Faculty of Business and Economics
The University of Hong Kong

Data Processing Domain-Specific Language in Clojure

Research Proposal

10 January, 2022

CHOI Chong Hing

Supervised by Dr. Matthias Buehlmaier

1 Background

Domain Specific Language (DSL) is a computer laugage, declared syntax or grammar that is specialised to a specific application. In contrast to General-Purpose Language, implemenation of DSL is designed with specific goals in that application domain. The use of macros in Lisp dialects enables developers to rewrite source code at compile time, making implementation of DSL more convenient. As one of the Lisp dialects, Clojure also inheriates such advantage. In addition to macros, the heavy use of core data literals in Clojure also gives an extensive developing opportunity in implementing DSLs.

In this project, a DSL extension to the existing data processing library, tech.ml.dataset[1], will be developed. A generic query using core data literal serves as the foundation of the DSL. This enables huge flexibility in defining the syntax, subject to Clojure's limitation.

2 Methodology

A generic query with heavy use of Clojure core data literals will be developed. It serves as the foundation of the project DSL. It includes a combination of data query operations, including selecting rows and columns, filtering with conditions and group by. The generic query should be able to execute multiple operations. To achieve such promise, the processing steps of Structed Query Language (SQL) will be used as a reference in designing the generic query.

With the generic query foundation implemented, the design of syntax will be the next step. In this step, the mix of Clojure literals and macros will be used, depending on the syntax design. The macro system in Clojure, and other Lisp dialects, allows developers to extend the compiler by manipulating code expression at compile-time. It suspends the compilation of an expression and brings the expression for user manipulation. Expressions with wrong grammar or unknown keywords could be correctly and manually compiled via macros, making different expressions possible in Clojure. With the use of macros, the implementation syntax will be much more flexible.

References

[1] "Techascent/tech.ml.dataset: A clojure high performance data processing system." (), [Online]. Available: `https://github.com/techascent/tech.ml.dataset`.