Wrangell

A DSL for Data-Wrangelling

Dana Iltis   Kenan Nalbant   Donald Pinckney

March 22, 2017

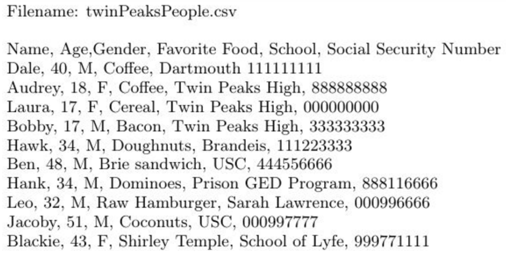
Data Wrangling: a time-consuming and tedious job

Why do we need a DSL for data-wrangelling?

- Wide variety of file formats
- Difficult to accomplish for users with minimal programming experience
- Often requires manual labor to create ad-hoc solutions

A Motivating Example

Perhaps we want to remove sensitive information from a database



```
Filename: twinPeaksPeople.csv

Name, Age,Gender, Favorite Food, School, Social Security Number
Dale, 40, M, Coffee, Dartmouth 111111111
Audrey, 18, F, Coffee, Twin Peaks High, 888888888
Laura, 17, F, Cereal, Twin Peaks High, 000000000
Bobby, 17, M, Bacon, Twin Peaks High, 333333333
Hawk, 34, M, Doughnuts, Brandeis, 111223333
Ben, 48, M, Brie sandwich, USC, 444556666
Hank, 34, M, Dominoes, Prison GED Program, 888116666
Leo, 32, M, Raw Hamburger, Sarah Lawrence, 000996666
Jacoby, 51, M, Coconuts, USC, 000997777
Blackie, 43, F, Shirley Temple, School of Lyfe, 999771111
```

Solution: Create a DSL

Why create *another* DSL

- We want ease of usage when it comes to this particular problem domain while still allowing for a lot of flexibility
- Make common place tasks easier to accomplish and eliminate tedious boilerplate that writing an ad-hoc solution in another language would require

Introducing Wrangell

Wrangell is a DSL which eliminates much of the boilerplate code that typically arises when data-munging.

- Based on a familiar Lisp-like syntax which while simple is very expressive.
- Very extensible, arbitrary new file formats can be supported for input and output as long as functions which map to and from Wrangell's internal data representation are provided.
- We have a working interpreter implemented in Haskell

Wrangell

- Has a dynamic type system as with Lisp, but much more strict, with less implicit type coercions
- Additionally Wrangell supports polymorphic expressions, if the functions used in an expression can accept many different types so will the compound expression.

Wrangell

- Wrangell also requires columns of the data set to have types.
- All column transformations are type checked to guarantee type-soundness.

Why Haskell

- Because Haskell is a functional language, we thought that data-wrangling, with its emphasis on data transformations would be well suited and also an instructive experience on how to build large scale functional programs

Future Work

- Currently the only supported type of input and output files are character delimited (e.g. CSV or TSV) so work can be done to make more file formats supported
- There is currently a degree of inefficiency with the current implementation of many of the internal data transformations which can absolutely be remedied

Conclusion

- Wrangell is a novel language which provides many primitives to easily deal with data-preprocessing tasks
- Wrangell additionally provides facilities to easily extend to new file types.
- The type system on both Wrangell expressions and internal data tables allows for both expresiveness and safety