HeisenBase: Understanding Database Performance in Haskell

Abstract
We explore the impacts of building a database management system in Haskell, a strongly typed, pure functional programming language.

Lazy Evaluation
Lazy evaluation puts off evaluation of functions until the results are needed. Our B+ Tree index can remain unevaluated until a query necessitates evaluation.

Motivations
• Millions of man-hours in DBs
• Typically implemented in C++
• What can we gain with a functional implementation?

Architecture
HeisenBase provides a REPL for executing queries in the supported subset of standard SQL.

Performance
Query evaluation causes the index to be computed, resulting in dramatically faster consecutive queries.

Contributions
• Proven prototype of Haskell database
• Insight into lazy evaluation performance

Zachary Wasserman
Advised by
Susan Davidson

Senior Project Poster Day 2013 – Department of Computer and Information Science – University of Pennsylvania