k-bushi

- Top
- Archives
- Tags
- Categories
- About

MySQLでchar/varchar/text型のカラムで大文字/小文字を区別できるようにする

by k-bushi
2023-03-05
技術

はじめに

MySQLを使用して文字列の検索を行ったときに、大文字でも小文字でも検索できるということに今更ながら気づいた。
そのため、どういう仕組み/どういう理由で大文字/小文字を比較しているのかが気になり調査をしてみた。
まずは、大文字/小文字を区別するようにするためにはどうするのかを記載する。

環境

```
Windows 10 Pro (操作PC)
Docker Desktop 4.17.0 (99724)
MySQL 8.0.32
```

準備

今回検証するにあたって、`Docker` を使用し、手で確かめられるようにした。
下記リポジトリをクローンし、 `Docker` のビルドを行うことで一緒に確かめられるので是非試してほしい。
https://github.com/katsuobushiFPGA/docker-mysql-binary-practice

特定の型のカラムで大文字/小文字を区別できるようにする

`char`, `varchar`, `text` 型の文字列検索では大文字/小文字は区別しない。
そのため、以下に解決方法できる方法を記載する。

SQLクエリにBINARY演算子を利用する

例えば、以下のようなテーブル構造とレコードがあったとする

テーブル構造

```
mysql> desc test_case_sensitive;
+--------------+--------------+------+-----+---------+-------+
| Field        | Type         | Null | Key | Default | Extra |
+--------------+--------------+------+-----+---------+-------+
| test_varchar | varchar(100) | YES  |     | NULL    |       |
| test_char    | char(100)    | YES  |     | NULL    |       |
| test_text    | text         | YES  |     | NULL    |       |
+--------------+--------------+------+-----+---------+-------+
```

レコード

```
INSERT INTO `test_case_sensitive` VALUES (
    'aaaAaaAaAaaAAaA',
    'aaaAaaAaAaaAAaA',
    'aaaAaaAaAaaAAaA'
);
```

```
mysql> select * from test_case_sensitive;
+-----------------+-----------------+-----------------+
| test_varchar    | test_char       | test_text       |
+-----------------+-----------------+-----------------+
| aaaAaaAaAaaAAaA | aaaAaaAaAaaAAaA | aaaAaaAaAaaAAaA |
```

```
+----------------+----------------+----------------+
```

このときに、いくつかの SELECT 文を実行し検索できるかを試してみる。

```
mysql> SELECT * FROM test_case_sensitive WHERE test_varchar = 'aaaaaaaaaaaaaa';
+----------------+----------------+----------------+
| test_varchar   | test_char      | test_text      |
+----------------+----------------+----------------+
| aaaAaaAaAaaAAaA | aaaAaaAaAaaAAaA | aaaAaaAaAaaAAaA |
+----------------+----------------+----------------+
1 row in set (0.00 sec)

mysql> SELECT * FROM test_case_sensitive WHERE test_char = 'aaaaaaaaaaaaaa';
+----------------+----------------+----------------+
| test_varchar   | test_char      | test_text      |
+----------------+----------------+----------------+
| aaaAaaAaAaaAAaA | aaaAaaAaAaaAAaA | aaaAaaAaAaaAAaA |
+----------------+----------------+----------------+
1 row in set (0.00 sec)

mysql> SELECT * FROM test_case_sensitive WHERE test_text = 'aaaaaaaaaaaaaa';
+----------------+----------------+----------------+
| test_varchar   | test_char      | test_text      |
+----------------+----------------+----------------+
| aaaAaaAaAaaAAaA | aaaAaaAaAaaAAaA | aaaAaaAaAaaAAaA |
+----------------+----------------+----------------+
1 row in set (0.00 sec)


mysql> SELECT * FROM test_case_sensitive WHERE test_varchar = 'AAAAAAAAAAAAAA';
+----------------+----------------+----------------+
| test_varchar   | test_char      | test_text      |
+----------------+----------------+----------------+
| aaaAaaAaAaaAAaA | aaaAaaAaAaaAAaA | aaaAaaAaAaaAAaA |
+----------------+----------------+----------------+
1 row in set (0.00 sec)

mysql> SELECT * FROM test_case_sensitive WHERE test_char = 'AAAAAAAAAAAAAA';
+----------------+----------------+----------------+
| test_varchar   | test_char      | test_text      |
+----------------+----------------+----------------+
| aaaAaaAaAaaAAaA | aaaAaaAaAaaAAaA | aaaAaaAaAaaAAaA |
+----------------+----------------+----------------+
1 row in set (0.01 sec)

mysql> SELECT * FROM test_case_sensitive WHERE test_text = 'AAAAAAAAAAAAAA';
+----------------+----------------+----------------+
| test_varchar   | test_char      | test_text      |
+----------------+----------------+----------------+
| aaaAaaAaAaaAAaA | aaaAaaAaAaaAAaA | aaaAaaAaAaaAAaA |
+----------------+----------------+----------------+
1 row in set (0.00 sec)
```

上記のように、文字列の大文字/小文字を区別せずに検索していることがわかる。

ここで、 BINARY 演算子をつけると結果が変わる。
下記のように、 SELECT 文を流してみる。

```
mysql> SELECT * FROM test_case_sensitive WHERE BINARY test_varchar = 'aaaaaaaaaaaaaa';
Empty set, 1 warning (0.00 sec)

mysql> SELECT * FROM test_case_sensitive WHERE BINARY test_char = 'aaaaaaaaaaaaaa';
Empty set, 1 warning (0.00 sec)

mysql> SELECT * FROM test_case_sensitive WHERE BINARY test_text = 'aaaaaaaaaaaaaa';
Empty set, 1 warning (0.00 sec)

mysql> SELECT * FROM test_case_sensitive WHERE BINARY test_varchar = 'AAAAAAAAAAAAAA';
Empty set, 1 warning (0.00 sec)

mysql> SELECT * FROM test_case_sensitive WHERE BINARY test_char = 'AAAAAAAAAAAAAA';
Empty set, 1 warning (0.00 sec)

mysql> SELECT * FROM test_case_sensitive WHERE BINARY test_text = 'AAAAAAAAAAAAAA';
Empty set, 1 warning (0.00 sec)

mysql> SELECT * FROM test_case_sensitive WHERE BINARY test_varchar = 'aaaAaaAaAaaAAaA';
+----------------+----------------+----------------+
| test_varchar   | test_char      | test_text      |
+----------------+----------------+----------------+
| aaaAaaAaAaaAAaA | aaaAaaAaAaaAAaA | aaaAaaAaAaaAAaA |
+----------------+----------------+----------------+
1 row in set, 1 warning (0.00 sec)

mysql> SELECT * FROM test_case_sensitive WHERE BINARY test_char = 'aaaAaaAaAaaAAaA';
+----------------+----------------+----------------+
| test_varchar   | test_char      | test_text      |
+----------------+----------------+----------------+
| aaaAaaAaAaaAAaA | aaaAaaAaAaaAAaA | aaaAaaAaAaaAAaA |
+----------------+----------------+----------------+
1 row in set, 1 warning (0.00 sec)

mysql> SELECT * FROM test_case_sensitive WHERE BINARY test_text = 'aaaAaaAaAaaAAaA';
+----------------+----------------+----------------+
| test_varchar   | test_char      | test_text      |
+----------------+----------------+----------------+
| aaaAaaAaAaaAAaA | aaaAaaAaAaaAAaA | aaaAaaAaAaaAAaA |
+----------------+----------------+----------------+
1 row in set, 1 warning (0.00 sec)
```

クエリのWHERE句にBINARY 演算子をつけることで比較できることがわかる。

カラムにBINARY属性をつける

```
ALTER TABLE `test_case_sensitive` MODIFY COLUMN `test_varchar` varchar(100) binary DEFAULT NULL;
ALTER TABLE `test_case_sensitive` MODIFY COLUMN `test_char` char(100) binary DEFAULT NULL;
ALTER TABLE `test_case_sensitive` MODIFY COLUMN `test_text` text binary DEFAULT NULL;

ALTER TABLE `test_case_sensitive` MODIFY COLUMN `test_text` text binary DEFAULT NULL;Query OK, 0 rows affected, 1 warning (0.17 sec)
Records: 0  Duplicates: 0  Warnings: 1

mysql> ALTER TABLE `test_case_sensitive` MODIFY COLUMN `test_varchar` char(100) binary DEFAULT NULL;
Query OK, 1 row affected, 1 warning (2.91 sec)
Records: 1  Duplicates: 0  Warnings: 1

mysql> ALTER TABLE `test_case_sensitive` MODIFY COLUMN `test_text` text binary DEFAULT NULL;
Query OK, 0 rows affected, 1 warning (0.62 sec)
Records: 0  Duplicates: 0  Warnings: 1

/*
戻すとき
ALTER TABLE `test_case_sensitive` MODIFY COLUMN `test_varchar` varchar(100) DEFAULT NULL;
ALTER TABLE `test_case_sensitive` MODIFY COLUMN `test_char` char(100) DEFAULT NULL;
ALTER TABLE `test_case_sensitive` MODIFY COLUMN `test_text` text DEFAULT NULL;
*/
```

この状態で再度チェックをしてみる。

```
mysql> SELECT * FROM test_case_sensitive WHERE test_varchar = 'aaaaaaaaaaaaaaa';
Empty set (0.00 sec)

mysql> SELECT * FROM test_case_sensitive WHERE test_char = 'aaaaaaaaaaaaaaa';
Empty set (0.00 sec)

mysql> SELECT * FROM test_case_sensitive WHERE test_text = 'aaaaaaaaaaaaaaa';
Empty set (0.00 sec)

mysql>
mysql> SELECT * FROM test_case_sensitive WHERE test_varchar = 'AAAAAAAAAAAAAAA';
Empty set (0.00 sec)

mysql> SELECT * FROM test_case_sensitive WHERE test_char = 'AAAAAAAAAAAAAAA';
Empty set (0.00 sec)

mysql> SELECT * FROM test_case_sensitive WHERE test_text = 'AAAAAAAAAAAAAAA';
Empty set (0.00 sec)

mysql> SELECT * FROM test_case_sensitive WHERE test_varchar = 'aaaAaaAaAaaAAaA';
+----------------+----------------+----------------+
| test_varchar   | test_char      | test_text      |
+----------------+----------------+----------------+
| aaaAaaAaAaaAAaA | aaaAaaAaAaaAAaA | aaaAaaAaAaaAAaA |
+----------------+----------------+----------------+
1 row in set (0.00 sec)

mysql> SELECT * FROM test_case_sensitive WHERE test_char = 'aaaAaaAaAaaAAaA';
+----------------+----------------+----------------+
| test_varchar   | test_char      | test_text      |
+----------------+----------------+----------------+
| aaaAaaAaAaaAAaA | aaaAaaAaAaaAAaA | aaaAaaAaAaaAAaA |
+----------------+----------------+----------------+
1 row in set (0.00 sec)

mysql> SELECT * FROM test_case_sensitive WHERE test_text = 'aaaAaaAaAaaAAaA';
+----------------+----------------+----------------+
| test_varchar   | test_char      | test_text      |
+----------------+----------------+----------------+
| aaaAaaAaAaaAAaA | aaaAaaAaAaaAAaA | aaaAaaAaAaaAAaA |
+----------------+----------------+----------------+
1 row in set (0.00 sec)
```

今度は BINARY 演算子を使用せずとも比較できていることがわかる。

どうしてこのようになっているのか？

大文字小文字の区別をしない理由についてはSQL標準で決まっており、それに準拠しているからというのが理由となる。
※SQL標準についての中身は確認できなかった。(規格のドキュメントを購入しないといけなさそう)

おまけ

今回使用したときの、各変数の文字コードセット↓

```
mysql> show variables like 'character¥_set¥_%';
+--------------------------+---------+
| Variable_name            | Value   |
+--------------------------+---------+
| character_set_client     | latin1  |
| character_set_connection | latin1  |
| character_set_database   | utf8mb4 |
| character_set_filesystem | binary  |
| character_set_results    | latin1  |
| character_set_server     | utf8mb4 |
| character_set_system     | utf8mb3 |
+--------------------------+---------+
7 rows in set (0.01 sec)
```

※ latin1 は日本語が扱えないので、 character_set_client は utf8mb4 で良いかなと思いました。
今回は日本語を使用しないのでデフォルトの設定をそのまま使っています。

テーブルの確認

```
mysql> show table status from test;
+--------------------+--------+---------+------------+------+----------------+-------------+-----------------+--------------+-----------+----------------
```

```
| Name                | Engine | Version | Row_format | Rows | Avg_row_length | Data_length | Max_data_length | Index_length | Data_free | Auto_increr
  | Checksum | Create_options | Comment |
+---------------------+--------+---------+------------+------+----------------+-------------+-----------------+--------------+-----------+------------
| test_case_sensitive | InnoDB |      10 | Dynamic    |    0 |              0 |       16384 |               0 |            0 |         0 |           |
+---------------------+--------+---------+------------+------+----------------+-------------+-----------------+--------------+-----------+------------
1 row in set (0.00 sec)
```

テーブルの確認 BINARY付与前

```
mysql> show create table test_case_sensitive;
+---------------------+-------------------------------------------------------------------------------------------------------------------------------
| Table               | Create Table
      |
+---------------------+-------------------------------------------------------------------------------------------------------------------------------
| test_case_sensitive | CREATE TABLE `test_case_sensitive` (
  `test_varchar` varchar(100) DEFAULT NULL,
  `test_char` char(100) DEFAULT NULL,
  `test_text` text
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci |
+---------------------+-------------------------------------------------------------------------------------------------------------------------------
1 row in set (0.00 sec)
```

テーブルの確認 BINARY付与後

```
mysql> show create table test_case_sensitive;
+---------------------+-------------------------------------------------------------------------------------------------------------------------------
| Table               | Create Table
                                                                                                                                                   |
+---------------------+-------------------------------------------------------------------------------------------------------------------------------
| test_case_sensitive | CREATE TABLE `test_case_sensitive` (
  `test_varchar` varchar(100) CHARACTER SET utf8mb4 COLLATE utf8mb4_bin DEFAULT NULL,
  `test_char` char(100) CHARACTER SET utf8mb4 COLLATE utf8mb4_bin DEFAULT NULL,
  `test_text` text CHARACTER SET utf8mb4 COLLATE utf8mb4_bin
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci |
+---------------------+-------------------------------------------------------------------------------------------------------------------------------
1 row in set (0.01 sec)
```

各カラムに `_bin` の `COLLATE` (照合順序)ができている。さて、これについて参考にした MySQLの公式ページを見ると

> たとえば、両方とも utf8mb4 文字セットを持つカラムと文字列を比較する場合は、COLLATE 演算子を使用して、いずれかのオペランドに utf8mb4_0900_as_cs 照合順序または utf8mb4_bin 照合順序を設定できます:

- 参考: https://dev.mysql.com/doc/refman/8.0/ja/case-sensitivity.html

ということであれば、 `COLLATE` に `_cs` 付きの文字コードをセットすれば比較できるということか。
照合順序は、`_ci`, `_cs`, `_bin` といくつか種類があり、それぞれ 大文字小文字の区別が `できない`, `できる`, `できる`。

- `ci`: `case_insensitive` の略
- `cs`: `case_sensitive` の略
- `bin`: `binary` バイナリーの略

※ここでは、大文字/小文字にフォーカスしているので、上記しか記載していないが他にもっと区別されるものはある。

`_cs` 付きの `COLLATE` でやってみる。

COLLATE を `_cs` にして再度やってみる。

まずは、指定できる `COLLATE` を見てみる。

```
mysql> show collation like 'utf8mb4%';
+----------------------------+---------+-----+---------+----------+---------+---------------+
| Collation                  | Charset | Id  | Default | Compiled | Sortlen | Pad_attribute |
+----------------------------+---------+-----+---------+----------+---------+---------------+
| utf8mb4_0900_ai_ci         | utf8mb4 | 255 | Yes     | Yes      |       0 | NO PAD        |
| utf8mb4_0900_as_ci         | utf8mb4 | 305 |         | Yes      |       0 | NO PAD        |
| utf8mb4_0900_as_cs         | utf8mb4 | 278 |         | Yes      |       0 | NO PAD        |
| utf8mb4_0900_bin           | utf8mb4 | 309 |         | Yes      |       1 | NO PAD        |
| utf8mb4_bg_0900_ai_ci      | utf8mb4 | 318 |         | Yes      |       0 | NO PAD        |
| utf8mb4_bg_0900_as_cs      | utf8mb4 | 319 |         | Yes      |       0 | NO PAD        |
| utf8mb4_bin                | utf8mb4 |  46 |         | Yes      |       1 | PAD SPACE     |
| utf8mb4_bs_0900_ai_ci      | utf8mb4 | 316 |         | Yes      |       0 | NO PAD        |
| utf8mb4_bs_0900_as_cs      | utf8mb4 | 317 |         | Yes      |       0 | NO PAD        |
| utf8mb4_croatian_ci        | utf8mb4 | 245 |         | Yes      |       8 | PAD SPACE     |
| utf8mb4_cs_0900_ai_ci      | utf8mb4 | 266 |         | Yes      |       0 | NO PAD        |
| utf8mb4_cs_0900_as_cs      | utf8mb4 | 289 |         | Yes      |       0 | NO PAD        |
| utf8mb4_czech_ci           | utf8mb4 | 234 |         | Yes      |       8 | PAD SPACE     |
| utf8mb4_danish_ci          | utf8mb4 | 235 |         | Yes      |       8 | PAD SPACE     |
| utf8mb4_da_0900_ai_ci      | utf8mb4 | 267 |         | Yes      |       0 | NO PAD        |
| utf8mb4_da_0900_as_cs      | utf8mb4 | 290 |         | Yes      |       0 | NO PAD        |
| utf8mb4_de_pb_0900_ai_ci   | utf8mb4 | 256 |         | Yes      |       0 | NO PAD        |
| utf8mb4_de_pb_0900_as_cs   | utf8mb4 | 279 |         | Yes      |       0 | NO PAD        |
| utf8mb4_eo_0900_ai_ci      | utf8mb4 | 273 |         | Yes      |       0 | NO PAD        |
| utf8mb4_eo_0900_as_cs      | utf8mb4 | 296 |         | Yes      |       0 | NO PAD        |
| utf8mb4_esperanto_ci       | utf8mb4 | 241 |         | Yes      |       8 | PAD SPACE     |
| utf8mb4_estonian_ci        | utf8mb4 | 230 |         | Yes      |       8 | PAD SPACE     |
| utf8mb4_es_0900_ai_ci      | utf8mb4 | 263 |         | Yes      |       0 | NO PAD        |
| utf8mb4_es_0900_as_cs      | utf8mb4 | 286 |         | Yes      |       0 | NO PAD        |
| utf8mb4_es_trad_0900_ai_ci | utf8mb4 | 270 |         | Yes      |       0 | NO PAD        |
| utf8mb4_es_trad_0900_as_cs | utf8mb4 | 293 |         | Yes      |       0 | NO PAD        |
| utf8mb4_et_0900_ai_ci      | utf8mb4 | 262 |         | Yes      |       0 | NO PAD        |
| utf8mb4_et_0900_as_cs      | utf8mb4 | 285 |         | Yes      |       0 | NO PAD        |
| utf8mb4_general_ci         | utf8mb4 |  45 |         | Yes      |       1 | PAD SPACE     |
| utf8mb4_german2_ci         | utf8mb4 | 244 |         | Yes      |       8 | PAD SPACE     |
| utf8mb4_gl_0900_ai_ci      | utf8mb4 | 320 |         | Yes      |       0 | NO PAD        |
| utf8mb4_gl_0900_as_cs      | utf8mb4 | 321 |         | Yes      |       0 | NO PAD        |
| utf8mb4_hr_0900_ai_ci      | utf8mb4 | 275 |         | Yes      |       0 | NO PAD        |
| utf8mb4_hr_0900_as_cs      | utf8mb4 | 298 |         | Yes      |       0 | NO PAD        |
```

```
| utf8mb4_hungarian_ci       | utf8mb4 | 242 |         | Yes      |       8 | PAD SPACE     |
| utf8mb4_hu_0900_ai_ci      | utf8mb4 | 274 |         | Yes      |       0 | NO PAD        |
| utf8mb4_hu_0900_as_cs      | utf8mb4 | 297 |         | Yes      |       0 | NO PAD        |
| utf8mb4_icelandic_ci       | utf8mb4 | 225 |         | Yes      |       8 | PAD SPACE     |
| utf8mb4_is_0900_ai_ci      | utf8mb4 | 257 |         | Yes      |       0 | NO PAD        |
| utf8mb4_is_0900_as_cs      | utf8mb4 | 280 |         | Yes      |       0 | NO PAD        |
| utf8mb4_ja_0900_as_cs      | utf8mb4 | 303 |         | Yes      |       0 | NO PAD        |
| utf8mb4_ja_0900_as_cs_ks   | utf8mb4 | 304 |         | Yes      |      24 | NO PAD        |
| utf8mb4_latvian_ci         | utf8mb4 | 226 |         | Yes      |       8 | PAD SPACE     |
| utf8mb4_la_0900_ai_ci      | utf8mb4 | 271 |         | Yes      |       0 | NO PAD        |
| utf8mb4_la_0900_as_cs      | utf8mb4 | 294 |         | Yes      |       0 | NO PAD        |
| utf8mb4_lithuanian_ci      | utf8mb4 | 236 |         | Yes      |       8 | PAD SPACE     |
| utf8mb4_lt_0900_ai_ci      | utf8mb4 | 268 |         | Yes      |       0 | NO PAD        |
| utf8mb4_lt_0900_as_cs      | utf8mb4 | 291 |         | Yes      |       0 | NO PAD        |
| utf8mb4_lv_0900_ai_ci      | utf8mb4 | 258 |         | Yes      |       0 | NO PAD        |
| utf8mb4_lv_0900_as_cs      | utf8mb4 | 281 |         | Yes      |       0 | NO PAD        |
| utf8mb4_mn_cyrl_0900_ai_ci | utf8mb4 | 322 |         | Yes      |       0 | NO PAD        |
| utf8mb4_mn_cyrl_0900_as_cs | utf8mb4 | 323 |         | Yes      |       0 | NO PAD        |
| utf8mb4_nb_0900_ai_ci      | utf8mb4 | 310 |         | Yes      |       0 | NO PAD        |
| utf8mb4_nb_0900_as_cs      | utf8mb4 | 311 |         | Yes      |       0 | NO PAD        |
| utf8mb4_nn_0900_ai_ci      | utf8mb4 | 312 |         | Yes      |       0 | NO PAD        |
| utf8mb4_nn_0900_as_cs      | utf8mb4 | 313 |         | Yes      |       0 | NO PAD        |
| utf8mb4_persian_ci         | utf8mb4 | 240 |         | Yes      |       8 | PAD SPACE     |
| utf8mb4_pl_0900_ai_ci      | utf8mb4 | 261 |         | Yes      |       0 | NO PAD        |
| utf8mb4_pl_0900_as_cs      | utf8mb4 | 284 |         | Yes      |       0 | NO PAD        |
| utf8mb4_polish_ci          | utf8mb4 | 229 |         | Yes      |       8 | PAD SPACE     |
| utf8mb4_romanian_ci        | utf8mb4 | 227 |         | Yes      |       8 | PAD SPACE     |
| utf8mb4_roman_ci           | utf8mb4 | 239 |         | Yes      |       8 | PAD SPACE     |
| utf8mb4_ro_0900_ai_ci      | utf8mb4 | 259 |         | Yes      |       0 | NO PAD        |
| utf8mb4_ro_0900_as_cs      | utf8mb4 | 282 |         | Yes      |       0 | NO PAD        |
| utf8mb4_ru_0900_ai_ci      | utf8mb4 | 306 |         | Yes      |       0 | NO PAD        |
| utf8mb4_ru_0900_as_cs      | utf8mb4 | 307 |         | Yes      |       0 | NO PAD        |
| utf8mb4_sinhala_ci         | utf8mb4 | 243 |         | Yes      |       8 | PAD SPACE     |
| utf8mb4_sk_0900_ai_ci      | utf8mb4 | 269 |         | Yes      |       0 | NO PAD        |
| utf8mb4_sk_0900_as_cs      | utf8mb4 | 292 |         | Yes      |       0 | NO PAD        |
| utf8mb4_slovak_ci          | utf8mb4 | 237 |         | Yes      |       8 | PAD SPACE     |
| utf8mb4_slovenian_ci       | utf8mb4 | 228 |         | Yes      |       8 | PAD SPACE     |
| utf8mb4_sl_0900_ai_ci      | utf8mb4 | 260 |         | Yes      |       0 | NO PAD        |
| utf8mb4_sl_0900_as_cs      | utf8mb4 | 283 |         | Yes      |       0 | NO PAD        |
| utf8mb4_spanish2_ci        | utf8mb4 | 238 |         | Yes      |       8 | PAD SPACE     |
| utf8mb4_spanish_ci         | utf8mb4 | 231 |         | Yes      |       8 | PAD SPACE     |
| utf8mb4_sr_latn_0900_ai_ci | utf8mb4 | 314 |         | Yes      |       0 | NO PAD        |
| utf8mb4_sr_latn_0900_as_cs | utf8mb4 | 315 |         | Yes      |       0 | NO PAD        |
| utf8mb4_sv_0900_ai_ci      | utf8mb4 | 264 |         | Yes      |       0 | NO PAD        |
| utf8mb4_sv_0900_as_cs      | utf8mb4 | 287 |         | Yes      |       0 | NO PAD        |
| utf8mb4_swedish_ci         | utf8mb4 | 232 |         | Yes      |       8 | PAD SPACE     |
| utf8mb4_tr_0900_ai_ci      | utf8mb4 | 265 |         | Yes      |       0 | NO PAD        |
| utf8mb4_tr_0900_as_cs      | utf8mb4 | 288 |         | Yes      |       0 | NO PAD        |
| utf8mb4_turkish_ci         | utf8mb4 | 233 |         | Yes      |       8 | PAD SPACE     |
| utf8mb4_unicode_520_ci     | utf8mb4 | 246 |         | Yes      |       8 | PAD SPACE     |
| utf8mb4_unicode_ci         | utf8mb4 | 224 |         | Yes      |       8 | PAD SPACE     |
| utf8mb4_vietnamese_ci      | utf8mb4 | 247 |         | Yes      |       8 | PAD SPACE     |
| utf8mb4_vi_0900_ai_ci      | utf8mb4 | 277 |         | Yes      |       0 | NO PAD        |
| utf8mb4_vi_0900_as_cs      | utf8mb4 | 300 |         | Yes      |       0 | NO PAD        |
| utf8mb4_zh_0900_as_cs      | utf8mb4 | 308 |         | Yes      |       0 | NO PAD        |
+----------------------------+---------+-----+---------+----------+---------+---------------+
89 rows in set (0.01 sec)
```

- ai: AcccessInsensitive の略 濁音、半濁音を区別しない
- as: AccessSensitive の略 濁音、半濁音を区別する 他、国のコードがあるので、その国の言語用のセットとなっている

さて、ここでは テーブルが `utf8mb4_0900_ai_ci` となっているので、 `utf8mb4_0900_as_cs` を選択する。

```
ALTER TABLE `test_case_sensitive` MODIFY COLUMN `test_varchar` varchar(100) COLLATE `utf8mb4_0900_as_cs`  DEFAULT NULL;
ALTER TABLE `test_case_sensitive` MODIFY COLUMN `test_char` char(100) COLLATE `utf8mb4_0900_as_cs` DEFAULT NULL;
ALTER TABLE `test_case_sensitive` MODIFY COLUMN `test_text` text COLLATE `utf8mb4_0900_as_cs` DEFAULT NULL;

mysql> show create table test_case_sensitive;
+---------------------+---------------------------------------------------------------------------------------------------------------------------
| Table               | Create Table

+---------------------+---------------------------------------------------------------------------------------------------------------------------
| test_case_sensitive | CREATE TABLE `test_case_sensitive` (
  `test_varchar` varchar(100) CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_as_cs DEFAULT NULL,
  `test_char` char(100) CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_as_cs DEFAULT NULL,
  `test_text` text CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_as_cs
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci |
+---------------------+---------------------------------------------------------------------------------------------------------------------------
1 row in set (0.00 sec)

mysql> SELECT * FROM test_case_sensitive WHERE test_varchar = 'aaaaaaaaaaaaaaa';
Empty set (0.00 sec)

mysql> SELECT * FROM test_case_sensitive WHERE test_char = 'aaaaaaaaaaaaaaa';
Empty set (0.00 sec)

mysql> SELECT * FROM test_case_sensitive WHERE test_text = 'aaaaaaaaaaaaaaa';
Empty set (0.00 sec)

mysql>
mysql> SELECT * FROM test_case_sensitive WHERE test_varchar = 'AAAAAAAAAAAAAAA';
Empty set (0.00 sec)

mysql> SELECT * FROM test_case_sensitive WHERE test_char = 'AAAAAAAAAAAAAAA';
Empty set (0.00 sec)

mysql> SELECT * FROM test_case_sensitive WHERE test_text = 'AAAAAAAAAAAAAAA';
Empty set (0.00 sec)

mysql>
mysql>
mysql> SELECT * FROM test_case_sensitive WHERE test_varchar = 'aaaAaaAaAaaAAaA';
+-----------------+-----------------+-----------------+
| test_varchar    | test_char       | test_text       |
+-----------------+-----------------+-----------------+
| aaaAaaAaAaaAAaA | aaaAaaAaAaaAAaA | aaaAaaAaAaaAAaA |
+-----------------+-----------------+-----------------+
1 row in set (0.00 sec)

mysql> SELECT * FROM test_case_sensitive WHERE test_char = 'aaaAaaAaAaaAAaA';
+-----------------+-----------------+-----------------+
| test_varchar    | test_char       | test_text       |
+-----------------+-----------------+-----------------+
| aaaAaaAaAaaAAaA | aaaAaaAaAaaAAaA | aaaAaaAaAaaAAaA |
+-----------------+-----------------+-----------------+
```

```
1 row in set (0.00 sec)

mysql> SELECT * FROM test_case_sensitive WHERE test_text = 'aaaAaaAaAaaAAaA';
+----------------+----------------+----------------+
| test_varchar   | test_char      | test_text      |
+----------------+----------------+----------------+
| aaaAaaAaAaaAAaA | aaaAaaAaAaaAAaA | aaaAaaAaAaaAAaA |
+----------------+----------------+----------------+
1 row in set (0.00 sec)
```

おお～いけてますね。
※全部のカラムで変更したいのであれば、テーブルの `COLLATE` を変えるほうが良いですね。

COLLATE の `_cs`, `_bin` どっちがいいのか

ケースバイケースではありますが、個人的には `_bin` はバイナリ比較し、すべて区別できるので良いのではないかと思います。
※なにか問題が発生した際に、ここの可能性をつぶせるというのはでかいですね。
`_cs`のメリットは何なんだろう…。わからない…。おそらくパフォーマンスの部分なのだろうか。

と思ったけど、chatGPT君に聞いてみたらこうだった。

> cs と bin の照合順序を比較した場合、一般的には bin の方が cs よりもパフォーマンスが良いとされています。
>
> cs は文字列の比較時に大文字と小文字を区別するため、文字列の比較には多少のオーバーヘッドが発生します。一方、bin はバイト単位で比較するため、cs よりも高速に文字列の比較ができます。
>
> ただし、照合順序のパフォーマンスは、使用するデータベースやデータベースの設定、クエリの複雑さなどによって異なります。そのため、アプリケーションの要件に応じて、適切な照合順序を選択する必要があります。性能の観点から照合順序を選択する場合は、実際に性能テストを実施して、適切な照合順序を決定することが重要です。

結論からすると要件によるという感じですね。
私は、 `utf8mb4_bin` を使います。

参考

- What is difference between binary and non-binary string data types?
  https://www.mysqlfaqs.net/mysql-faqs/Data-Types/What-is-difference-between-binary-and-non-binary-string-data-types
- MySQL - 10.3.5 カラム文字セットおよび照合順序
  https://dev.mysql.com/doc/refman/8.0/ja/charset-column.html
- MySQL - 10.8.5 バイナリ照合順序と_bin 照合順序
  https://dev.mysql.com/doc/refman/8.0/ja/charset-binary-collations.html
- MySQL - B.3.4.1 文字列検索での大文字/小文字の区別
  https://dev.mysql.com/doc/refman/8.0/ja/case-sensitivity.html
- MySQLのencodingをutf8からutf8mb4に変更して寿司ビール問題に対応する
  https://techracho.bpsinc.jp/hachi8833/2020_11_26/25044#mysqlutf8mb4
- 寿司ビール問題①　初心者→中級者へのSTEP20/25
  https://qiita.com/kamohicokamo/items/3cc05f63a90148525caf
- Collation(デフォルト照合順序, 大文字と小文字, 半角と全角)
  https://www.wakuwakubank.com/posts/797-mysql-collation/
- MariaDB(MySQL)の照合順序の話
  https://blog.ver001.com/mariadb_collation/

最後に

データベースに関して多少なりとも知識はあったが、それはSQLの構文とかそういうレベルの話で、一歩踏み込んだ部分には何もわからないことがわかった。
このあたり知識をつけていきたいなあ。
参考にさせていただいたサイト様は大変勉強になりました、ありがとうございます。m(_ _)m

MySQL DB RDBMS
❮ barcodeを使ってバーコードを生成する 前へ ChatGPT APIが提供されたので使ってみる 次へ ❯
目次

- はじめに
- 環境
- 準備
- 特定の型のカラムで大文字/小文字を区別できるようにする