### **前言：实际上设计任何一种数据库应用系统，不论是基于何种数据模型，都会遇到如何构造合适的数据模式即逻辑结构的问题。由于关系模型有严格的数学理论基础，并且可以向别的数据模型转换。所以要设计合适的关系模式，使其逻辑结构更加符合要求，出现了规范化理论。而**[三大范式](https://so.csdn.net/so/search?q=%E4%B8%89%E5%A4%A7%E8%8C%83%E5%BC%8F&spm=1001.2101.3001.7020" \t "https://blog.csdn.net/weixin_52014130/article/details/_blank)**即第一，第二，和第三范式就是规范化理论重要部分，是**为了在设计中更好的解决数据冗余，数据有效性检查，提高存储效率。** 另外还有第四范式，第五范式等。**

## **1.**第一范式：强调的是列的原子性，即数据库表的每一列都是不可分割的原子数据项。

## 2. 第二范式：要求实体的属性完全依赖于主关键字。所谓完全依赖是指不能存在仅依赖主关键字一部分的属性。

## **3.**第三范式：任何非主属性不依赖于其它非主属性。

在实际的开发中需要考虑诸多问题，如：

* 考虑商业化的需求和目标（成本，用户体验），数据库的性能更加重要
* 在规范性能问题的时候，需要适当地考虑一下规范性
* 有时候会故意给某些表增加一些冗余的字段（多表查询→单表查询）
* 有时会故意增加一些计算列（比如当有几百万条数据时，SELECT COUNT(\*)会非常慢，直接增加一个计算列，每次增加一条数据时，这个列自动加一 ）（大数据量→小数据量）

由此可见，实际中并没有去严格遵守这种规范。因为规范和性能不可兼得！规范越高，性能越低。

# **范式（Normal Form,NF）**