Atitit 基于sql编程语言的oo面向对象大规模应用解决方案attilax总结

[1. Sql语言应该得到更大的范围的应用， 1](#_Toc22155)

[1.1. 在小型系统项目中，很适合存储过程写业务逻辑 2](#_Toc10031)

[1.2. 大型项目或产品型项目（有换数据库需求的），大规模使用js编写业务逻辑，把后端大力简化为一个rest接口的数据库驱动 2](#_Toc13406)

[1.3. 小部分模块，重视安全的，使用后端业务逻辑模式 2](#_Toc3184)

[1.4. 貌似比较理想的模式 前端h5（ui界面）+js（业务逻辑）+rest（通讯层）+ sql（业务逻辑）+数据库（存储） 2](#_Toc20104)

[1.5. Oo知识图谱 2](#_Toc3525)

[2. 基本的oo机制 3](#_Toc21039)

[2.1. 命名空间namespace package机制 3](#_Toc1663)

[2.2. 原型机制 vs Class类模板机制 4](#_Toc3354)

[2.3. 定义class Class模板就使用表来实现啦 4](#_Toc22038)

[2.4. 绑定class方法 getName建立一个com.attilax.city/city\_getName函数 5](#_Toc20071)

[3. 新建对象，并调用对象方法getName 5](#_Toc28312)

[3.1. 创建一个obj， 设置对象属性 5](#_Toc10697)

[3.2. 获得对象引用 5](#_Toc16219)

[3.3. 调用对象方法，有点类似于反射，使用类方法调用对象标识和其他参数 6](#_Toc6631)

[4. 对象的gc 6](#_Toc9834)

[5. 测试主函数main 6](#_Toc8859)

[6. 参考资料 6](#_Toc9806)

测试数据库为mysql ,版本5.6 ，ide工具为navicate10.1

# Sql语言应该得到更大的范围的应用，

但是sql是个dsl语言，函数式语言为主，编程抽象单位以函数（存储过程）为主。。映射到oo稍微有点麻烦。。模拟实现。。

优点，毕竟sql是个4gl语言，比起java c#这类3gl，高了整整一个语言代差。原则上可以更加的简洁化，提升可读性。。

灵活性，热部署

缺点：不适合换数据库的场合。如果需要换数据库，需要翻译到目标数据库的sql语言。

需要很高性能的场合可能不适用。

Sql貌似没有很好的ide和调试工具。

## 在小型系统项目中，很适合存储过程写业务逻辑

## 大型项目或产品型项目（有换数据库需求的），大规模使用js编写业务逻辑，把后端大力简化为一个rest接口的数据库驱动

## 小部分模块，重视安全的，使用后端业务逻辑模式

## 貌似比较理想的模式 前端h5（ui界面）+js（业务逻辑）+rest（通讯层）+ sql（业务逻辑）+数据库（存储）

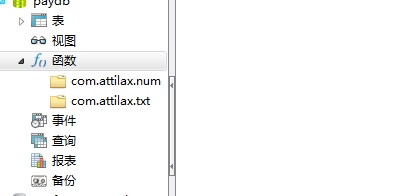
## Oo知识图谱

|  |  |  |
| --- | --- | --- |
| 类与对象实现 |  | oo三大特性 |
| 类与对象实现 |  | solid原则 |
| 类与对象实现 |  | 对象实现 |
| 类与对象实现 |  | class模板 |
| 类与对象实现 |  | 方法重载 |
| 类与对象实现 |  | 属性读写 |
| 类与对象实现 |  | 构造函数 |
| 类与对象实现 |  | 抽象类和 |
| 类与对象实现 |  | 抽象方法 |
| 类与对象实现 |  | 单例 |
| 类与对象实现 |  | 实例 |
| 类与对象实现 |  | 接口用interface |
| 类与对象实现 |  | is-a has-a |
| 类与对象实现 |  | 对象持久性（object persistence |
| 类与对象实现 |  | 访问方法（accessor method）和 |
| 类与对象实现 |  | 设置方法(mutator method) |
| 类与对象实现 |  | 动态绑定（dynamic binding |
| 类与对象实现 |  | 消息 |
| 类与对象实现 |  | 覆盖（Overriding |
| 类与对象实现 |  | dock type |
| 类与对象实现 |  | 基于原型 |
| 类与对象实现 |  | 封装 |
| 类与对象实现 |  | 多态 |
| 类与对象实现 |  | 继承 |
| 类与对象实现 |  | 多继承 |
| 类与对象实现 |  | 动态对象 |
| 类与对象实现 |  | 类加载 |
| 类与对象实现 | 程序集加载 | 自动语句加载 |
| 类与对象实现 | 程序集加载 | 反射api加载 |
| 类与对象实现 | 程序集加载 | 字节数组加载 |
| 类与对象实现 | 程序集加载 | 外部文件加载 |
| 类与对象实现 |  | 魔术属性 |
| 类与对象实现 |  | 魔术方法 |
| 类与对象实现 |  | 构造方法 |
| 类与对象实现 |  | 析勾方法 |

# 基本的oo机制

## 命名空间namespace package机制

Sql标准化没有默认的命名空间机制，使用mysql提供的组机制解决。。大概常用的命名空间模块约100个左右，可以参考java ，net系列的sdk命名空间。。



貌似组和存储过程都是存储在mysql数据库文件里面的。在文件夹没有找到他们的痕迹。。

## 原型机制 vs Class类模板机制

在实现oo的时候，有原型and Class俩种机制。。Sql也是函数式语言，类似js，不过还是使用了class类模板机制

## 定义class Class模板就使用表来实现啦

Class模板就使用表来实现啦。表结构相当于class模板，记录就是具体的对象了。。

比如创建一个class city( city\_id，cityName ） 方法，getName（）

CREATE TABLE `city` (

`city\_id` int(10) NOT NULL,

`city\_name` varchar(50) NOT NULL,

`province\_id` varchar(20) NOT NULL,

`first\_letter` varchar(20) DEFAULT NULL,

`is\_hot` int(10) NOT NULL DEFAULT '0',

`state` int(10) NOT NULL DEFAULT '1',

PRIMARY KEY (`city\_id`)

) ENGINE=MyISAM DEFAULT CHARSET=utf8;

## 绑定class方法 getName建立一个com.attilax.city/city\_getName函数

Mysql的group机制只有开发室有用，运行时是不存在的，所以我们只好在方法名称前面添加命名空间和类名前缀来区分了。

BEGIN

#Routine body goes here...

declare namex VARCHAR(200);

#call log( obj\_id);

select city.city\_name into namex from city where city\_id=obj\_id ;

RETURN namex;

END

# 新建对象，并调用对象方法getName

## 创建一个obj， 设置对象属性

Sql里面默认只有int string等细粒度数据类型。。复合数据类型是使用表集合实现的。。

创建对象，其实就是insert了

insert city(city\_id,city\_name) VALUES(888,'mycity')

创建city对象，并设置city\_name属性。。

## 获得对象引用

对象引用其实就是一个标识id。。这里直接使用city\_id 作为city对象的引用即可。

## 调用对象方法，有点类似于反射，使用类方法调用对象标识和其他参数

set namex=city\_getName(883);

# 对象的gc

使用sql编程，对象的存储是存储在表里面的，因为我们一般很多时候就是为了持久化的对象的，所以就不需要个gc。但是少数撤下，可能只是个临时对象，此时就需要gc，普通的，delete生产的临时对象就可以了，完成了gc过程。。。 虽然需要手动管理一下gc过程，不过还是很容易的。

# 测试主函数main

BEGIN

#Routine body goes here...

declare namex VARCHAR(200);

##创建对象，并制定对象引用标识

insert city(city\_id,city\_name) VALUES(883,'mycity883');

##调用对象方法

set namex=city\_getName(883);

select namex;

END

# 参考资料

Mysql存储过程查询结果赋值到变量的方法 - mysql数据库栏目 - 红黑联盟.html

作者:: 绰号:老哇的爪子claw of Eagle 偶像破坏者Iconoclast image-smasher

捕鸟王"Bird Catcher 王中之王King of Kings 虔诚者Pious 宗教信仰捍卫者 Defender of the Faith. 卡拉卡拉红斗篷 Caracalla red cloak

简称：： Emir Attilax Akbar 埃米尔 阿提拉克斯 阿克巴

全名：：Emir Attilax Akbar bin Mahmud bin attila bin Solomon Al Rapanui

埃米尔 阿提拉克斯 阿克巴 本 马哈茂德 本 阿提拉 本 所罗门  阿尔 拉帕努伊

常用名：艾提拉（艾龙），   EMAIL:1466519819@qq.com

头衔：uke总部o2o负责人，全球网格化项目创始人，uke宗教与文化融合事务部部长，Uke部落首席大酋长，uke制度与重大会议委员会委员长，uke保安部首席大队长,uke制度检查委员会副会长，奶牛科技cto ，uke波利尼西亚区大区连锁负责人，克尔格伦群岛区连锁负责人，莱恩群岛区连锁负责人，uke汤加王国区域负责人，。布维岛和南乔治亚和南桑威奇群岛大区连锁负责人。。

转载请注明来源：attilax的专栏  http://blog.csdn.net/attilax

--Atiend