

# *Properly use of charset and collation*

Dipl.-Inf. Susanne Ebrecht

MySQL Support Engineer  
**Sun Microsystems GmbH**

**MySQL University 2009**

# Character Set

A collection of signs ...

ð\*#⊗Ⓜ}!%&?≠♣☺→.:.⊥≡  
 {}-€∂ ← ↓ →↓ ϕ ~+

1-9

1	2	3
4	5	6
7	8	9

The German alphabet

AaÄäBbCcDdEeFfGgHhIijJkKlIMmNnO  
 oÖöPpQqRrSsßTtUuÜüVvWwXxYyZz

## UNICODE

The Greek alphabet

ΑαΒβΓγΔδΕεΖζΗηΘθΙιΚκΛλΜμΝν  
 ΞξΟοΠπΡρΣσςΤτΥυΦφΧχΨψΩω

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	ı	ç	£	€	¥	Š	š	š	©	ª	«	¬	SHY	®	-
°	±	²	³	Ž	μ	¶	·	ž	¹	º	»	Œ	œ	ÿ	ı
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	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

# Encoding

Implementation of abstract signs into bits and bytes

UTF-32

KOI8-R

UTF-8

KOI8-U

UTF-7

ISO-8859-15

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

ASCII

EUC-JP

UTF-16

BIG5

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

# Collation

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

# Collation

DIN 5007-1, "Duden"

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

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, Δ:δ, Ω:ω

# Collation

- ★ Character set and collation have to match to each other
- ★ There are none rules in i.e. a latin collation for utf8
- ★ MySQL character sets have default collations which will be taken when none collation is given.
- ★ **SHOW CHARSET** will also display the default collations

# Collation Examples

```
SHOW VARIABLES LIKE 'character_set_database';  
=> LATIN1
```

```
SHOW CHARSET LIKE 'utf8';  
=> Default collation: utf8_general_ci
```

```
CREATE TABLE t01(v varchar(100));  
INSERT INTO t01(v) VALUES('a');  
SELECT COLLATION(v) FROM t01;  
=> latin1_swedish_ci
```

```
CREATE TABLE t02(v varchar(100)) CHARSET utf8;  
INSERT INTO t02(v) VALUES('a');  
SELECT COLLATION(v) FROM t02;  
=> utf8_general_ci
```

```
CREATE TABLE t03(v varchar(100)) COLLATE utf8_bin;  
INSERT INTO t03(v) VALUES('a');  
SELECT COLLATION(v) FROM t03;  
=> utf8_bin
```

```
CREATE TABLE t04(v varchar(100)) CHARSET utf8 COLLATE utf8_bin;  
INSERT INTO t04(v) VALUES('a');  
SELECT COLLATION(v) FROM t04;  
=> utf8_bin
```

# Collation Examples

```
CREATE TABLE t05(v varchar(100) CHARSET utf8);  
INSERT INTO t05(v) VALUES('a');  
SELECT COLLATION(v) FROM t05;  
=> utf8_general_ci
```

```
CREATE TABLE t06(v varchar(100) CHARSET utf8) CHARSET utf8;  
INSERT INTO t06(v) VALUES('a');  
SELECT COLLATION(v) FROM t06;  
=> utf8_general_ci
```

```
CREATE TABLE t07(v varchar(100) CHARSET utf8) COLLATE utf8_bin;  
INSERT INTO t07(v) VALUES('a');  
SELECT COLLATION(v) FROM t07;  
=> utf8_general_ci
```

```
CREATE TABLE t08(v varchar(100) CHARSET utf8) CHARSET utf8 COLLATE utf8_bin;  
INSERT INTO t08(v) VALUES('a');  
SELECT COLLATION(v) FROM t08;  
=> utf8_general_ci
```

# Collation Examples

```
CREATE TABLE t09(v varchar(100) COLLATE utf8_bin);  
INSERT INTO t09(v) VALUES('a');  
SELECT COLLATION(v) FROM t09;  
=> utf8_bin
```

```
CREATE TABLE t10(v varchar(100) COLLATE utf8_bin) CHARSET utf8;  
INSERT INTO t10(v) VALUES('a');  
SELECT COLLATION(v) FROM t10;  
=> utf8_bin
```

```
CREATE TABLE t11(v varchar(100) COLLATE utf8_bin) COLLATE utf8_unicode_ci;  
INSERT INTO t11(v) VALUES('a');  
SELECT COLLATION(v) FROM t11;  
=> utf8_bin
```

```
CREATE TABLE t12(v varchar(100) COLLATE utf8_bin) CHARSET utf8 COLLATE utf8_unicode_ci;  
INSERT INTO t12(v) VALUES('a');  
SELECT COLLATION(v) FROM t12;  
=> utf8_bin
```

# Collation Examples

```
CREATE TABLE t13(  
v varchar(100) CHARSET utf8 COLLATE utf8_bin);  
INSERT INTO t13(v) VALUES('a');  
SELECT COLLATION(v) FROM t13;  
=> utf8_bin
```

```
CREATE TABLE t14(  
v varchar(100) CHARSET utf8 COLLATE utf8_bin) CHARSET utf8;  
INSERT INTO t14(v) VALUES('a');  
SELECT COLLATION(v) FROM t14;  
=> utf8_bin
```

```
CREATE TABLE t15(  
v varchar(100) CHARSET utf8 COLLATE utf8_bin) COLLATE utf8_unicode_ci;  
INSERT INTO t15(v) VALUES('a');  
SELECT COLLATION(v) FROM t15;  
=> utf8_bin
```

```
CREATE TABLE t16(  
v varchar(100) CHARSET utf8 COLLATE utf8_bin) CHARSET utf8 COLLATE utf8_unicode_ci;  
INSERT INTO t16(v) VALUES('a');  
SELECT COLLATION(v) FROM t16;  
=> utf8_bin
```

# Environment Encoding

Encoding of the environment

Examples:

- ★ SHELL => locale
- ★ Terminal => given in Settings
- ★ Windows CMD => CHCP
- ★ Browser => given in Settings
- ★ Editor => given in Settings

# File System Encoding

- ★ Encoding of the file system
- ★ Which encoding is used to store files on filesystem
- ★ MySQL variable `character_set_filesystem`
- ★ On Unix => locale setting of the user is taken
- ★ On Windows => Codepage from output of CHCP

# Server Encoding

- ★ The encoding that the server is using internally
- ★ Management of data storage on the server (on the disk)
- ★ MySQL variable `character_set_system`
  - Always `utf8` (not changeable)
  - used for storing identifiers
- ★ MySQL variable `character_set_server` / `collation_server`
  - Default character set of the server
  - Default `character_set_server` is `latin1`
  - Default `collation_server` is `latin1_swedish_ci`

# Server Encoding

## Database/Table/Column

### ★ Database encoding

- MySQL variable `character_set_database` / `collation_database`
- Default value is value of `character_set_server`
- `CREATE DATABASE db CHARSET mychar COLLATE mycoll;`
- Is taken for tables, views, routines, ... as default value

### ★ Table encoding

- `CREATE TABLE t(...) CHARSET mychar COLLATE mycoll;`
- Is taken for columns as default value
- When collation is not given here columns will take the given charset with default collation of the charset (see collation examples)  
When only collation is given here, columns will take matching collation charset and default charset collation

### ★ Column encoding

- `CREATE TABLE t(..., v varchar(100) CHARSET mychar COLLATE mycoll, ...) ...;`
- Is taken for the column in which it is given

# Client Encoding

- ★ 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. CLI, Workbench, 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

# Client Encoding

- ★ MySQL variable `character_set_client`
  - To let the server know which encoding
  - Is used for input data
- ★ MySQL variable `character_set_result`
  - To let the server know which encoding
  - It should use for output data
- ★ MySQL variable `character_set_connection`
  - Used for special connectors like MyODBC/JDBC
  - Should not be changed separately by a user manually
  - Besides when connectors are used it always should have Same value as `character_set_client/character_set_result`
- ★ **SET NAMES** is setting all three variables
- ★ All three variables should have same value  
(besides connectores i.e. MyODBC or JDBC are involved)

# Client Encoding

- ★ CLI: Default latin1
  - ➔ Use **SET NAMES** when your environment is using another encoding
    - On Windows ask **CHCP** in CMD
    - On Unix GUI (i.e. Gnome/KDE) look into setting of your terminal
    - On Unix console ask **locale**
- ★ JDBC: always utf8
- ★ MyODBC: always utf8
- ★ Other connectors: should be documented
- ★ Workbench: always utf8
- ★ Others: should be documented

# Identify Client Encoding

## ★ CLI

→ SHOW VARIABLES LIKE '%char%';

● character\_set\_client, character\_set\_result, character\_set\_connection

## ★ JAVA/JDBC

→ Doesn't matter/automatic

## ★ Workbench

→ Doesn't matter/automatic

## ★ Web software (PHP)

→ Form data encoding will be negotiated between browser and webserver

→ Webserver/Browser encoding is used here and should be set as client encoding

## ★ Other development environments

→ Should be documented

# *Automatic Conversion*

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

# 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 => Ãœ
        - ◆ Grüße => Gr or Café => Caf
      - Output like:
        - ◆ Grüße => Gre
- usually is a mismatch between ASCII and something else.

# Mismatch (Input)

Terminal encoding: UTF8

```
mysql> SHOW VARIABLES LIKE '%char%';
```

=> character\_set\_client/character\_set\_result => latin1

```
mysql> CREATE TABLE t(v varchar(100) CHARSET latin1);
```

```
mysql> INSERT INTO t(v) values ('Café'),('Grüße'),('Bär');
```

```
mysql> SELECT length(v) FROM t; => 5, 7 and 4
```

- ★ Because of LATIN1 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 happen:
  - Take care of environment and client encoding
    - Switch environment (i.e. terminal) encoding to ISO or
    - SET NAMES utf8;

# Mismatch (Input)

Terminal encoding: UTF8

```
mysql> SHOW VARIABLES LIKE '%char%';
```

=> character\_set\_client/character\_set\_result => latin1

```
mysql> CREATE TABLE t(v varchar(100) CHARSET utf8);
```

```
mysql> INSERT INTO t(v) values ('Café'),('Grüße'),('Bär');
```

```
mysql> SELECT length(v) FROM t; => 7, 11 and 6
```

- ★ Because of UTF8 the byte length should be: 5, 7 and 4
  - 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 happen:
  - Take care of environment and client encoding
    - Switch environment (i.e. terminal) encoding to ISO or
    - SET NAMES utf8;

# Mismatch (Output)

```
Terminal encoding: UTF8
mysql> SHOW VARIABLES LIKE '%char%';
=> character_set_client/character_set_result => latin1
mysql> SHOW CREATE TABLE t;
=> Column should have utf8
mysql> SELECT v FROM t;
Caf?
Gr??e
B?r
```

- ★ Reason: environment and client encoding don't match
- ★ Solution that this won't happen:
  - Take care of environment and client encoding
    - Switch environment (i.e. terminal) encoding to ISO or
    - SET NAMES utf8;

# Mismatch (Output)

```
Terminal encoding: CP850
mysql> SHOW VARIABLES LIKE '%char%';
=> character_set_client/character_set_result => utf8
mysql> SHOW CREATE TABLE t;
=> Column should have utf8
mysql> SELECT v FROM t;
CafÃ©
GrÃ¼e e
BÃ¼r
```

- ★ Reason: environment and client encoding don't match
- ★ Solution that this won't happen:
  - **Take care of environment and client encoding**
    - Switch environment (i.e. terminal) encoding to utf8 or
    - **SET NAMES cp850;**

# Summary

- ★ sort sequence is given with collation
- ★ Server encoding is managing data storage
- ★ Client and environment encoding has to match
  - If not => byte chaos
- ★ Character set and collation have to match
- ★ **Think about** encoding **before** you are **doing** something at the database
  - If not => chaos, not repairable data, wrong output

# *Closing Words*

Thanks for listening