Atitit 数据存储设计与范式

为什么需要范式

数据库范式为数据库的设计、开发提供了一个可参考的典范，在许多教学材料中也是作为关键的课程内容。

那么范式的提出是为了解决什么问题？

第一范式，要求将列尽可能最小的分割，希望消除某个列存储多个值的冗余的行为

比如用户表中的地址信息，拆分为省、市这种明确的字段，可以按独立的字段检索、查询

第二范式，要求唯一的主键，且不存在对主键的部分依赖，希望消除表中存在冗余(多余)的列

比如订单表中的商品分类、详情信息，只需要由商品信息表存储一份即可。

第三范式，要求没有间接依赖于主键的列，即仍然是希望消除表中冗余的列

比如用户表中不需要存储额外的 其所在城市的人口、城市特点等信息。

## 很明显，这些范式大都是为了消除冗余而提出的，即尽可能的减少存储成本。

————————————————

借助三范式的理念，你可以设计出很精炼的数据库表结构。然而现有的项目应用

# 并不会完全遵循范式的理念，原因在于：

## 性能原因，没有任何冗余的表设计会产生更多的查询行为

，这意味着会产生更多次的数据库IO操作。在一些实时交互的系统中，可能会慢得让人难以忍受。

当然，你可以使用数据库的 连接(join) 操作，而事实上数据库提供 join 也就是为了来缓解这种问题。但一旦用到了分库分表方案的面前，这个问题就会非常的棘手。

## 成本结构的变化，数据库范式是在20世纪提出的，当时的磁盘存储成本还很高。

随着科技发展，数据存储的成本已经大幅度缩减，对于采用范式设计(规避冗余)带来的成本缩减收益已经不那么明显。

————————————————

# 反范式设计

## 既然范式是为了消除冗余，那么反范式就是通过增加冗余、聚合的手段来提升性能。

比如，为了提升查询的性能，在CMS的文章表中同时冗余作者的信息。

当然，除了冗余(存储多份拷贝) 之外，

## 还有另外的理念，即数据的聚合，或者叫嵌套。

这种做法相当于是将多个字段(列)合并存储到数据库表的一个列中。

————————————————

这种灵活的结构几乎是 NoSQL的专利，比如MongoDB文档数据库就可以直接以内嵌数组、对象的形式来实现聚合式存储，这无疑带来了极大的灵活性。

而 MySQL 在5.2.7版本开始支持JSON结构化列，也进入了聚合式存储的队伍，与其对标的PostGreSQL 则是9.4版本就已经支持。

————————————————

一方面，除了能获得性能的提升之外，数据压缩、高度灵活扩展(非结构化) 也是反范式设计能获得青睐的理由。

当然，这里并非一律反对数据库范式，理解范式仍然是做好数据库设计的一门基础，比如选择合适的主键、清晰的划分每一列属性等等。

# ****根据自身的业务特点在范式和反范式中找到平衡点(通常是两者的结合****

在项目中仍然需要****根据自身的业务特点在范式和反范式中找到平衡点(通常是两者的结合)****。类似于架构设计中空间换时间的一些做法，这其中涉及到的各种取舍都是需要经过权衡的。

也可以说这是一门艺术，因为没有标准答案...

## 阿提拉觉得反范式更加好些 28原则