**Database of Laboratory Creep and Shrinkage Experiments – MySQL and xlsx versions**

A user’s and reference manual

V. Šmilauer, L. Dohnalová, M. Jirásek, J. Sanahuja, R. Wan-Wendner, Z. P. Bažant

July 15, 2023

# Introduction

The first large worldwide database of creep and shrinkage tests was assembled at Northwestern University (NU) in 1978. It was later expanded as the RILEM database in 1992, with major restructuring and verification in 2015 [1,2]. The previous database versions used linked spreadsheets (xls, xlsx). As the database has gradually expanded, the need for larger data curation system became obvious. During 2020, the last maintained version *CreepShrinkData\_20191209GB.xlsx* was revamped to this MySQL version. The conversion led to the following improvements:

* Data separation into 11 referenced tables, e.g. ***mix****,* ***specimen****,* ***creep\_test****,* ***creep\_data****.* This eliminated duplicate entries to a large extent. For example, a single mix can be used in several specimens and several creep and shrinkage tests linked with the same mix. See Fig. 1 further.
* Enumerated lists wherever possible, e.g. list of countries (DE, JP, US,...), aggregates (Andesite, Amphibolite, Basalt…), aggregate processing (Crushed, Quarried,…).
* Description of each column in the table.
* Specifying mix design according to current practice. Introducing fine and coarse aggregates. Automatic calculation of w/c, w/b, a/c, a/b, total aggregates from the mix design.
* Two cement classification according to ASTM (Type I, Type II, ...) and EN codes (CEM II/B-S 32.5R, …)
* Handling of mineral admixtures (slag, fly-ash, limestone).
* Added mass loss to shrinkage data (column SD\_massLoss).
* Basic cement properties in ***cement*** table.
* Fixing obviously wrong datasets. Positive shrinkage strain means expansion, negative strain contraction, etc.
* Fixes of known problematic datasets mentioned in several research papers.
* Added missing data, such as aggregate specification in Keeton, J.R.: Study of creep in concrete, Technical reports R333-I, R333-II, R333-III, 1965.
* Added new data, such as Aguilar, C. P.: Study of the Behavior and Development of a Prediction Methodology for Drying Shrinkage of Concretes (in Spanish), 2005.

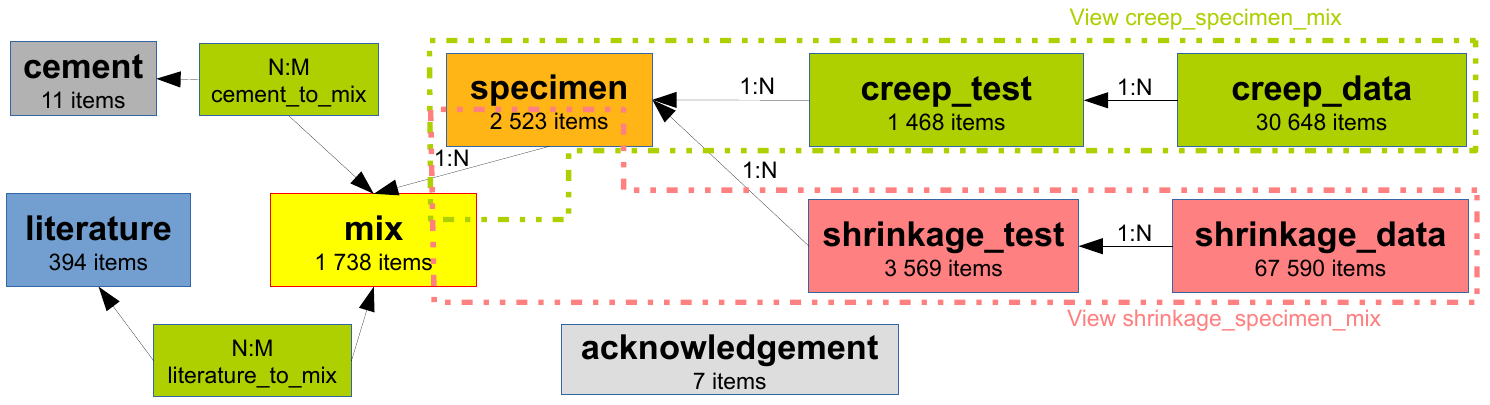
The database can be downloaded in two formats:

* Native MySQL file **\*.sql** (5.6 MB). This file needs deployment on any MySQL server. MySQL server can run as an open-source with local or remote access. The file was created on MySQL server 8.0.33.
* Exported version *xlsx* with individual tables and with joined tables for creep and shrinkage data, \***.xlsx** (22.0 MB). Xlsx version may be used for a quick searching, however, lacks several SQL database features.

The most recent version was deployed at Zenodo repository, older version from 2021-05-03 is downloadable from home page of Prof. Bažant [3].

# Database structure

The MySQL database contains 11 tables, see Fig. 1. Each table contains a primary key (so called a parent table), which can be referenced from another table (so called a child table).

Fig. 1: Tables and structure of Creep and Shrinkage Database

The majority of relations is of 1:N type, e.g. one ***mix*** is referenced in several ***specimens***. Only the table ***literature\_to\_mix*** is of N:M type, i.e. one mix can be mentioned in more resources and one resource can contain several mixes. Two auxiliary tables ***change\_log*** and ***acknowledgment*** contain additional information.

The arrows in Fig. 1 present relation logic. For example, each record in ***creep\_data*** contains a reference *CD\_CT\_id* to a primary key *CT\_id* in ***creep\_test***. Those two tables can be joined on the condition *CD\_CT\_id= CT\_id*, leading to another table containing all information (and also duplicating values from ***creep\_test*** in ***creep\_data*** from the same test.

MySQL contains so-called Views, that virtually join several tables on-the-fly, see Fig. 1. This database defines ***creep\_specimen\_mix*** and ***shrinkage\_specimen\_mix*** views, each joining four particular tables. This largely alleviates tedious joining of four tables, the view appears as a new table with the same functionalities. Each row in these views represent a set of all values from four tables, related to given creep or shrinkage data.

# MySQL Database installation

On **Windows** we suggest to install XAMPP application, which can be downloaded from <https://sourceforge.net/projects/xampp/> It contains Apache server, MySQL and phpMyAdmin (A web interface for easy management of databases).

On **Ubuntu** XAMPP can also be installed. Another option is to install MySQL from APT repository as package “mysql-server”. It requires some configuration, further described here for e.g. Ubuntu 20.04: <https://www.digitalocean.com/community/tutorials/how-to-install-mysql-on-ubuntu-20-04>

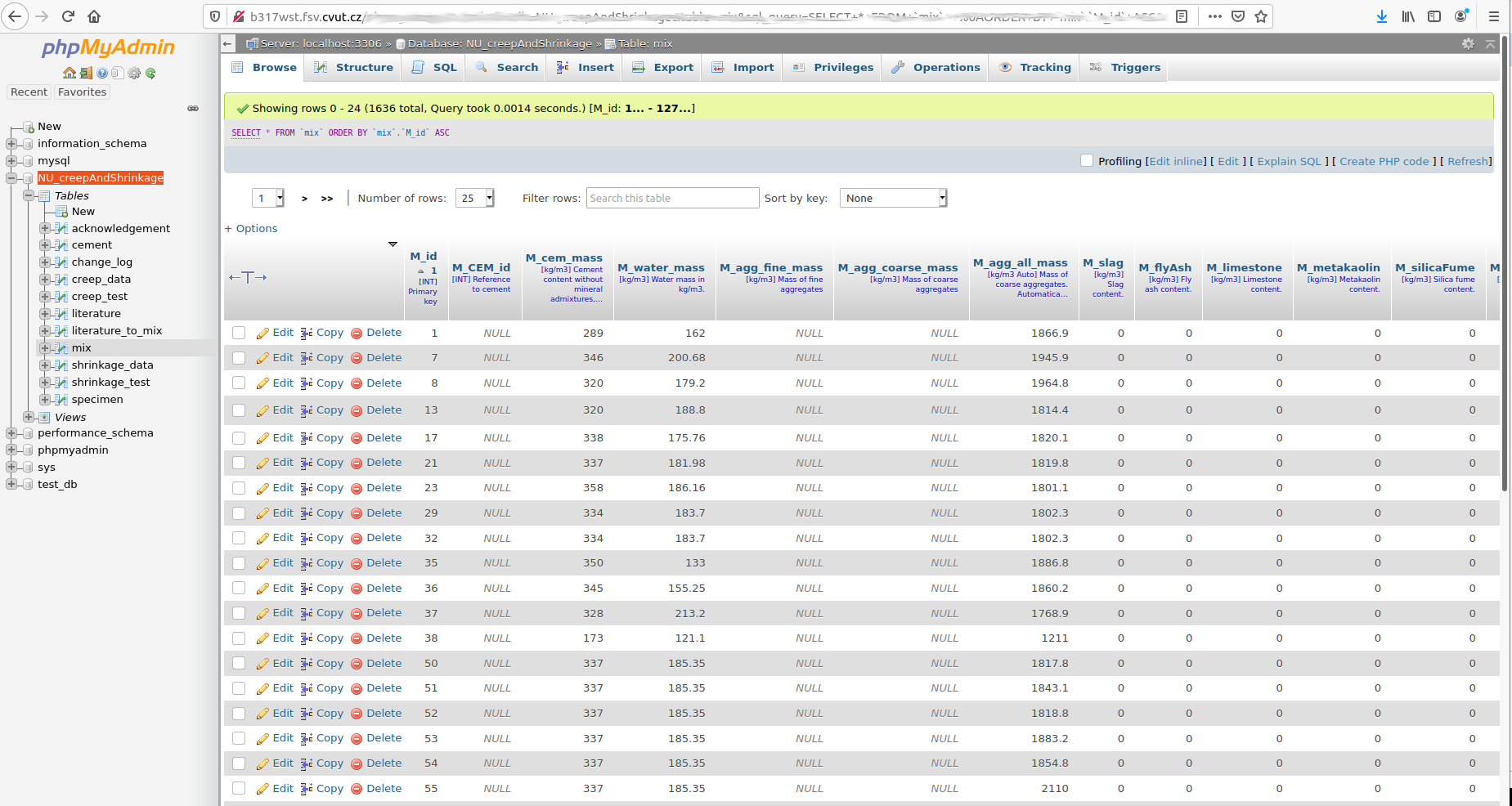
A new database needs to be created e.g. “NU\_creepAndShrinkage” and import the data from the \*.sql file into it. In case of Ubuntu with the “mysql-server” package we create a new database and import the data with the following SQL commands (as root):

The same can be facilitated by phpMyAdmin’s web interface, which usually comes with the MySQL installation, see Fig. 2. Click on “New” in the list of databases, insert the name and click on “Create” button. Select the new database and click on “Import”, choose the file and submit it to the server.

CREATE DATABASE NU\_creepAndShrinkage;

USE NU\_creepAndShrinkage;

SOURCE /path/to/filename.sql;

Fig. 2: phpMyAdmin web interface for MySQL

# Accessing the database and SQL querying

The data from the database can be searched and filtered using queries in the SQL language, which can be submitted using a command line, phpMyAdmin or e.g. Python3 interface. The latter is quite versatile since the use of many existing tools for data handling and plotting is possible. For a connection to MySQL database we can use *pymysql* module or *mysql.connector* one. The following example of a Python3 script displays mixes in the particular range of water/cement ratio.

It produces the following output:

import pymysql

from pprint import \*

cnx = pymysql.connect(host='localhost', database='csdb', user='username', password='password')

c = cnx.cursor()

c.execute('SELECT M\_id, M\_cem\_mass, M\_water\_mass, M\_wToc FROM mix WHERE M\_wToc >= 0.39 AND M\_wToc <= 0.40;')

cnx.commit()

results=c.fetchall()

pprint(results)

cnx.close()

The query can be modified to add more conditions.

((726, 343.0, 136.86, 0.399),

(1062, 390.0, 154.83, 0.397),

(1063, 390.0, 154.83, 0.397),

(1238, 429.1, 171.6, 0.3999),

(1782, 416.0, 163.9, 0.394),

(1792, 516.0, 203.82, 0.395),

(2165, 390.0, 154.83, 0.397),

(2166, 390.0, 154.83, 0.397),

(2590, 394.0, 154.84, 0.393),

(2809, 429.1, 171.6, 0.3999),

(3029, 450.0, 176.7, 0.3927),

(3077, 424.0, 167.0, 0.3939))

Each column has a unique name, which starts with the abbreviation of the table name. Columns from ***cement*** table begin their naming with “CEM\_”, ***creep\_data***’s columns begin with “CD\_” etc. It enhances orientation in the queries with multiple conditions, as we can analogically search data in the views, which in general means searching with conditions containing column names from multiple tables. For example, we can search all creep data of specimens with a ‘slab’ geometry:

Which produces the following output with 299 data records:

import pymysql

from pprint import \*

cnx = pymysql.connect(host='localhost', database='csdb', user='username', password='password')

c = cnx.cursor()

c.execute('SELECT CT\_id, CD\_id, CD\_dt, CD\_Jcreep FROM creep\_specimen\_mix WHERE S\_geometry="Slab" ORDER BY CT\_id, CD\_dt')

cnx.commit()

results=c.fetchall()

pprint(results)

cnx.close()

((1404, 28591, -0.10753, 0.06463),

(1404, 28592, 0.81513, 8.949),

(1404, 28593, 2.808, 13.876),

(1404, 28594, 5.7452, 16.034),

(1404, 28595, 6.7162, 18.203),

(1404, 28596, 8.5334, 20.962),

(1404, 28597, 12.626, 22.923),

(1404, 28598, 19.277, 26.462)…

# Conclusion

MySQL database presents an efficient tool for data curation and querying. Using standardized SQL approach, it is easy to pair creep and shrinkage data or to work remotely on a SQL server. Long-term aim at integration of this creep and shrinkage database into larger concrete databases brings further options for concrete mix design and analyses.

# Acknowledgement

The revamp was supported by Horizon 2020 project “Towards improved assessment of safety performance for long-term operation of nuclear civil engineering structures - ACES”, number 900012, granted under the topic NFRP-2019-2020-01 “Ageing phenomena of components and structures and operational issues”.

# References

1. M. Hubler, R. Wendner, Z.P. Bažant: Comprehensive database for concrete creep and shrinkage: Analysis and recommendations for testing and recording, ACI Materials Journal 112 (4), 547-558, 2015.
2. M. Hubler, R. Wendner, Z.P. Bažant: Statistical justification of Model B4 for drying and autogenous shrinkage of concrete and comparisons to other models, Materials and Structures 48:797–814, 2015.
3. <http://cee.northwestern.edu/people/bazant/downloads.html>