介绍一个超好用的orm库gorm

Go语言中文网 1周前

以下文章来源于架构师精进 ， 作者章为忠

架构师精进

「架构师精进」会分享各种的技术架构资料，也会将我学到的技技术和知识通过分享给...

为提高开发效率，一般都会使用一些orm框架，把数据库层屏蔽，用户看到的只有对象而无需我们手动做一些转换，这样在使用的时候就非常方便。这种操作方式基本上已经成了标准做法。golang也有很多优秀的orm框架，今天就来介绍介绍gorm。

Gorm的功能

- hook机制(Before/After Create/Save/Update/Delete/Find)
- 对象关系Has One, Has Many, Belongs To, Many To Many, Polymorphism
- 热加载
- 支持原生sql操作
- 事务性
- 链式api
- 支持的数据库有：mysql、postgre、sqlite、sqlserver
- 查询操作

以上是gorm的功能，至于为什么是gorm？gorm 跟其他框架有什么不一样？这里就不在介绍了。直接讲用法吧。

库安装

```
go get -u github.com/jinzhu/gorm
```

数据库连接

```
db, err = gorm.Open("mysql", "root:root@tcp(127.0.0.1:3306)/irisapp?chars
if err != nil {
    panic("连接数据库失败")
```

```
}
```

连接比较简单，直接调用 gorm.Open 传入数据库地址即可。gorm支持基本上所有主流的关系数据库，只是连接方式上略有不同，这里我用的 mysql为例吧。

表定义

```
type Product struct {
    ID        int    `gorm:"primary_key"`
    Code      string `gorm:"type:varchar(20);"`
    Price     int     `gorm:"type:int;"`
    Name      string `gorm:"type:varchar(64);"`
    Mail      string `gorm:"type:varchar(256);"`
    CreatedAt time.Time
}
```

创建表

```
if !db.HasTable(&Like{}) {
    if err := db.Set("gorm:table_options", "ENGINE=InnoDB DEFAULT CHARSET
        panic(err)
    }
}
```

直接通过 db.CreateTable 就可以创建表了，非常方便，还可以通过 db.Set 设置一些额外的表属性

另外，还有自动同步创建表的方法:

```
// 自动迁移模式
db.AutoMigrate(&Product{})
```

查询

```
var product Product
db.First(&product, 1) // 查询id为1的product
db.First(&product, "code = ?", "ik01001") // 查询code为l1212的product
```

插入

```
// 创建
db.Create(&Product{Code: "ik01001", Price: 1000})
```

构造已给对象，直接调用 db.Create() 就可以插入一条记录。不用拼接sql语句，是不是很方便。

更新

```
// 更新 - 更新product的price为2000
db.Model(&product).Update("Price", 2000)
```

删除

简单对象删除：

```
db.Delete(&product)
复杂条件的删除：
if err := db.Where(&Product{ID: 1}).Delete(Product{}).Error; err != nil {
    return err
}
```

事务

```
func CreateProducts(db *gorm.DB) err {
    tx := db.Begin()
    // 注意，一旦你在一个事务中，使用tx作为数据库句柄

    if err := tx.Create(&Product{Code: "ik01003", Price: 3000}).Error; e
        tx.Rollback()
        return err
    }

    tx.Commit()
    return nil
}
```

事务的处理也很简单，用 db.Begin() 声明开启事务，结束的时候调用 tx.Commit()，异常的时候调用 tx.Rollback()

最后

以上就把基本的增删改查介绍完了，实际使用中还有很多高级的用法，比如关联查询，主外键设置等。大家可以看看官方的使用说明：https://gorm.io/zh_CN/docs/

推荐阅读

- GCTT出品|使用 Go、Postgresql、JWT 和 GORM 搭建安全的 REST API

喜欢本文的朋友，欢迎关注“Go语言中文网”：

Go语言中文网启用微信学习交流群，欢迎加微信：274768166