# gin.Context和context.Context

gin.Context是Gin框架中的一个结构体，它封装了HTTP请求和响应的信息，并提供了许多有用的方法来处理HTTP请求。例如，可以通过gin.Context对象获取请求参数、设置响应状态码、设置响应头、渲染模板等。

context.Context是Go语言标准库中的一个接口，它提供了一种跨多个Goroutine传递请求范围数据的机制。在Go语言中，每个Goroutine都有自己的一个上下文，这个上下文中可以存储一些请求相关的数据，例如请求ID、用户信息等。当一个请求被转发到多个Goroutine中进行处理时，可以使用context.Context将这些数据传递给其他的Goroutine。

虽然gin.Context和context.Context是两个不同的概念，但它们之间是有联系的。在Gin框架中，每个请求都会创建一个gin.Context对象，并将其绑定到当前Goroutine的上下文中。因此，在处理HTTP请求时，可以在gin.Context对象中存储一些请求相关的数据，例如请求ID、用户信息等。如果需要跨多个Goroutine传递这些数据，可以将gin.Context对象中的数据复制到context.Context对象中，并将该对象作为参数传递给其他的Goroutine。

使用场景方面，gin.Context主要用于处理HTTP请求和响应，例如获取请求参数、设置响应状态码、渲染模板等。而context.Context则主要用于跨多个Goroutine传递请求相关的数据，例如请求ID、用户信息等。在实际的Web应用程序中，通常需要同时使用这两个上下文。

# gorm Update方法

使用Update方法时需要注意：如果直接使用struct类型(表的映射对象)进行更新时，零值将被忽略。加入字段之前有值，现在需要更新为零值。直接使用struct进行更新是无效的。零值被忽略之后，相当于字段还保留有旧值。

这种情况下，需要将struct转换为map[string]interface{}之后再更新