<https://blog.csdn.net/dreamrealised/article/details/10474391>

<https://segmentfault.com/a/1190000013695030>

一、第一范式

1NF是对属性的原子性，要求属性具有原子性，不可再分解；

表：字段1、 字段2(字段2.1、字段2.2)、字段3 ......

如学生（学号，姓名，性别，出生年月日），如果认为最后一列还可以再分成（出生年，出生月，出生日），它就不是一范式了，否则就是；

二、第二范式

2NF是对记录的惟一性，要求记录有惟一标识，即实体的惟一性，即不存在部分依赖；

表：学号、课程号、姓名、学分;

这个表明显说明了两个事务:学生信息, 课程信息;由于非主键字段必须依赖主键，这里学分依赖课程号，姓名依赖与学号，所以不符合二范式。

可能会存在问题：

数据冗余:，每条记录都含有相同信息；

删除异常：删除所有学生成绩，就把课程信息全删除了；

插入异常：学生未选课，无法记录进数据库；

更新异常：调整课程学分，所有行都调整。

正确做法:  
学生：Student(学号, 姓名)；  
课程：Course(课程号, 学分)；  
选课关系：StudentCourse(学号, 课程号, 成绩)。

三、第三范式

3NF是对字段的冗余性，要求任何字段不能由其他字段派生出来，它要求字段没有冗余，即不存在传递依赖；

表: 学号, 姓名, 年龄, 学院名称, 学院电话

因为存在依赖传递: (学号) → (学生)→(所在学院) → (学院电话) 。

可能会存在问题：

数据冗余redundancy:有重复值；

更新异常：有重复的冗余信息，修改时需要同时修改多条记录，否则会出现数据不一致的情况 。

正确做法：

学生：(学号, 姓名, 年龄, 所在学院)；

学院：(学院, 电话)。

四、反范式化

一般说来，数据库只需满足第三范式（3NF）就行了。

没有冗余的数据库设计可以做到。但是，没有冗余的数据库未必是最好的数据库，有时为了提高运行效率，就必须降低范式标准，适当保留冗余数据。具体做法是：在概念数据模型设计时遵守第三范式，降低范式标准的工作放到物理数据模型设计时考虑。降低范式就是增加字段，允许冗余，达到以空间换时间的目的。

〖例〗：有一张存放商品的基本表，如表1所示。“金额”这个字段的存在，表明该表的设计不满足第三范式，因为“金额”可以由“单价”乘以“数量”得到，说明“金额”是冗余字段。但是，增加“金额”这个冗余字段，可以提高查询统计的速度，这就是以空间换时间的作法。在Rose 2002中，规定列有两种类型：数据列和计算列。“金额”这样的列被称为“计算列”，而“单价”和“数量”这样的列被称为“数据列”。

**BCNF范式：**

<https://www.2cto.com/database/201404/290140.html>