Atitit 数据库中的多对多关联关系对设计

目录

[1.1. 增加全文索引倒排索引 1](#_Toc4485)

[1.2. 双集合+关联字段法+全文索引（倒排索引） 1](#_Toc24756)

[1.3. 一般需采用中间表的方式处理 2](#_Toc21622)

[1.4. 中间表+view 2](#_Toc8755)

[1.5. 数据更新问题，，实时检查 2](#_Toc5918)

## 增加全文索引倒排索引

也可以增加个json字段 记录

## 双集合+关联字段法+全文索引（倒排索引）

一般不要拆成三个集合，这是传统的关系型数据库的思维方式。都NoSQL了一定要换个角度思考问题。常见的情况就是拆成两个集合，然后有一部分冗余，对最常用的查询做一个索引。这个还得多多实践。【Mars】

1、Team 里面存一个 teammates: [] 存 User 的 \_id 或其它索引值，这样就知道一个 team 有哪些 user   
2、User 可以反查 Team 的 find({teammates: \_id})，这样就知道一个 user 在哪些 team【理想】

## 一般需采用中间表的方式处理

## 中间表+view

View作为大表join 俩边即可。。

## 数据更新问题，，实时检查

****于是第三个想法:****  
我们还是用第二个想法, 只是管理员删除一个标签对象时, 我们不做任何操作, 而是在查看产品的标签数据时, 检查tags里面的标签是否有效(查询Tag表). 那么问题又来了, 查看产品次数远比删除标签的次数多, 岂不是又浪费效率性能?

一般两层关系的数据结构可以设计主从表，通过主表主键进行关联：如通过ID关联主从表Header 和Details 两部分。超过两层关系的数据结构建议只创建一张表，主表数据重复冗余即可。不建议创建多个关系复杂的中间表，表越多后期数据处理就约麻烦，出错率越高

一种是关联性关系，比如赞、感谢、收藏，这种就得靠关系表来维护，当然可以建个通用类型的关系表来容纳这些操作，毕竟都是话题的相关动作。