Minerva: A modern, free software database system

James Percent
james@empty-set.net

Syndetic Logic, Boston, MA USA

June 8, 2011

Abstract

Minerva is a modern database system with its design at the cross-roads of software engineering and database implementation theory. It's state-of-art and it's also free for everyone to modify and learn from.

Contents

List of Figures

1 Introduction

Includes modern features: injection, testability composition, performance anlaysiable

Why? Mostly because I just love creating great software. But also I believe there is a gaping hole in availability of free software that embodies the ideas, widely accepted in the research communities.

Basic ideas use modern software engineering technicques to create a fully functional database that supports researching new algorithms. The lego principle of connecting components

Supports decorating major subsystems with changes. Interrupting the behavior of subsystems. Building in correctness checking software; parallelizing algorithms.

I never got the whole uproar regarding the CAP theorem. Let me be clear, I understand what the CAP theorem says, but draws focus to an issue that is irrelevant. Hierarchies of systems.

1.1 Injectable

Software composition. It fucking matters.

1.2 Testable

Performance and correctness.

1.3 Infrastructure for research

Includes all kinds of infrastructure for reasearch to take advantage. Cloud deployment for testing on large scale. Already mentioned, performance and correctness.

2 Linearis

As abstract as possible, without being too abstract. There is a thing called a namespace that encapsulates a set, which contains 1 or more arrays. These corrospond to schema, relation and attribute. You might be thinking, why the fuck did he name everything differently. It's because they do not describe a schema, relation or attribute, respectively, in Linearis. In another system, in fact they are. But future systems may find a different use for Linearis, and then an artificial naming scheme, based on relational database terminology,

would be even more inconvenient. Therefore, I decided it was better to use naming that served only inside of Linearis and not out.

Set name should go away.

2.1 CompositeKeys

I have to make this really work tomorrow. I need to take notes on how it works because I'm doing some wacky stuff in there. The keys are coded as follows. The first byte encodes the first 4 types. Types are byte=0, int=1, long=2, string=3, the first 4 types follow, and so on. This HAS A PROBLEM because the first byte encodes type BYTE as a 0. But then empty data after that may be consider an element.

Multi-version concurrency control. I can achieve it by breaking my maps into sections and storing, if necessary multiple sections per version. this will work well given i do not expect too many updates. Further, i can provide more index information. Actually, after some reflection, I like the idea of keeping deltas for MVCCs better. I can make deltas without explicit keys. When I worked on the Vertical Store at Akiban Technologies we had the btree+ hand me the deltas with an associated key. We needed the key to keep the hierarchical mapping.

3 Bratus

4 Distributus

5 Logicus

References