Socrates 动态语义建模工具

刘鑫 <march.liu@gmail.com>

动态数据库建模与三值逻辑

March 14, 2010

关系型数据库（一）

库、表、字段、记录

- 库是表的集合

关系型数据库（一）

库、表、字段、记录

- 库是表的集合
- 通过字段集合定义表结构

关系型数据库（一）

库、表、字段、记录

- 库是表的集合
- 通过字段集合定义表结构
- 同构的记录保存于同一个表中

关系型数据库（二）

传统的设计理念可以直接对应强类型静态语言

- 程序包含类

关系型数据库（二）

传统的设计理念可以直接对应强类型静态语言

- 程序包含类
- 类型定义由成员组成

关系型数据库（二）

传统的设计理念可以直接对应强类型静态语言

- 程序包含类
- 类型定义由成员组成
- 对象结构由类型决定

传统思路遇到的问题

所有静态强类型开发技术会遇到的问题

- 随着程序运行，出现结构变更

传统思路遇到的问题

所有静态强类型开发技术会遇到的问题

- 随着程序运行，出现结构变更
- 快速开发过程，不断修改数据库结构

传统思路遇到的问题

所有静态强类型开发技术会遇到的问题

- 随着程序运行，出现结构变更
- 快速开发过程，不断修改数据库结构
- 数据形成稀疏矩阵效应

动起来

我们该怎么做？

- 用字典管理数据

动起来

我们该怎么做？

- 用字典管理数据
  - PG HStore

动起来

我们该怎么做？

- 用字典管理数据
  - PG HStore
  - XML

动起来

我们该怎么做？

- 用字典管理数据
  - PG HStore
  - XML
  - JSON

动起来

我们该怎么做？

- 用字典管理数据
  - PG HStore
  - XML
  - JSON
- 动态语言

动起来

我们该怎么做？

- 用字典管理数据
  - PG HStore
  - XML
  - JSON
- 动态语言
  - EAV

动起来

我们该怎么做？

- 用字典管理数据
  - PG HStore
  - XML
  - JSON
- 动态语言
  - EAV
  - 持久化 DSL

动起来

我们该怎么做？

- 用字典管理数据
  - PG HStore
  - XML
  - JSON
- 动态语言
  - EAV
  - 持久化 DSL
  - Socrates

项目介绍

http://bitbucket.org/March/socrates

- 基于三元语义的数据建模工具

项目介绍

`http://bitbucket.org/March/socrates`

- 基于三元语义的数据建模工具
- 面向关系型数据库的访问工具

项目介绍

`http://bitbucket.org/March/socrates`

- 基于三元语义的数据建模工具
- 面向关系型数据库的访问工具
- 针对动态数据结构实现

项目介绍

`http://bitbucket.org/March/socrates`

- 基于三元语义的数据建模工具
- 面向关系型数据库的访问工具
- 针对动态数据结构实现
- 面向数据关系

三元语义与三值逻辑

三元语义就是（主语，谓语，宾语）三元组

- 每个宾语代表一个具体的数据

三元语义与三值逻辑

三元语义就是（主语，谓语，宾语）三元组

- 每个宾语代表一个具体的数据
- 每个主语代表一个相关主题

三元语义与三值逻辑

三元语义就是（主语，谓语，宾语）三元组

- 每个宾语代表一个具体的数据
- 每个主语代表一个相关主题
- 谓语解释主语与宾语的关系

三元语义与三值逻辑

三元语义就是（主语，谓语，宾语）三元组

- 每个宾语代表一个具体的数据
- 每个主语代表一个相关主题
- 谓语解释主语与宾语的关系
- 修改主题的结构，即增删子句

三元语义与三值逻辑

三元语义就是（主语，谓语，宾语）三元组

- 每个宾语代表一个具体的数据
- 每个主语代表一个相关主题
- 谓语解释主语与宾语的关系
- 修改主题的结构，即增删子句
- 主题可以作为宾语

三值逻辑

三元语义的定义与划分

- 用谓词表达结构

三值逻辑

三元语义的定义与划分

- 用谓词表达结构
- 用宾语表达数据

三值逻辑

三元语义的定义与划分

- 用谓词表达结构
- 用宾语表达数据
- 元语义

动态语义实现

强类型动态类型工具

- 基于 SQLAlchemy 实现

动态语义实现

强类型动态类型工具

- 基于 SQLAlchemy 实现
- 宾语可以是任何类型

动态语义实现

强类型动态类型工具

- 基于 SQLAlchemy 实现
- 宾语可以是任何类型
- 可以书写为三值逻辑解释

动态语义实现

强类型动态类型工具

- 基于 SQLAlchemy 实现
- 宾语可以是任何类型
- 可以书写为三值逻辑解释
- 类似字典的结构（持续改进中）

元语义 I

用三元语义解释三元语义

- subject is type
- predicate is subject
- is is predicate
- is objType subject
- objType is predicate
- objType objType subject
- type is subject
- string is type
- name is predicate
- name objType string
- storage is predicate
- storage objType string

实现目标 I

- 使用 subject predicate object 解释数据
- 解决关系型数据库结构固定，不容易修改的问题
- 解决稀疏表的存储问题
- 建立于数据存储层之上，兼容不同类型的数据库
- 规则简单
- 元数据信息，自描述
- 结构开放，可以开发第三方接口
- 结构简单，基于递归和引用，容易使用
- 数据粒度小，伸缩性好，容易并发
- 可插入到其它体系中使用

已经实现的部分（一）I

基于 SQLAlchemy 的智能存储接口

- 自适应多种不同的数据库
- 构建架构
- 简单查询
- 支持简单的谓词表达式
- 类型动态注册和智能感知

已经实现的部分（二）I

基于 SQLAlchemy 的智能数据对象

- 数据库平台中立
- 友好支持具体平台的特色功能
- object/objects
- 基于谓词表达式的查询
- 数据维护
- 匹配 SQLAlchemy 成熟功能
- 可以直接使用 SQLAlchemy 已有的查询技术

近期目标

提高可用性

- 子句对象字典化

近期目标

提高可用性

- 子句对象字典化
- 理清容器行为逻辑

近期目标

提高可用性

- 子句对象字典化
- 理清容器行为逻辑
- 探索 URI 机制

近期目标

提高可用性

- 子句对象字典化
- 理清容器行为逻辑
- 探索 URI 机制
- 优化查询性能

近期目标

提高可用性

- 子句对象字典化
- 理清容器行为逻辑
- 探索 URI 机制
- 优化查询性能
- 规范化谓词表达式

近期目标

提高可用性

- 子句对象字典化
- 理清容器行为逻辑
- 探索 URI 机制
- 优化查询性能
- 规范化谓词表达式
- 支持查询表达式和查询构建

中期目标

DSL

- DSQL 初步

中期目标

DSL

- DSQL 初步
- SELECT *fields WHERE filters GROUP BY *columns ORDER BY *columns

中期目标

DSL

- DSQL 初步
- SELECT *fields WHERE filters GROUP BY *columns ORDER BY *columns
- UPDATE ... WHERE

中期目标

DSL

- DSQL 初步
- SELECT *fields WHERE filters GROUP BY *columns ORDER BY *columns
- UPDATE ... WHERE
- WRITE ... WHERE/NEW

长期目标

不靠谱的地平线

EUCLID 微观的 Socrates 内存数据结构组件

长期目标

不靠谱的地平线

EUCLID 微观的 Socrates 内存数据结构组件

LIGHTING 基于内存集群的非持久化高速三元语义引擎

长期目标

不靠谱的地平线

EUCLID 微观的 Socrates 内存数据结构组件

LIGHTING 基于内存集群的非持久化高速三元语义引擎

PLATO 基于 PostgreSQL 的原生动态引擎

长期目标

不靠谱的地平线

EUCLID	微观的 Socrates 内存数据结构组件
LIGHTING	基于内存集群的非持久化高速三元语义引擎
PLATO	基于 PostgreSQL 的原生动态引擎
UTOPIAN	组合 Socrates 项目技术的三元语义 EAV 数据库

感谢

谢谢大家在这里与我共度这次分享！