# Character set / Collation basics

Justin Swanhart

# What are character sets



- Character sets inform the database about the storage characteristics of strings
  - A string is a series of bytes
  - The character set maps sequences of bytes to particular characters

# Single byte character sets



- Many character sets use only one byte per character
- This means a string of 30 characters uses 30 bytes
- Some single byte character sets include
  - latin1
  - ascii
  - hebrew

# Multi-byte character sets



- Single byte character sets are limited to 256 maximum characters
- Many languages, especially pictographic languages require many more characters
- Emojii are now common and represent a multitude of characters

# Multi-byte means some problems



- Storage of strings when sorting and grouping uses the maximum number of bytes for a string
- For example, in UTF8, the string 'aaa' requires 9 bytes when sorting and grouping

# Common multi-byte characters sets



- UTF8 - 3 byte UTF-8, most languages
- UTF8MB4 – 4 byte UTF-8 ( adds CJK+emoji)
- UJIS – 3 byte Japanese
- big5 – traditional Chiense

# Picking a character set



- Choose the character set based on the characters you have to store
- utf8mb4 is the best for compatibility with all languages
  - But it uses 4 bytes per character
  - Avoid sorting large amounts of utf8mb4 data if possible
- Use smaller character sets for data which must be sorted on (like category names, or product names in many cases)

# Specifying the character set



- Database/Schema level
- Table level
- Column level
- String literals

# Database level



- If you do not specify a character set at the table level at creation time, or for a particular column, then the database level collation is used

# Table level and column level



```
mysql> create table table_level(c1 varchar(10), c2 varchar(10)
character set latin1) character set utf8mb4;
Query OK, 0 rows affected (0.01 sec)

mysql> show create table table_level\G
*************************** 1. row ***************************
       Table: table_level
Create Table: CREATE TABLE `table_level` (
  `c1` varchar(10) DEFAULT NULL,
  `c2` varchar(10) CHARACTER SET latin1 DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4
1 row in set (0.00 sec)
```

# What happens if I pick the wrong character set?



- Using STRICT_TRANS_TABLES you can not insert bad data directly into the database

```
mysql> create table bad (c1  char(10) character
set latin1, c2 char(10) character set utf8mb4);
Query OK, 0 rows affected (0.01 sec)

mysql> insert into bad values ('鮟','鮟');
ERROR 1366 (HY000): Incorrect string value:
'\xF0\xA9\xB6\x98' for column 'c1' at row 1
```

# But if application does not properly encode data..



```
mysql> set names utf8mb4;
Query OK, 0 rows affected (0.00
sec)

mysql> insert into bad values
(binary '鮟','鮟');
Query OK, 1 row affected (0.00 sec)
```

# GIGO



```
mysql> select * from bad;
+-----------+-------+
| c1        | c2    |
+-----------+-------+
| ð©¶˜      | 魤    |
+-----------+-------+
1 row in set (0.00 sec)
```

# Determining available character sets



```
mysql> show character set where maxlen =4;
+---------+------------------+--------------------+--------+
| Charset | Description      | Default collation  | Maxlen |
+---------+------------------+--------------------+--------+
| utf8mb4 | UTF-8 Unicode    | utf8mb4_general_ci |      4 |
| utf16   | UTF-16 Unicode   | utf16_general_ci   |      4 |
| utf16le | UTF-16LE Unicode | utf16le_general_ci |      4 |
| utf32   | UTF-32 Unicode   | utf32_general_ci   |      4 |
+---------+------------------+--------------------+--------+
4 rows in set (0.00 sec)
```

# Use STRICT MODE



- Not only because it prevents bad text from being entered
- It prevents invalid dates and numbers from being inserted
- Does not allow insertion of NULL values into not-null columns

# Collations



- Each character set has multiple collations available
- Collations
  - Specicy string sort order
  - Specify string comparisons characteristics

# Sort order is affected by collation



```
mysql> select c1 from txt order by c1 collate latin1_swedish_ci;
+------+
| c1   |
+------+
| a    |
| A1   |
| a1   |
| b2   |
| B2   |
| z    |
+------+
6 rows in set (0.00 sec)

mysql> select c1 from txt order by c1 collate latin1_general_cs;
+------+
| c1   |
+------+
| A1   |
| a    |
| a1   |
| B2   |
| b2   |
| z    |
+------+
6 rows in set (0.00 sec)
```

# String comparisons are too



```
mysql> alter table txt modify c1 varchar(100) character set latin1 collate latin1_general_cs;
Query OK, 6 rows affected (0.04 sec)
Records: 6  Duplicates: 0  Warnings: 0

mysql> select * from txt where c1 = 'a1';
+------+
| c1   |
+------+
| a1   |
+------+
1 row in set (0.00 sec)

mysql> alter table txt modify c1 varchar(100) character set latin1 collate latin1_general_ci;
Query OK, 6 rows affected (0.03 sec)
Records: 6  Duplicates: 0  Warnings: 0

mysql> select * from txt where c1 = 'a1';
+------+
| c1   |
+------+
| A1   |
| a1   |
+------+
2 rows in set (0.00 sec)
```

# Character set literals



- select _utf8'This is a UTF-8 string';

# Selecting a character set at runtime



- For proper display of strings
- And for proper encoding of literals
- Use SET NAMES
  - SET NAMES 'UTF8'
- Possibly put this in your *init_sql* setting