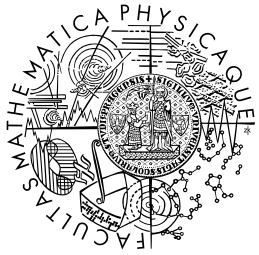

FACULTY
OF MATHEMATICS
AND PHYSICS
Charles University

BACHELOR THESIS

Přemysl Šťastný

LSQL language and it's lsql-csv implementation for csv files

Department of Applied Mathematics

Supervisor of the bachelor thesis: Jan Hubička
Study programme: General Information
Study branch: Informatics

Prague 2021

I declare that I carried out this bachelor thesis independently, and only with the cited sources, literature and other professional sources. It has not been used to obtain another or the same degree.

I understand that my work relates to the rights and obligations under the Act No. 121/2000 Sb., the Copyright Act, as amended, in particular the fact that the Charles University has the right to conclude a license agreement on the use of this work as a school work pursuant to Section 60 subsection 1 of the Copyright Act.

In ..... date ....

Author's signature

Dedication.

Title: LSQL language and it's lsql-csv implementation for csv files

Author: Přemysl Šťastný

Department: Department of Applied Mathematics

Supervisor: Jan Hubička, Department of Applied Mathematics

Abstract: LSQL is language, which is new database query language optimilized for simple onetime queries in command line enviroment. lsql-csv is project implementing LSQL for csv files. The thesis is about reason and design of the language and it's lsql-csv implementation.

Keywords: lsql lsql-csv unix kiss unix-philosophy haskell database csv new-language data-analysis data-query

Contents

Introduction

1. Introduction

1.1 What is LSQL? What it is good for?

Why would anyone come with a new query language for flat data, when there are standartized languages for doing so? It's easy.

The widly used standard for querying flat data is SQL. The SQL is designed to make a human (and machine) readable queries, which can be contained in large projects with many people. The readability of written code comes in the first place and therefore it was designed not for the comfort of the programmer, who actually writes the code, but for hunderts of people, who comes after him and tries to find out, what his code does.

This is the opposite of what LSQL is developed for. We try to make a language, which will make an unix user more comfortable on his machine in the first place. It doesn't care about constraints, try to ignore types as much as possible and is designed by the means of unix philosophy.

Why should we use a poweruser-friendly tool, when we have user-friendly tool to do the same thing, faster[1] and maybe better? Like Excel, Calc or Django admin? Simply put, there are use cases, where user-friendly tools unnecessarly complicate the whole situation, and you want the solution to be as simple as possible. For the sake of your brain, your time, your psyché, the maintability and lifetime[2] of data and scripts and amount of information, you have to remember.

All of these questions will be discussed later in text.

1.2 What is lsql-csv?

lsql-csv is a tool implementing LSQL for quering csv files. The main[3] ambition of this project is to get into standartized UNIX ecosystem. It is simple, useful tools coresponding the KISS (keep it simple stupid) and UNIX philosophy (mainly do only one thing and do it right).

Similary of the Java slogan is Write once, run anywhere, the author thinks, the UNIX ecosystem slogan should be Write once, run forever. The reason, why would you want to store data in csv is not only its simplicity and usefulness, but that you can be sure, you can open UTF-8 csv file 30 years later, and you will be probably able to run your UNIX ecosystem scripts.

1.3 Who is this text for?

This text is for people, who wants to understand the usefulness of LSQL, lsql-csv and the reason, why and how they exist. If you want to try the tool without knowing anything deeper about it, you might consider to read README.md instead of this text.

[1] You don't have to learn new language.

[2] Have you noticed, how offen Microsoft Excel or Postgresql are changing the database format and how they are complex, when we compare them to csv?

[3] and unrealistic

2. Use cases and their standard solutions with or without lsql-csv

It was once sayed, that a simple example can be more then thousand words. The author absolutly agrees with this thesis and therefore before any generilezed analysis outcome, he has decided to write down the real world motivation.

2.1 Student subject evidence

Let's say, you are an teacher, who needs to make an evidence of grades given to his students. This is normal thing, what teachers does, but there is no best way how to do it. Each solution has its pros and contras and we will try to examine the most used of them to see, why lsql-csv can be useful.

2.1.1 Usage of paper

The classical solution since the paper and pen is widly avaliable. The first problem with this aproach is losing the paper...this can become an absolute hell, when you are in the end of the semester and you have to remake your evidence from the students paperwork.

Even it is the most simple solution available, I would not recommend it as it isn't much scalable, you waste part of your life on paper sorting and paper saving, evaluating data (average, pass/not passed,...), you will feel bad if you are an ecoactivist, and finally you will be worried about losing the paper, which can't be even read by other people because of GPDR.

One of the largest problem of this approach are the students, who wants to know their grades, but you can't tell it to all of them at one because of their privacy rights. You have to speak with each student separatly or give them an paper/letter with their results. Nightmare. - Yes, you can make the nicknames to anonymize, but the whole magic of the simplicity of the solution will be gone. Students may also have problem to remember their chosen nicknames and therefore you may be bombarded by disoriented parents[1] and forgotful students.

Pros

- The easiest reasonable solution
- Fast to plant
- Readable for centuries (if well archived)

Cons

- Hard to share (with all respects to the law[2]

[1] In elementary school

[2] You can't scan the paper and put it in the internet like in the old times and nicknames make a unwanted chaos.

- Hard to well archive and it is all up to you and your reponsibility
- Hard to evaluate statistical analysis[3] and final results for students

2.1.2 Usage of specialized software - SIS, Bakalari,...

This is obviosly a working solution, but there can be many unexpected problems, if you are using nonstandard grading system.

If you don't have any special requirements and your institution is rich enough, this may be the right solution for you. It is fast to learn, tested and gives access students to see their grades.

There is also one problem, which is being underestamated by many administrators, and that is the timelife of the data and the software. The project being unmaintained[4] in future may be a large problem for you, if you are obligated by law to archive old results. You may have problems with future architecture compatibility, API compatibility,...

If the software is maintained in unproper way, there can be also security problems. There may be few deliquents, who may sabotage the database and make the whole system untrusted. - If every teacher maintain their own database (for example on a paper), it may be much harder to do the same amount of damage to the evidence as to the centralized system using a fatal security bug.

Pros

- Easy data processing for an institution, teachers and students
- Easy to learn
- Easy sharing respecting privacy standarts
- Possibly easy archivation

Cons

- Usually not flexible enough
- Possibly money and time eating solution for your institution
- Possible security problems
- Very slow plantation

2.1.3 Usage of Microsoft Excel or Libreoffice Calc - offline version

We may use a general spreadsheet software to save data. It is well working and easy solution for everyone. Everyone loves it![5] So where is the problem?

[3] It is sometimes wanted to have Gaussian function like results.

[4] Absence of a motivation, people or money, or death of the author may be good reason.

[5] ...sarcasm...

First of all there is no schema for your data, so if you or your organisation wants to analyse data across more spreadsheet, it may have a really hard time. But let's say we don't employ any "expert" to do that so nobody wants anything like that.

The solution looks nice at first look. You may compute averages, evaluate conditions,... without almost any effort. But when you want to do some real scripting, you may have a problem. Imagine, you want to automatically select subset of data (grades without real names(, which you give students. You will have to write a large amount a code for that...it is time eating, but after some time, you might realise, that it is easier and faster to do CTRL+C and CTRL+V after each new input rather then trying to program it yourself. Yes, you can convert spreadsheet format to csv file using some utility, but you can never be sure whether the next version of your spreadsheet software won't break your scripts up. The best thing is, you will find out few years later after your scripts and work on them is done.

This is the first place, where the frustration begins. You have your hands tied up with the limits, size, complexity and inovation of your spreadsheet software and you can't the things which would make your life easier.

The problem with the inovation of format of spreadsheet processor might not be only an problem for scripting, but also for archivation. We don't know, how many times and how much Microsoft (or someone else) will change their spreadsheet format and how good backward compatibility they will maintain. You might find yourself one day in position, when you are obligated to submit data and you will have to waste days your time to decode obsolated formats. That's not good vision for the future.

Pros

- Moderatetly hard to learn
- Fast plantation
- Easy basic statistical analysis and evaluation
- Flexible

Cons

- Chaos in data across an organisation
- Possible incompatibility through time
- Incompability with other aplications and languages
- Nontrivial archivation
- Hard to keep shared data with students up to date

2.1.4 Usage of online spreadsheet software

2.1.5 Usage of csv

2.2 Rapid analysis of statistical data

2.3 Madhouse internet censorship

Conclusion

List of Figures

List of Tables

List of Abbreviations

A. Attachments

A.1 First Attachment