Georgian: Comments on Database Stability

L2/17-050

Author: Steven R. Loomis @srl295 srloomis@us.ibm.com

URL to this document: https://gist.github.com/srl295/05ff5730bcc40a66bb0a9a4b1af6d843

Introduction

Relevant to the response to point 2 in L2/17–045 (Razmadze), here are a few comments on how addition of codepoints can cause database instability when different implementations are used.

Background

> 1. If a new set of capital letters were added now, it would be very destabilizing and lead to serious representation and interoperability issues. Names in databases, for example, would be destabilized if capitalization was introduced. Searching and comparison operations would no longer give expected results, which would be significant when analyzing the large corpus of existing documentation.
>
> We have had consults with local specialists and based on their conclusions, names in databases will not be causing destabilization. There won't be any problems with searching, either, since Mtavruli and Mkhedruli letters will be linked through the case pairing function and Mtavruli letter results will also be represented in Mkhedruli letter searches, just like in Latin letter search for the word "GEORGIA" we will receive the lowercase result – "georgia" as well.

Step One: Today

Suppose database table `d#1` (perhaps a list of singers) has a constraint on it, which only allows Georgian text.

As of now, Mkhedruli and Mtavruli are unified. So certain codepoints are allowed:

- `U+10D0 ა GEORGIAN LETTER AN`
- `U+10D1 ბ GEORGIAN LETTER BAN`
- …

To compare the example above to Latin, it would be as if the database allowed only the letters `a…z` such as `georgian`.

Another database table `d#2` (perhaps a list of songs) could refer to `d#1`, and have the same constraint.

Also, note that `UPPER('georgian')` would produce `georgian`.

A real example from mysql today illustrates this: (imagine it is `d#2`)

```
UPPER('მადღობა')
მადღობა
UPPER('суп')
СУП
```

and sqlite:

```
sqlite> select UPPER('მადღობა');
მადღობა
```

So today `UPPER('მადღობა')` just produces `მადღობა`.

Step Two: Some, but not all, implementations support disunified Mkhedruli/Mtavruli

If `d#1` were to allow disunified Mtavruli characters as proposed, its constraint might be changed to allow these, such as `GEORGIAN`. Also, `select UPPER('georgian')` would evaluate to `GEORGIAN`.

If `d#2`, however, did *not* support the disunified Mtavruli characters, as it was still stuck on Unicode 10.0, then an entry such as `GEORGIAN` in `d#1` could not match the corresponding entry in `d#2`. `select UPPER('georgian')` would NOT evaluate to `GEORGIAN`. It would evaluate to `georgian`. Searching for `GEORGIAN` from `d#1` would not match anything in `d#2`

`d#2` and `d#1` would produce different results when calling `UPPER('georgian')`. A `SELECT` statement using both of these tables would not match until/unless `d#2` was updated.