Talk 2008

Encoding Issues

An overview to understand and be able to handle encoding issues in a better way

Susanne Ebrecht

PostgreSQL Usergroup Germany
PostgreSQL European User Group
PostgreSQL Project

February, 2008

Definition

Character Set

A collection of signs ...

⌘#⊗™⇐℅&⅔♣∃¬ ∴ ⊥≅
{}⊢ ₫€ ŧ←↓⇓↦ ðđŋɸh ĸł«»~

1-9

1	2	3
4	5	6
7	8	9

The German alphabet

AaÄäBbCcDdEeFfGgHhIiJjKkLlMmNnO
oÖöPpQqRrSsßTtUuÜüVvWwXxYyZz

The Greek alphabet

AαBβΓγΔδΕεΖζΗηΘθΙιΚκΛλΜμΝν
ΞξΟοΠπϖΡρΣσςΤτΥυΦφΧχΨψΩω

UNICODE

A-Z

ABCDEFGHIJKLMNOPQRSTUVWXYZ

Roman numbers

I V X L C D M A

ISO-8859-15

NUL	*SOH*	*STX*	*ETX*	*EOT*	*ENQ*	*ACK*	*BEL*	*BS*	*HT*	*LF*	*VT*	*FF*	*CR*	*SO*	*SI*
DLE	*DC1*	*DC2*	*DC3*	*DC4*	*NAK*	*SYN*	*ETB*	*CAN*	*EM*	*SUB*	*ESC*	*FS*	*GS*	*RS*	*US*
SP	!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	*DEL*
PAD	*HOP*	*BPH*	*NBH*	*IND*	*NEL*	*SSA*	*ESA*	*HTS*	*HTJ*	*VTS*	*PLD*	*PLU*	*RI*	*SS2*	*SS3*
DCS	*PU1*	*PU2*	*STS*	*CCH*	*MW*	*SPA*	*EPA*	*SOS*	*SGCI*	*SCI*	*CSI*	*ST*	*OSC*	*PM*	*APC*
NBSP	¡	¢	£	€	¥	Š	§	š	©	ª	«	¬	*SHY*	®	¯
°	±	²	³	Ž	µ	¶	·	ž	¹	º	»	Œ	œ	Ÿ	¿
À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï
Đ	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß
à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï
ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ

Definition Encoding

Implementation of abstract signs, bits and bytes

UTF-32

UTF-8

UTF-7

KOI8-R

KOI8-U

A => 1
B => 2
C => 3
D => 4
...

ASCII

ISO-8859-15

	...0	...1	...2	...3	...4	...5	...6	...7	...8	...9	...A	...B	...C	...D	...E	...F
0...	*NUL*	*SOH*	*STX*	*ETX*	*EOT*	*ENQ*	*ACK*	*BEL*	*BS*	*HT*	*LF*	*VT*	*FF*	*CR*	*SO*	*SI*
1...	*DLE*	*DC1*	*DC2*	*DC3*	*DC4*	*NAK*	*SYN*	*ETB*	*CAN*	*EM*	*SUB*	*ESC*	*FS*	*GS*	*RS*	*US*
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	{	|	}	~	*DEL*
8...	*PAD*	*HOP*	*BPH*	*NBH*	*IND*	*NEL*	*SSA*	*ESA*	*HTS*	*HTJ*	*VTS*	*PLD*	*PLU*	*RI*	*SS2*	*SS3*
9...	*DCS*	*PU1*	*PU2*	*STS*	*CCH*	*MW*	*SPA*	*EPA*	*SOS*	*SGCI*	*SCI*	*CSI*	*ST*	*OSC*	*PM*	*APC*
A...	*NBSP*	¡	¢	£	€	¥	Š	§	š	©	ª	«	¬	*SHY*	®	¯
B...	°	±	²	³	Ž	µ	¶	·	ž	¹	º	»	Œ	œ	Ÿ	¿
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

UTF-16

BIG5

EUC-JP

Encoding

Names in PostgreSQL

- Encoding names are partially defined by the SQL standard
- Encoding names are SQL identifiers
  - Spaces are not allowed

Most of all languages

UTF8 or UNICODE

Japanese

EUC_JP

Turkish

LATIN5 or ISO_8859_9 or ISO88599

Western European

LATIN1 or ISO_8859_1 or ISO88591

Greek

ISO_8859_7

LATIN1 with Euro and accents

LATIN9 or ISO_8859_15 or ISO885915

More informations: http://www.postgresql.org/docs/current/static/multibyte.html

Definition

Collation

- sort sequence
- configuration which guideline is used for sorting
- UPPER(), LOWER()
- LIKE

DIN 5007-1, “Duden”

ä is equivalent to a
ö is equivalent to o
ü is equivalent to u
ß is equivalent to s

DIN 5007-2, “phone book”

ä is equivalent to ae
ö is equivalent to oe
ü is equivalent to ue
ß is equivalent to ss

DIN 5007-2, Austria

ä after az
ö after oz
ü after uz
ß is equivalent to ss

DIN 5007-2, Sweden, Finl.

å after z
ä after å
ö after ä
ü is equivalent to y

DIN 5007-2, British

ä after a
ö after o
ü after u
ß after s
Mc is treated as Mac

Example for capitalisation

a:A, b:B, c:C, ä:Ä, ö:Ö, ü:Ü, ß:SZ, å:Å, ø:Ø

Collation

What is important?

- The encoding type has to match the collation type
  - There are no rules in an ISO collation for UTF-8
- You are able to choose the collation type for your system when you are making the initdb:
  - **$ initdb –lc_collate=de_DE**
- Usually initdb will get the collation type from the locale
- Changing the collation type after initdb is not possible

Definition

Locale

collection of political, cultural or language specific computerised rules

Currency sign

€ or EUR
$ or USD
¥ or JPY
£ or MLT
£ or GBP
元 or HKD
...

Capitalisation rules

Sheet size

DIN-A4
LETTER A
...

Sorting rules

Numbers

1618.03
1618,03
1.618,03
1,618.03
1 618,03
1'618.03
1'618,03
...

System messages

No space left on device
Auf dem Gerät ist kein Speicherplatz mehr verfügbar
Aucun espace disponible sur le périphérique
Geen ruimte meer over op apparaat
Spazio insufficiente sul dispositivo
Inget utrymme kvar på enheten
Ikke mere plads på enheden
Laitteella ei ole tilaa jäljellä
No queda espacio libre en el dispositivo
...

Date

2008-02-24
24.02.2008
02/24/2008
2008/02/24
24. Feb. 2008
Feb, 24th 2008
...

Locale

PostgreSQL

How to figure out the locale

- Unix:
  - **$ locale**
  - Which locales are possible on the system:
    - **$ locale -a**
    - Examples:
      - C/POSIX means no locale
      - de_DE.UTF-8
      - de_DE.ISO8859-15
      - en_EN.UTF-8
      - tr_TR.ISO8859-9
- Windows:
  - System language setting

Locale

Categories

- lc_ctype
  - classification of signs
    - What is a letter?
- lc_collate
  - sort sequence rules
  - capitalisation rules
- lc_messages
  - language of the system messages
- lc_numeric
  - number format (i.e. to_char)
- lc_monetary
  - currency sign (i.e. to_char)
- lc_time
  - date format (not used at the moment)

Locale

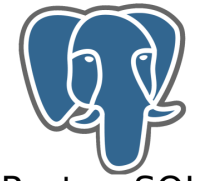
Be careful

Automatically, the system gets all values
from the locale of the user
who builds the cluster (made the initdb).
Usually, this is the user: postgres.

After initialising you can only change:

lc_monetary, lc_messages, lc_numeric

You can change them by
editing postgresql.conf or using SET



Locale

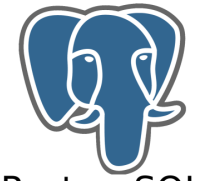
initdb

Before making initdb you should take care of the locale of your corresponding user.

You can add the locale or the single values to initdb:

- $ initdb –locale=utf8
- $ initdb --lc_collate=de_DE --lc_messages=en_US ...

Encoding Server



- Management of data storage on the server (on the disk)
- Default is defined by initdb
- Default set up can be seen by using \l in psql
  - It is the encoding that is listed for the databases:
    - template0 and template1
- Encoding definition (i.e. LATIN9) for a new database:
  - **$ createdb -E LATIN9 dbname**
  - **CREATE DATABASE dbname ENCODING 'LATIN9';**
- Changing database encoding later is impossible.

Encoding

Client

- Defines the *interpretation* of the data that are sent/received from the client
- The actual binary data are defined by the client software
  - i.e. psql, PGAdminIII, own software
- The client software has to inform the server
  - about the encoding of the sent data
  - about the encoding that received data should have
- Changing client encoding is possible
- The client encoding has to fit to the environment

Encoding

Client encoding definition

- Default: server encoding
- Shell: $ export PGCLIENTENCODING=UTF8
- psql: \encoding UTF8
- libpq: PQsetClientEncoding()
- PHP: pg_set_client_encoding()
- JDBC: automatic (always UTF-8)
- and similar more ...

Encoding

Automatic conversion

PostgreSQL

- During transfer the data will be converted from client encoding to server encoding and vice versa.
- This is automatic and transparent if client and server encoding match.

Encoding

Client encoding identification

- psql
  - \encoding
- Console
  - $ locale charmap
- Java/JDBC software
  - Doesn't matter/automatic
- Web software (PHP, Perl, ...)
  - Form data encoding will be negotiated between browser and web server
  - Web server encoding is the database client encoding
- Other development environments
  - Should be documented

Encoding Mismatch

- ISO encoding always use 1 byte for characters
- UTF8 encoding use 1-4 byte for characters
- One of the famous mistakes occurs during INSERT/UPDATE
- The function length() displays the byte length of the text
- The other famous mistake is during SELECT:
  - You will recognise this because of weird outputs:
    - Examples (ISO/UTF8 mismatch):
      - ö => Ã¶ or üß => ÃŒÃ
      - Grüße => Gr or Café => Caf
    - Output like:
      - Grüße => Gre

      usually is a mismatch between ASCII and something else.

Mismatch

Stored data example

```
Terminal encoding: UTF8
$ createdb -E LATIN9 dbname
dbname=# \encoding => LATIN9
dbname=# create table t(id serial, txt text);
dbname=# insert into t(txt) values ('Café'),('Grüße'),('Bär');
dbname=# select length(txt) from t; => 5, 7 and 4
```

- Because of LATIN9 the byte length should be: 4, 5 and 3
  - Data are stored wrong in the database
  - Reason: wrong environment (terminal) encoding during insert
- Repairing this needs a huge effort.
  - i.e. dump => recode => restore
- Solution that this won't happens:
  - Take care of environment and client encoding
    - Switch environment (i.e. terminal) encoding to ISO or
    - Switch client encoding to UTF8 (i.e. \encoding UTF8)

Mismatch

Error message example

```
Default database settings: UTF8
Terminal: ISO-8859-15
$ createdb dbname
dbname=# \encoding => UTF8
dbname=# create table t(id serial, txt text);
dbname=# insert into t(txt) values ('Café');
ERROR:  invalid byte sequence for encoding "UTF8": 0xe92729
```

- Reason: environment and client encoding don't match
- Solution that this won't happens:
  - Take care of environment and client encoding
    - Switch environment (i.e. terminal) encoding to UTF8 or
    - Switch client encoding to LATIN9 (i.e. \encoding LATIN9)

Mismatch

Output example

```
Database: UTF8
Terminal: ISO-8859-15
dbname=# \encoding => UTF8
dbname=# select txt from t;
-------
CafÃ©
GrÃŒÃÃe
BÃ€r
```

- Reason: environment and client encoding don't match
- Solution that this won't happens:
  - Take care of environment and client encoding
    - Switch environment (i.e. terminal) encoding to UTF8 or
    - Switch client encoding to LATIN9 (i.e. \encoding LATIN9)

Mismatch

Output example

```
Database: LATIN9
Terminal: UTF8
dbname=# \encoding => LATIN9
dbname=# select txt from t;
-------
 Caf
 Gr
 B
```

- Reason: environment and client encoding don't match
- Solution that this won't happens:
  - Take care of environment and client encoding
    - Switch environment (i.e. terminal) encoding to ISO or
    - Switch client encoding to UTF8 (i.e. \encoding UTF8)

Recommendation

Which encoding?

- Always recommended: UTF8
  - Locale: i.e. de_DE.UTF-8 or fr_FR.UTF-8
  - Server encoding: UTF8
  - Caution! No Windows UTF8 support before PostgreSQL 8.1
- Also recommended: LATIN9/ISO-8859-15 (if UTF8 occurs trouble)
  - Locale: i.e. de_DE.ISO8859-15 or fr_FR.ISO8859-15
  - Server encoding: LATIN9
- Be careful with SQL_ASCII
  - It is advised not to use it
- Asian encoding
  - Ask a specialist or
  - look at the documentation
- Recommendation for special languages: MULE_INTERNAL

Summary

Dependency Encoding/Locale

- Sort sequence is defined by locale
- libc (OS libraries) requires a special encoding for sorting
  - This is defined by locale
- Server encoding and locale settings has to match
  - If not => byte chaos during sorting
- Server encoding and lc_collate has to match
  - Server encoding should be the same for all databases

Summary

The right way

- Think about encoding and locale before initialise PostgreSQL
- Elect the locale for initdb
  - which kind of sort sequence is necessary for my software?
- Automatically intidb will elect the matching server encoding
- Don't use database specific encodings
  - Always convert client encoding or
  - make sure that client and server environment are equal
- Make sure that environment and client encoding are equal

Summary

Summary

- Specify locale for the initdb process
- Server encoding is managing the data storage
- Client encoding and environment encoding has to match

Encoding Issues

Closing Words

Thank you Peter for once let me in on this topic

Thank you Wikipedia for existing

Thank you PostgreSQL project for the excellent documentation

Thanks for listening