z/OS: absolute data set name
  - LOBs are unloaded into PDS-E, limit of ~500,000 members
  - Other systems: relative data set name possible
- Alternative: Inline LOBs in Unload data sets
  - Can result in spanned records, make sure to transfer them correctly

File transfer

File formats

- Common denominator: CSV
- All other formats are usually platform specific
  - Platform = combination of operating system and database system
  - Different endianness, different representation of floating point numbers
- Use hexadecimal strings for binary data

Correctly handling text

- Goal: Transfer data sets with non-ASCII characters
- Reminder:
  - Pure ASCII contains 128 characters
  - 33 of which (00 – 1F and 7F) are non-printable control characters
- Non-ASCII: Diacritics (â, ç, è, ñ, ö), æ, ð, þ, ß
- Found most often in proper names, but also in English loan words: café, resumé, naïveté, jalapeño
- Greek, Cyrillic, Chinese, Japanese, Korean, Arabic, etc.

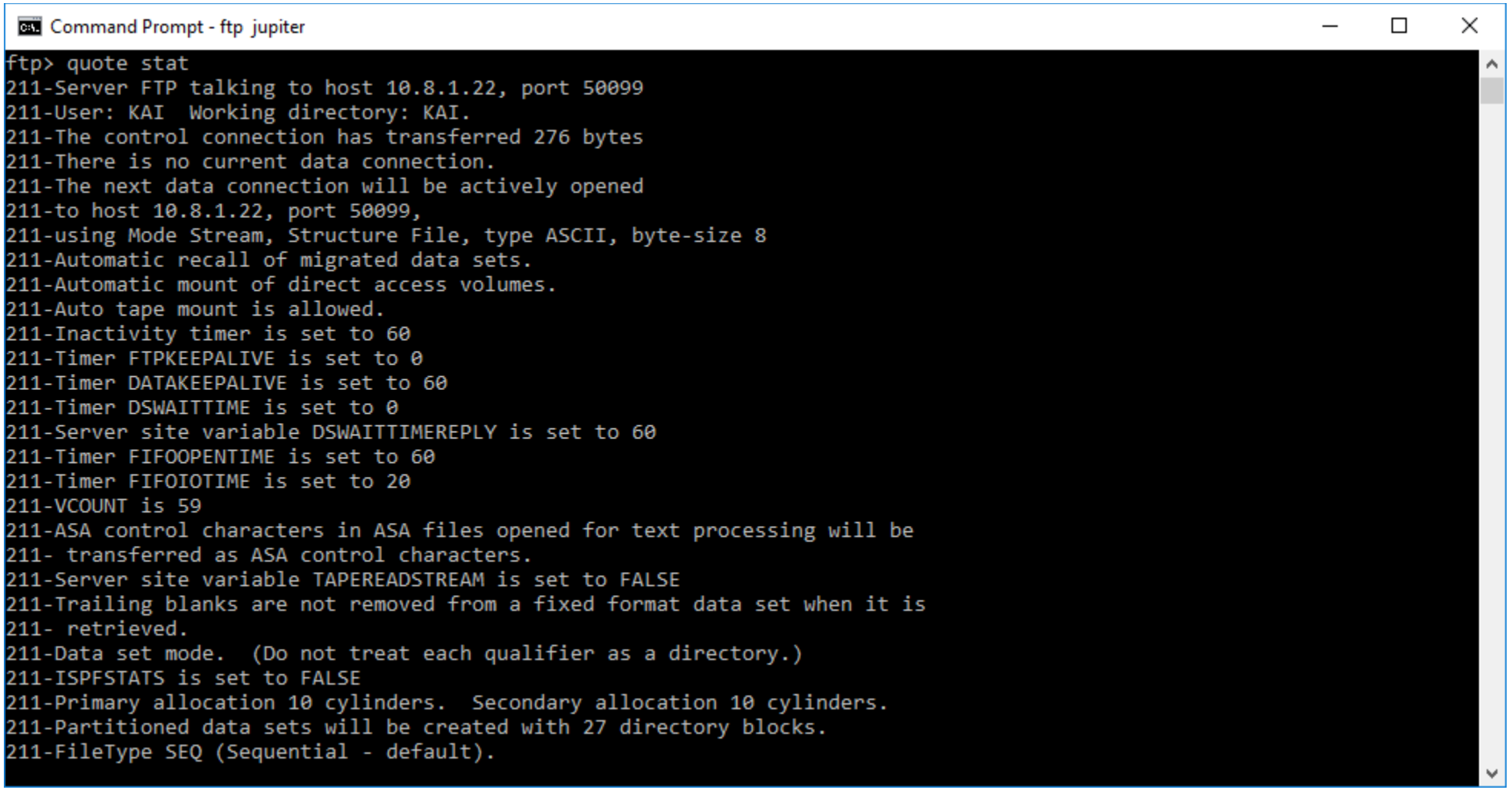
UBS HAINER

Copying files from/to z/OS

- Most common method: FTP or FTPS
- FTP server settings affect file transfer
- Always check current settings by using:
  - `QUOTE STAT` (Windows client)
  - `STAT` (Unix/Linux client)

UBS HAINER

Navigating the pitfalls of cross platform copies



Codepage conversion

```
211-SBDataconn codeset names: IBM-1047,IBM-850
```

- First value = z/OS CCSID, second value = PC CCSID
- Plain text files are assumed to use these CCSIDs, otherwise data corruption will occur
- Change via site command to match actual code pages:

```
ftp> quote site sbdatacon=(ibm-1047,ibm-819)
200 SITE command was accepted
```


UBS HAINER

Common EBCDIC / ASCII code pages

Code page name	Type	CCSID
US ANSI X3.4 ASCII	ASCII	367
INTL EBCDIC	EBCDIC	500
ISO8859-1	ASCII	819
LATIN-1 PC	ASCII	850
ISO8859-15	ASCII	923
LATIN OPEN SYS EB	EBCDIC	1047
AUS/GERM ECECP	EBCDIC	1141
ITALIAN ECECP	EBCDIC	1144
FRENCH ECECP	EBCDIC	1147

Common Unicode code pages

Code page name	Type	CCSID
UTF-16 BE	Unicode	1200
UTF-16 LE	Unicode	1202
UTF-16 BOM	Unicode	1204
UTF-8	Unicode	1208
UTF-32 BE	Unicode	1232
UTF-32 LE	Unicode	1234
UTF-32 BOM	Unicode	1236

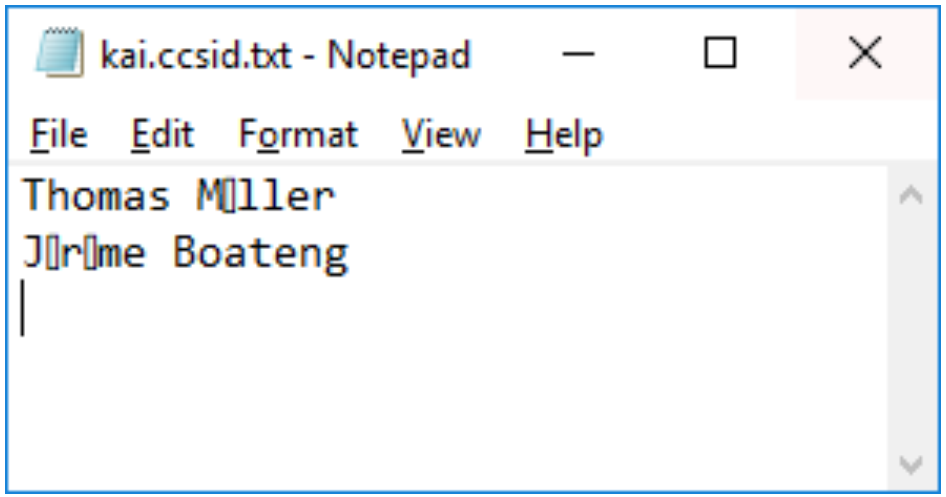
Full list: https://www-01.ibm.com/software/globalization/ccsid/ccsid_registered.html

Example: PC encoding incorrect

Original data set: CCSID 1141
(German/Austrian EBCDIC with Euro symbol)

```
000100 Thomas Müller
       E8998A4DD99894444
       3864120403359000
-------------------------
000200 Jérôme Boateng
       D59C984C98A898444
       119B4502613557000
```

```
quote site sbdatacon=(1141,367)
```



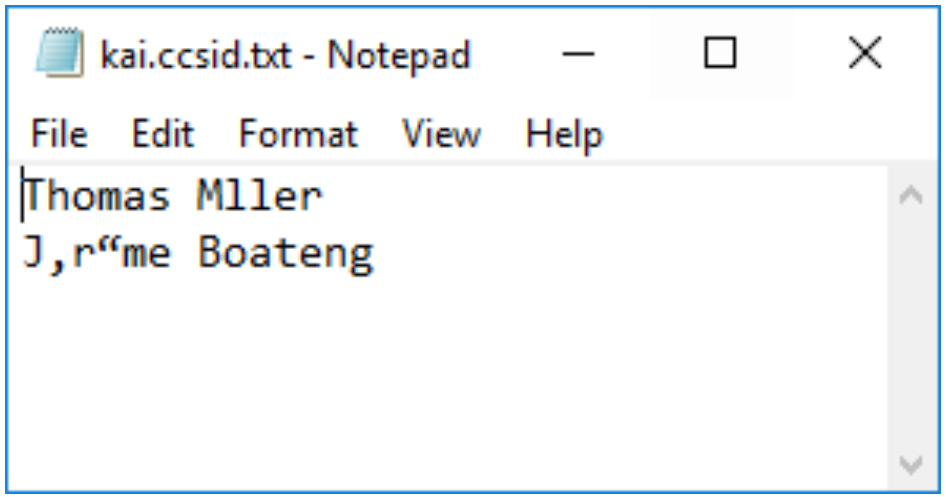

```
Offset(h)  00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F

00000000   54 68 6F 6D 61 73 20 4D 1A 6C 6C 65 72 0D 0A 4A  Thomas M.ller..J
00000010   1A 72 1A 6D 65 20 42 6F 61 74 65 6E 67 0D 0A     .r.me Boateng..
```

Example: PC encoding unexpected

Original data set: CCSID 1141
(German/Austrian EBCDIC with Euro symbol)

```
000100 Thomas Müller
       E8998A4DD99894444
       3864120403359000
-------------------------
000200 Jérôme Boateng
       D59C984C98A898444
       119B4502613557000
```

```
quote site sbdatacon=(1141,850)
```



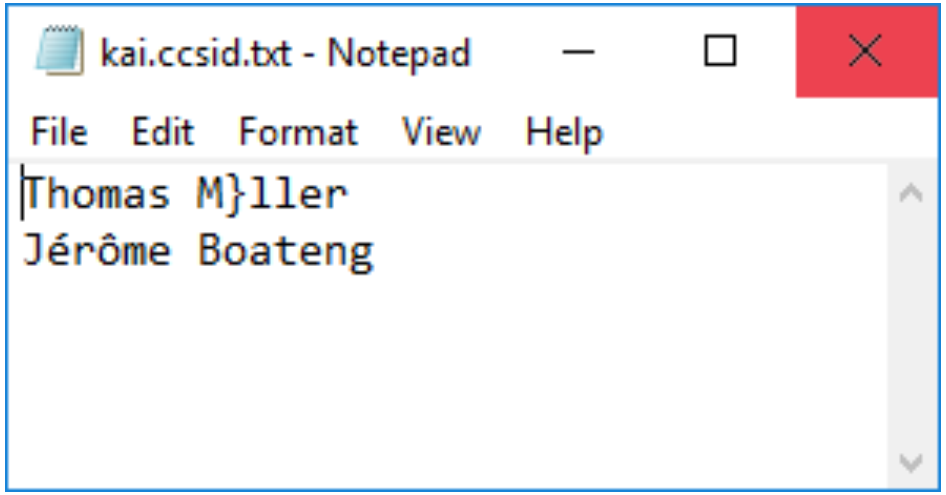

```
Offset(h)  00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
00000000   54 68 6F 6D 61 73 20 4D 81 6C 6C 65 72 0D 0A 4A  Thomas M.ller..J
00000010   82 72 93 6D 65 20 42 6F 61 74 65 6E 67 0D 0A     ,r“me Boateng..
```

Example: Host encoding incorrect

Original data set: CCSID 1141
(German/Austrian EBCDIC with Euro symbol)

```
000100 Thomas Müller
       E8998A4DD99894444
       3864120403359000
-------------------------
000200 Jérôme Boateng
       D59C984C98A898444
       119B4502613557000
```

```
quote site sbdatacon=(1047,819)
```



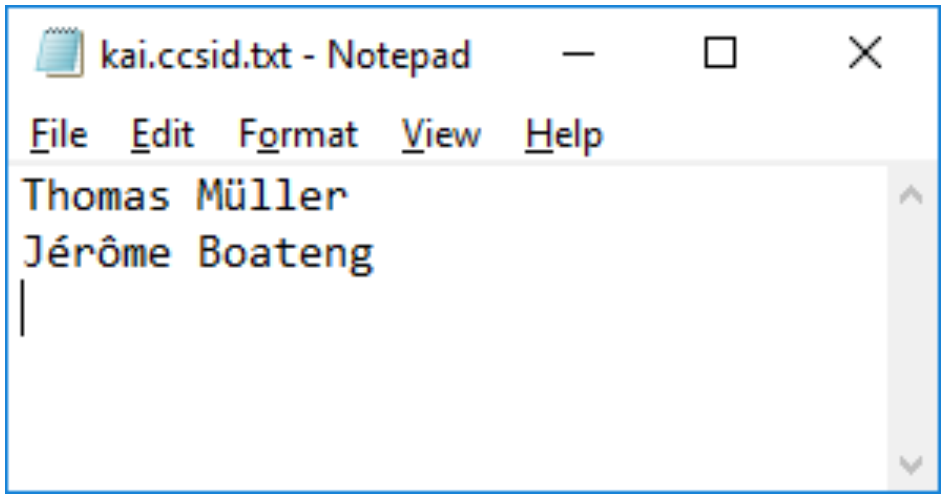

```
Offset(h)  00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F

00000000   54 68 6F 6D 61 73 20 4D 7D 6C 6C 65 72 0D 0A 4A  Thomas M}ller..J
00000010   E9 72 F4 6D 65 20 42 6F 61 74 65 6E 67 0D 0A     érôme Boateng..
```

Example: Both encodings correct

Original data set: CCSID 1141
(German/Austrian EBCDIC with Euro symbol)

```
000100 Thomas Müller
       E8998A4DD99894444
       3864120403359000
------------------------
000200 Jérôme Boateng
       D59C984C98A898444
       119B4502613557000
```

```
quote site sbdatacon=(1141,819)
```



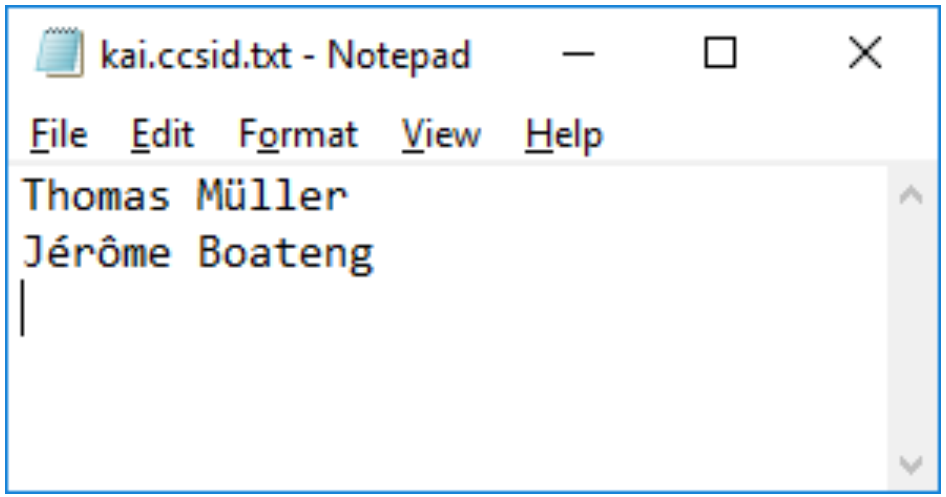

```
Offset(h)  00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
00000000   54 68 6F 6D 61 73 20 4D FC 6C 6C 65 72 0D 0A 4A  Thomas Müller..J
00000010   E9 72 F4 6D 65 20 42 6F 61 74 65 6E 67 0D 0A     érôme Boateng..
```

Example: Download EBCDIC as UTF-8

Original data set: CCSID 1141
(German/Austrian EBCDIC with Euro symbol)

```
000100 Thomas Müller
       E8998A4DD99894444
       3864120403359000
-------------------------
000200 Jérôme Boateng
       D59C984C98A898444
       119B4502613557000
```

```
quote site encoding=mbcs
quote site mbdatacon=(1141,utf-8)
```




```
Offset(h)  00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F

00000000   54 68 6F 6D 61 73 20 4D C3 BC 6C 6C 65 72 0D 0A  Thomas MÃ¼ller..
00000010   4A C3 A9 72 C3 B4 6D 65 20 42 6F 61 74 65 6E 67  JÃ©rÃ´me Boateng
00000020   0D 0A                                            ..
```

How to use UTF-8 everywhere

- Unload Db2 for z/OS data with:

```
UNLOAD DATA FROM TABLE ...
UNICODE CCSID(1208,1208,1208) DELIMITED
```

- Download via FTP using:

```
quote site mbdatacon=(utf-8,utf-8)
quote site encoding=mbcs
ascii
get <host-dsn> <pc-filename>
```

- Table must not contain binary columns or binary data in text columns (this includes line breaks, tab stops etc. in text columns)

Other important server settings

- LRECL
- RECFM
- CYL PRI=*m* SEC=*n*
- DSNTYPE=LARGE
- TRAILINGBLANKS / NOTRAILINGBLANKS
- TRUNCATE / NOTRUNCATE
- WRAP / NOWRAP

UBS HAINER

Transferring binary VB data sets

- Problem: ASCII end of line characters (0x0D0A or Windows, 0x0A in Unix/Linux) could be actual data
- Use `BIN`, then use `QUOTE STRU R` to enter record mode
- Prepare your data:
  - Indicate end of each record with 0xFF01
  - Indicate end of file with 0xFF02
  - Inside the data, replace 0xFF with 0xFFFF
- No code page conversion will be done

Example

- Binary transfer with file structure: Record ends are lost

```
 -----------
1.3
F0F
103
 -----------
1234...8
FFFF00FF
123412F8
 -----------
```

```
Offset(h)  00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
00000000   F1 00 F3 F1 F2 F3 F4 01 02 FF F8                ñ.óñòóô..ÿø
```

First record ends right here, but there is nothing that would indicate that

Example

- Binary transfer with record structure

```
 -----------
1.3
F0F
103
 -----------
1234...8
FFFF00FF
123412F8
 -----------
```

```
Offset(h)  00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
00000000   F1 00 F3 FF 01 F1 F2 F3 F4 01 02 FF FF F8 FF 01  ñ.óÿ.ñòóô..ÿÿøÿ.
00000010   FF 02                                            ÿ.
```

End of record

End of record

End of file

Masked data byte 0xFF

Transferring spanned data sets

- Common when unloading LOB or XML data
- Possible via FTP if you use `quote site rdw`
- Downloaded file will include a record descriptor word (RDW) or segment descriptor word (SDW) for each record
- Problem: After transfer to PC, raw data must be interpreted and partial records must be joined

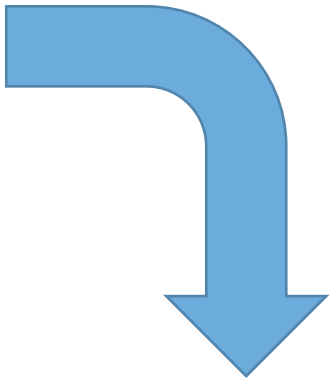
The RDW / SDW

- 4 byte prefix for every record in a VB/VBS data set
- Bytes 1 and 2: Length of the segment, including the SDW
- Byte 3: left 6 bits reserved, right 2 bits:
  - 00 = Non-spanned record
  - 01 = First segment of a spanned record
  - 10 = Last segment of a spanned record
  - 11 = Middle segment of a spanned record
- Byte 4: reserved

UBS HAINER

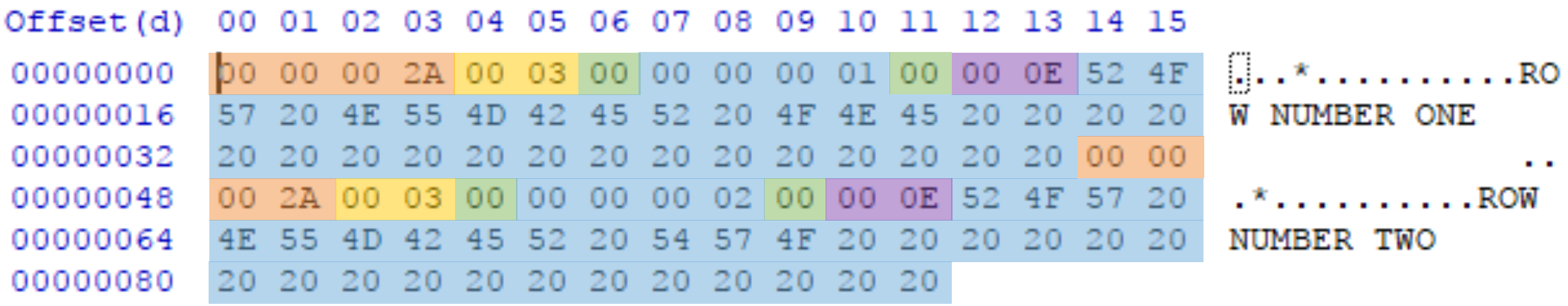
Better solution: Use the z/OS Unix file system

COL1 INT	COL2 VARCHAR(32)
1	ROW NUMBER ONE
2	ROW NUMBER TWO



```
TEMPLATE T1 PATH '/u/SYSREC.bin'
UNLOAD DATA
   FROM TABLE TEST.TABLE123
   UNLDDN T1
```

```
Offset(d)  00 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15
00000000   00 00 00 2A 00 03 00 00 00 00 01 00 00 0E 52 4F  ...*..........RO
00000016   57 20 4E 55 4D 42 45 52 20 4F 4E 45 20 20 20 20  W NUMBER ONE
00000032   20 20 20 20 20 20 20 20 20 20 20 20 20 20 00 00                ..
00000048   00 2A 00 03 00 00 00 00 02 00 00 0E 52 4F 57 20  .*..........ROW
00000064   4E 55 4D 42 45 52 20 54 57 4F 20 20 20 20 20 20  NUMBER TWO
00000080   20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
```



```
Offset(d)  00 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15
00000000   00 00 00 2A 00 03 00 00 00 00 01 00 00 0E 52 4F  ...*..........RO
00000016   57 20 4E 55 4D 42 45 52 20 4F 4E 45 20 20 20 20  W NUMBER ONE
00000032   20 20 20 20 20 20 20 20 20 20 20 20 20 20 00 00                ..
00000048   00 2A 00 03 00 00 00 00 02 00 00 0E 52 4F 57 20  .*..........ROW
00000064   4E 55 4D 42 45 52 20 54 57 4F 20 20 20 20 20 20  NUMBER TWO
00000080   20 20 20 20 20 20 20 20 20 20 20 20
```

- **Length** of the row, **not** including this length field (4 byte field)
- **OBID** of the table (2 byte field)
- For each column:
  - **NULL indicator** (1 byte field, exists if column can be NULL)
  - **Column length** (2 byte field, exists if varying length column)
  - **Data**

Questions? Comments?

Thank you for your attention!

For more information visit
www.ubs-hainer.com
or send an email to
s.tursman@ubs-hainer.com