cdisc

2023
CHINA
INTERCHANGE
BEIJING | 25-26 AUGUST

How invisible encoding influence data storage and display

罗斯元 Siyuan Luo, Senior Statistical Programmer
Biostatistics and Computing Science, Elixir

Meet the Speaker

罗斯元 Siyuan Luo

Title: Senior Statistical Programmer

Organization: Elixir (Shanghai) Clinical Research Co., Ltd.

Siyuan graduated from Fudan University majoring in preventative medicine, has 4 years experience in statistical programming for CRO since 2019

Disclaimer and Disclosures

- *The views and opinions expressed in this presentation are those of the author(s) and do not necessarily reflect the official policy or position of CDISC.*

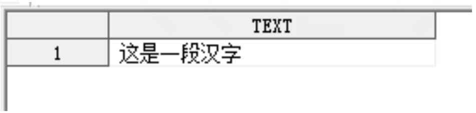
Agenda

1. Introduction to encoding
2. Encoding schemes
3. Encoding and Submission

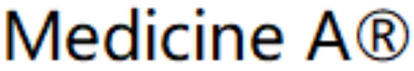
Introduction to encoding

Introduction to encoding

Sender A

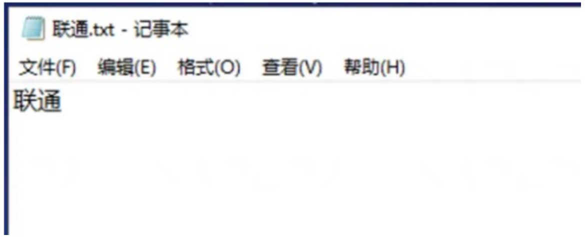


	TEXT
1	这是一段汉字



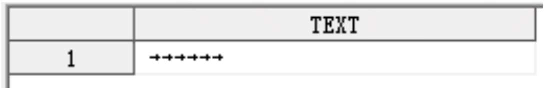
Medicine A®

输入“联通”两个字的时候：

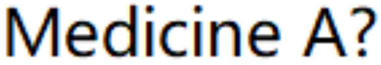


联通

Receiver B

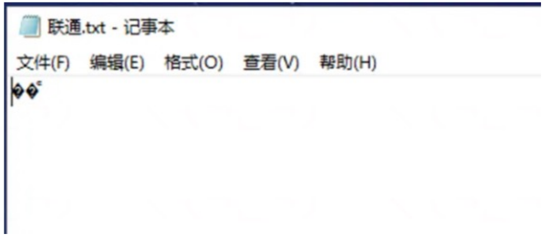


	TEXT
1	++++++



Medicine A?

保存并关闭文件，双击打开后的结果：



��ͨ

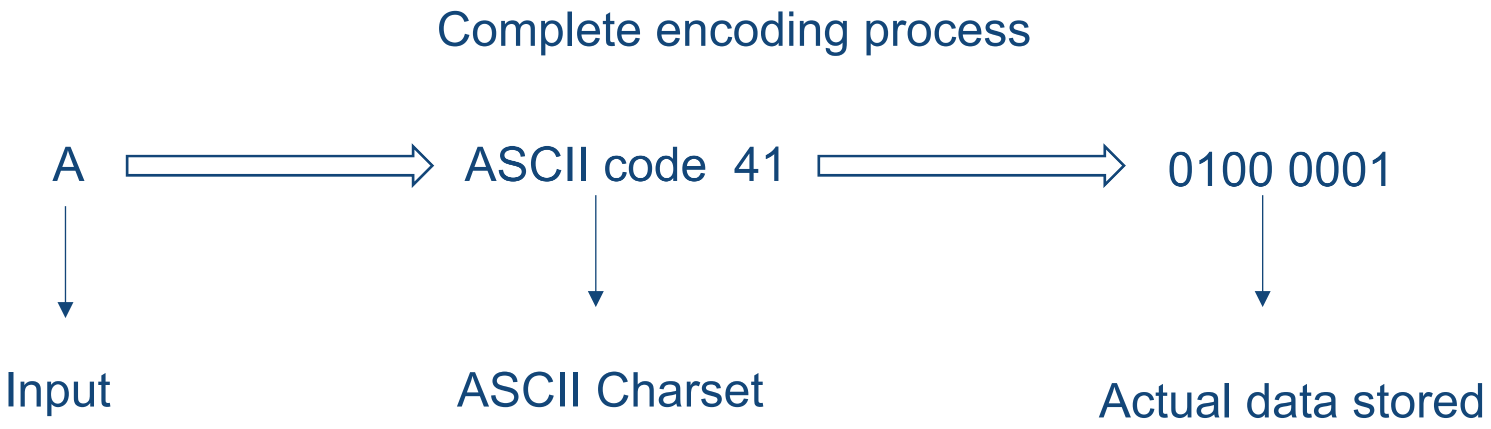
Introduction to encoding

- Garbage characters? Garbled? Gibberish?

- All means your encoding is wrong!

- What is encoding?
  - Encoding is the process of converting data into a format required for a number of information processing needs.
  - In computers, encoding is the process of putting a sequence of characters (letters, numbers, punctuation, and certain symbols) into a specialized format for efficient transmission or storage.

Introduction to encoding



Complete encoding process

A → ASCII code 41 → 0100 0001

Input

ASCII Charset

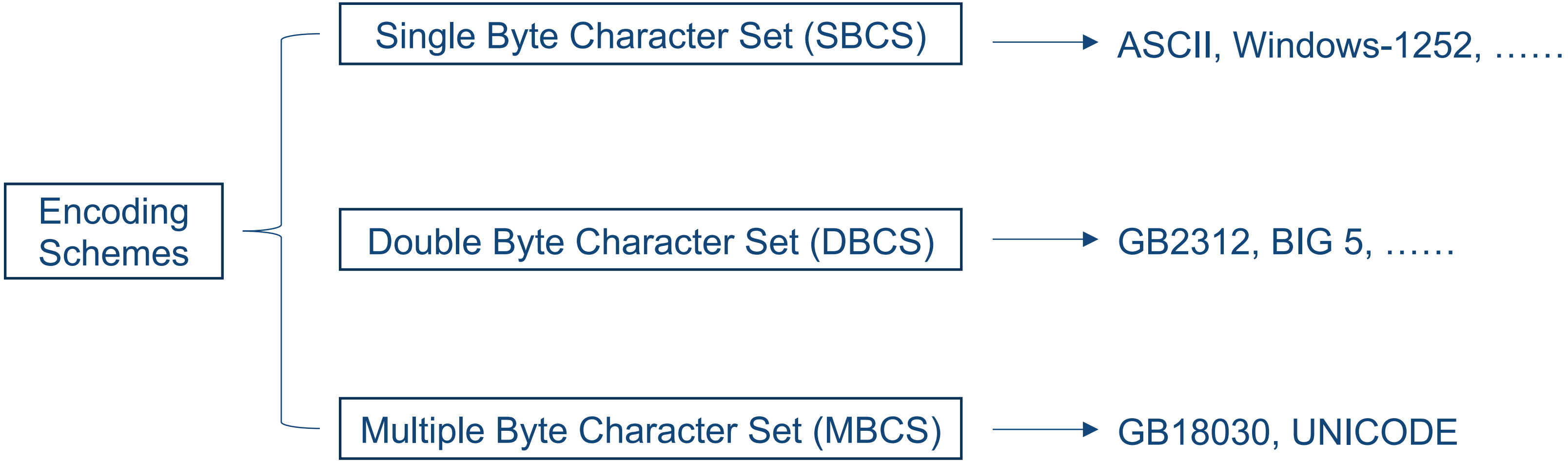
Actual data stored

Computers are based on the binary numbers. Computers can represent numbers using binary code in the form of digital 1s and 0s inside the central processing unit (CPU) and RAM. These digital numbers are electrical signals that are either on or off inside the CPU or RAM. To make sense of complicated data, your computer has to convert it into binary.

Encoding schemes

Encoding schemes

We can design innumerable encoding schemes as long as that characters are one-to-one mapping to data stored/transmitted.



Encoding Schemes

Single Byte Character Set (SBCS) → ASCII, Windows-1252, ……

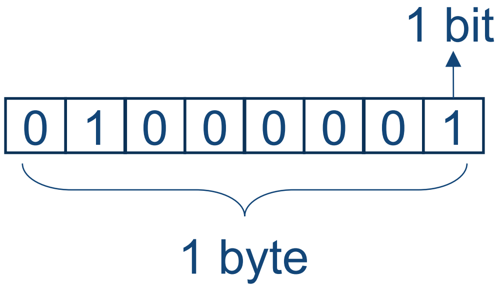
Double Byte Character Set (DBCS) → GB2312, BIG 5, ……

Multiple Byte Character Set (MBCS) → GB18030, UNICODE

Encoding schemes

Code points are usually represented in hexadecimal, because:

1. All data will be converted into binary before computer processing.
2. Bit (binary digit) is the smallest unit of data transmission and processing.
3. Byte is the smallest unit of data storage, 1 byte = 8 bits.



1 bit

0	1	0	0	0	0	0	1

1 byte

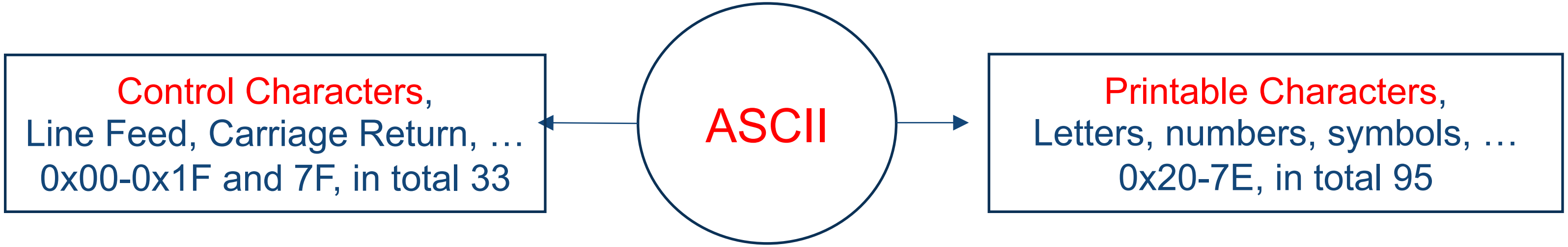
To refer to characters in an unambiguous way, each character is associated with a number, called a code point. Usually, we use hexadecimal to represent the number of code points.

In hexadecimal, we can use 2 digits to represent any data of 1 byte size.

Encoding schemes – ASCII

ASCII

in full American Standard Code for Information Interchange, designed by America National Standard Insititute (ANSI) in 1960s, a standard data-encoding format for electronic communication between computers. ASCII assigns standard numeric values to letters, numerals, punctuation marks, and other characters used in computers. Code points range from 0 to 127.

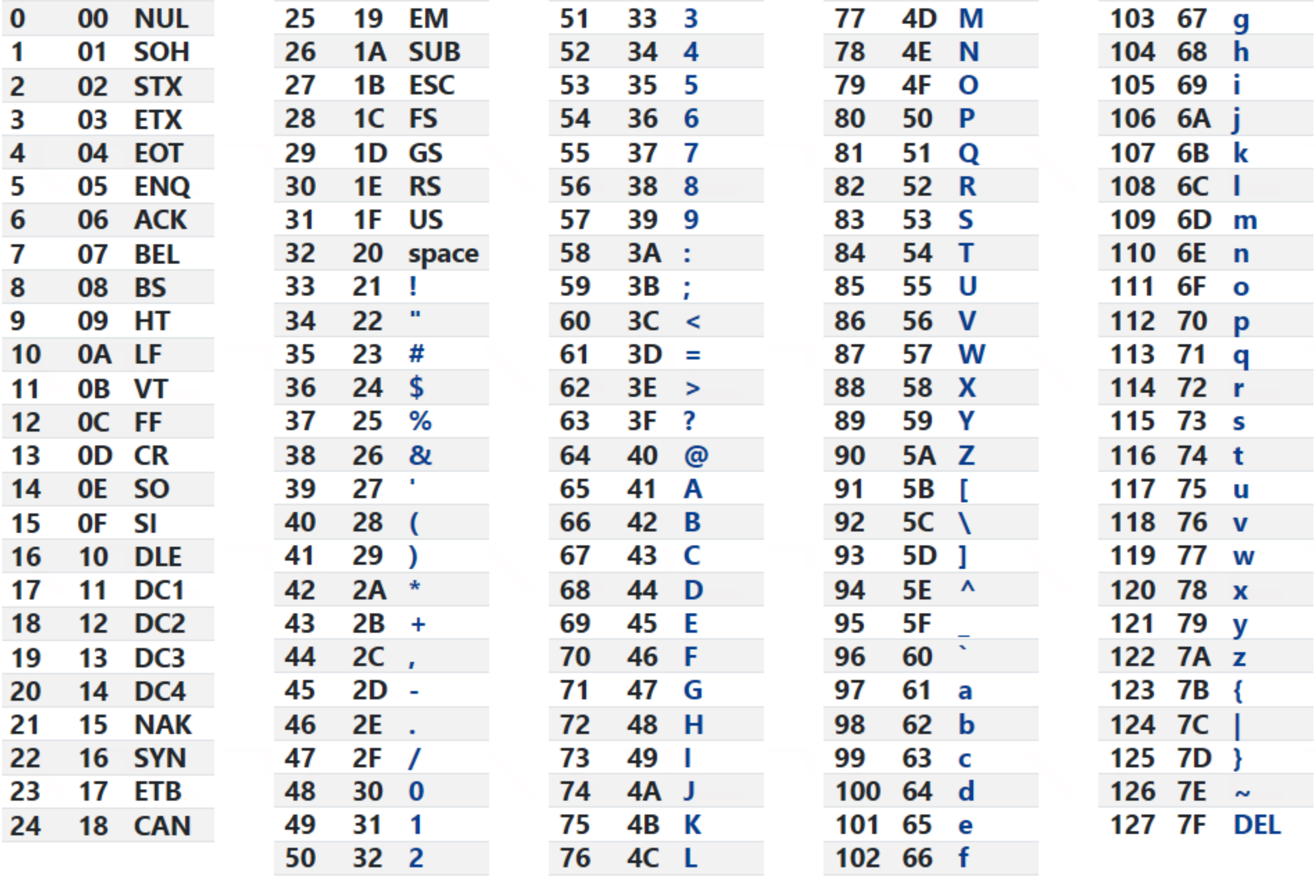


Control Characters,
Line Feed, Carriage Return, …
0x00-0x1F and 7F, in total 33

ASCII

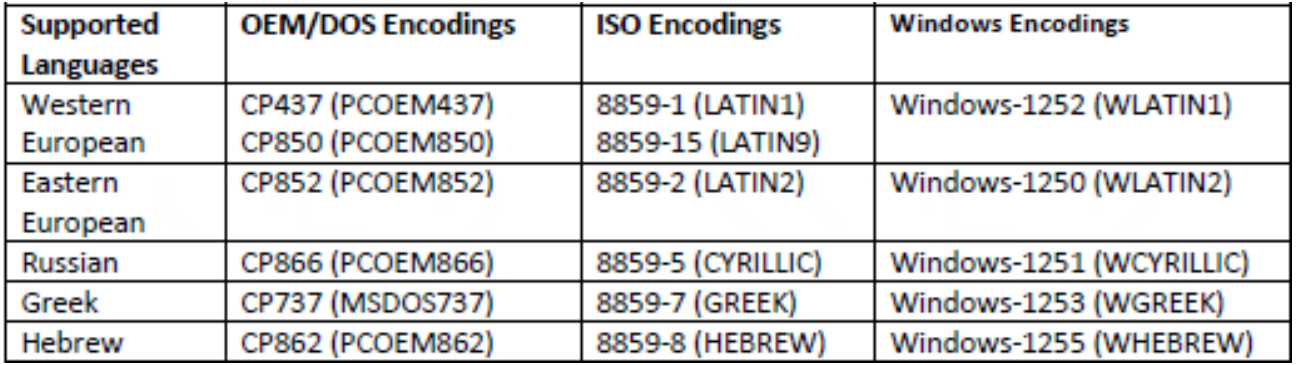
Printable Characters,
Letters, numbers, symbols, …
0x20-7E, in total 95

Encoding schemes – ASCII Table



0	00	NUL
1	01	SOH
2	02	STX
3	03	ETX
4	04	EOT
5	05	ENQ
6	06	ACK
7	07	BEL
8	08	BS
9	09	HT
10	0A	LF
11	0B	VT
12	0C	FF
13	0D	CR
14	0E	SO
15	0F	SI
16	10	DLE
17	11	DC1
18	12	DC2
19	13	DC3
20	14	DC4
21	15	NAK
22	16	SYN
23	17	ETB
24	18	CAN
25	19	EM
26	1A	SUB
27	1B	ESC
28	1C	FS
29	1D	GS
30	1E	RS
31	1F	US
32	20	space
33	21	!
34	22	"
35	23	#
36	24	$
37	25	%
38	26	&
39	27	'
40	28	(
41	29	)
42	2A	*
43	2B	+
44	2C	,
45	2D	-
46	2E	.
47	2F	/
48	30	0
49	31	1
50	32	2
51	33	3
52	34	4
53	35	5
54	36	6
55	37	7
56	38	8
57	39	9
58	3A	:
59	3B	;
60	3C	<
61	3D	=
62	3E	>
63	3F	?
64	40	@
65	41	A
66	42	B
67	43	C
68	44	D
69	45	E
70	46	F
71	47	G
72	48	H
73	49	I
74	4A	J
75	4B	K
76	4C	L
77	4D	M
78	4E	N
79	4F	O
80	50	P
81	51	Q
82	52	R
83	53	S
84	54	T
85	55	U
86	56	V
87	57	W
88	58	X
89	59	Y
90	5A	Z
91	5B	[
92	5C	\
93	5D	]
94	5E	^
95	5F	_
96	60	`
97	61	a
98	62	b
99	63	c
100	64	d
101	65	e
102	66	f
103	67	g
104	68	h
105	69	i
106	6A	j
107	6B	k
108	6C	l
109	6D	m
110	6E	n
111	6F	o
112	70	p
113	71	q
114	72	r
115	73	s
116	74	t
117	75	u
118	76	v
119	77	w
120	78	x
121	79	y
122	7A	z
123	7B	{
124	7C	|
125	7D	}
126	7E	~
127	7F	DEL

Encoding schemes – Other SBCS



Supported Languages	OEM/DOS Encodings	ISO Encodings	Windows Encodings
Western European	CP437 (PCOEM437) CP850 (PCOEM850)	8859-1 (LATIN1) 8859-15 (LATIN9)	Windows-1252 (WLATIN1)
Eastern European	CP852 (PCOEM852)	8859-2 (LATIN2)	Windows-1250 (WLATIN2)
Russian	CP866 (PCOEM866)	8859-5 (CYRILLIC)	Windows-1251 (WCYRILLIC)
Greek	CP737 (MSDOS737)	8859-7 (GREEK)	Windows-1253 (WGREEK)
Hebrew	CP862 (PCOEM862)	8859-8 (HEBREW)	Windows-1255 (WHEBREW)

Encoding schemes – GB2312

The Facts

1. Either ASCII or Windows-125 or any western encoding else is single-byte.
2. A single byte can only represent 2^8=256 different characters.
3. There are far more than 256 characters in Chinese!

Obviously, any single-byte encoding standard can not meet the needs for Chinese characters.

China developed its own double-byte encoding charset, using up to 2 bytes to represent characters

GB2312 is the first encoding enforced by China government in 1981, it is widely used in Chinese Mainland, Singapore.

Encoding schemes – GB2312, GBK, GB18030

GB2312

Since 1981, double-byte character set, variable length encoding. Uses 1 byte code to represent ASCII codes, 2 byte for Chinese characters. Has Collects 6763 commonly used Chinese characters, and other characters including Latin letters, Greek letters, Japanese hiragana and katakana, Russian Cyrillic letters, etc.

GBK

Since 1995, double-byte character set, extension of GB2312, has 23940 code points, including 21003 Chinese characters, completely compatible with GB2312.

GB18030

Extension of GB2312 and GBK established by the government of China in 2005, uses a 4-byte variable length encoding to match the capacity of the surrogate character mechanism introduced in Unicode 2.0. Include 87887 Chinese characters. It becomes the enforced encoding standard from August 1st 2023.

Encoding standard – Unicode

Unicode

The Unicode Standard is the universal character encoding designed to support the worldwide interchange, processing, and display of the written texts of the diverse languages and technical disciplines of the modern world.

The first version of Unicode was introduced by Unicode Consortium in 1991.

It is designed to display all the characters in the world!

Development History

2 bytes → 4 bytes

U+0000 – U+FFFF

65536 code points

U+0000 – U+10FFFF

17 plains, 17*65536=1 million+ code points!

Encoding schemes – UTF-8, UTF-16, UTF-32

Unicode is a encoding standard, which defines code points of each character. But how to store and transmit data, how to encode and decode are not limited. So 3 UTF encoding schemes are invented.

UTF is short for Unicode Transformation Format, is an algorithmic mapping from every Unicode code point to a unique byte sequence. There are 3 types, UTF-8, UTF-16, UTF-32.

Character	Unicode	UTF-8	UTF-16	UTF-32
A	U+0041	0x41	0x0041	0x00000041

UTF-8: 1 to 4 bytes variable length encoding, completely compatible with ASCII. Most commonly used Unicode encoding schemes. Space-saving, convenient for data storage and transmission.

UTF-16: 2 or 4 bytes variable length encoding. For Unicode characters with code points in [U+0000-U+FFFF], their encoding are the same as their code points in Unicode. Surrogate pairs are used in the encoding of remaining characters ([U+10000-U+10FFFF]).

UTF-32: 4 bytes fixed length encoding, any single character is encoded in a 4 bytes (32 bits) code unit. It takes too much space, so is rarely used.

Encoding schemes – UTF-8

How to convert characters into UTF-8 encoding?

1. For 1 byte characters, first bit of byte is set to 0, the remaining 7 bits are the Unicode code points of the characters, which means for ASCII characters, UTF-8 is equal to ASCII.

2. For n-byte characters (n>1), the first n bits of the first byte are set to 1, the n+1 bit is set to 0, and the first 2 bits of the following bytes are set to 10. The remaining binary bits are Unicode code points of the characters.

Start	End	Byte order	Byte 1	Byte 2	Byte 3	Byte 4
U+0000	U+007F	1	0xxxxxxx			
U+0080	U+07FF	2	110xxxxx	10xxxxxx		
U+0800	U+FFFF	3	1110xxxx	10xxxxxx	10xxxxxx	
U+10000	U+1FFFF	4	11110xxx	10xxxxxx	10xxxxxx	10xxxxxx

Encoding schemes – UTF-8

Character	汉
Unicode code point (Hexadecimal)	U+6C49
Unicode code point (Binary)	**0110 1100 0100 1001**
UTF-8 code (Binary)	**1110 0110 1011 0001 1000 1001**
UTF-8 code (Hexadecimal)	E6B189

Encoding schemes – UTF-8

Why “联通” becomes garbage characters in windows notepad?

In the old version of windows notepad, the default encoding is ANSI (GB2312 in China).

Characters	联通
GB2312 code point	C1 AA CD 18
GB2312 code point (Binary)	11000001 10101010 11001101 10101000

Windows notepad thinks this is a text file in UTF-8 encoding. Code points of the texts are wrongly resolved, so they become garbage characters.

Encoding and Submission

Requirements for encoding in submission

NMPA	FDA
Language: Chinese	English only
Dataset format: XPT v5 or higher	XPT v5 only
Encoding: Not limited, GB2312/UTF-8 both allowed, should be described previously.	ASCII only

Any data with Non-English data should be translated into English before submission to FDA.
Also, the encoding of dataset should be converted into ASCII, which involves encoding conversion.

Some common encoding problems

- External database uses different encoding, which causes garbled text.
- While submitting clinical data to both NMPA and FDA, messy text found when data are translated into English from Chinese.
- Invisible codes cause errors, including control characters and truncated characters and garbled codes.

……

When encoding problems appear, we shall position the garbled codes first.

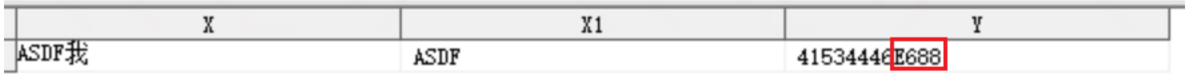
Find garbled codes

- Perl regular expression is a good tool

  Such as finding non-ASCII characters, [^\x00-\x7F], invisible control characters, ([\x00-\x1F]|[\x7F]), and matching any hexadecimal code points.

- Converting to hexadecimal may help

  We can convert text into hexadecimal to help positioning garbled codes.



X	X1	Y
ASDF我	ASDF	41534446E688

Garbled codes

When garbled codes are found, we need to remove them or correct the wrong encoding.

Encoding conversion

Rules:

- Assure that all the characters are represented in the new encoding, or removing the characters which is not in it.
- Find the one-to-one mapping relationship between character sets.
- Sufficient room to hold the converted characters.

Encoding conversion

SAS:

Convert encoding of single variable:
NEW=KCVT(text, intype, outtype, <options,…>)   newvar=kcvt(oldvar, ‘ZEUC’, ’UTF8’);

Convert encoding of datasets:

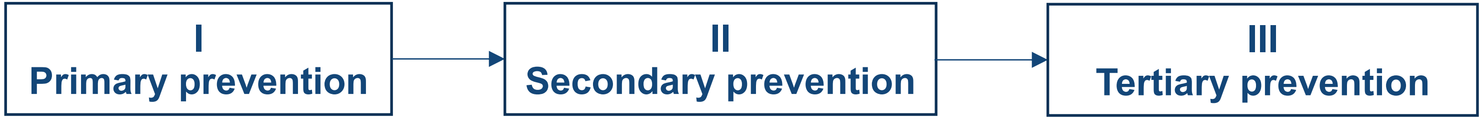
libname inlib cvp ‘path1’ cvpengine=v9 CVPMULTIPLIER=1.5;

libname outlib 'path2' outencoding=utf8; /*zeuc*/

proc copy in=inlib out=outlib; select datasetname; run;

%COPY_TO_NEW_ENCODING(from_dsname, to_dsname, new_encoding)

How to avoid encoding problems



I
Primary prevention

II
Secondary prevention

III
Tertiary prevention

Prevent problems from the source
Use same encoding in the entire process, do not use characters not supported, extract characters in the right way…..

Find the error early
Position garbled codes.

Fix garbled codes
Remove error codes, convert garbled codes into right encoding.

Conclusion

- Encoding is the process of converting data into specialized format for storage and/or transmission.
- Conflicts between different encodings lead to garbled text/codes.
- Unicode supports a wide range of characters from different languages, making it ideal for use in multilingual environments.
- UTF-8 is the most commonly used Unicode encoding scheme.
- As more and more data becomes global in nature, using Unicode encoding standard ensures that your data will be compatible with a wide range of software and systems in the future.
- To avoid encoding errors, the best way is to keep the encoding unchanged in the entire data processing and submission. Then detect and handle garbled text early. Lastly, convert garbled codes into normal codes.

Thank You!

siyuan.luo@ecr-global.com