数据库主要有4种范式，1NF，2NF，3NF，BCNF，按从左至右的顺序一种比一种严格。一般项目的数据库设计达到3NF就可以了。

1NF就是要求一张表里只放相互关联的字段，一个字段只放一条信息，最基本的要求。

1NF，消除非主属性对键的部分函数依赖，--->2NF

2NF，消除非主属性对键的传递函数依赖，--->3NF

3NF，消除主属性对键的部分函数依赖和传递函数依赖,--->BCNF

第一范式1NF：

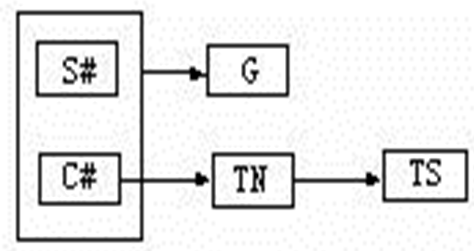
如果每个属性值，都是不可再分的最小数据单位，满足第一范式。

第二范式2NF：

如果关系模式R为1NF，并且R中的每一个非主属性都完全依赖于R的某个候选关键字，则称R是第二范式的，简记为2NF。

【例】

设有关系模式R(学号S#,课程号C#,成绩G,任课教师TN,教师专长TS)，基于R的函数依赖集F={(S#,C#)→G,C#→TN,TN→TS},判断R是否为2NF。



用阿氏推理规则由F可推出：(S#,C#)→{S#,C#,G,TN,TS}，即属性组合(S#,C#)是R的候选关键字(R只有这一个候选键)。(S#,C#)的一个值可惟一标识R中的一个元组(并且没有多余的属性)。

我们可以看到，**非主属性G对键是完全依赖：(S#,C#)→G。但非主属性TN,TS对键是部分依赖(他们仅依赖于键的真子集C#)**。由于R中存在非主属性对候选键的部分依赖，所以**关系模式R不是2NF**

R中存在非主属性对候选键的部分依赖，将会引起数据冗余、数据操作异常等问题。可以把关系R无损联接地分解成两个2NF的关系模式：

ρ={R1,R2}，R1={S#.C#,G},R2={C#,TN,TS}。

第三范式3NF：

如果关系模式R为2NF，并且R中的每一个非主属性都不传递依赖于R的某个候选关键字，则称R是第三范式。

【例】

续上例（*R(学号S#,课程号C#,成绩G,任课教师TN,教师专长TS)*），*判断关系模式R1={S#.C#,G},R2={C#,TN,TS} 是否为3NF。*

解：

(1) 在关系模式R1={S#,C#,G}，候选关键字是(S#,C#)，主属性是S#,C#，非主属性是G，函数依赖为(S#,C#)→G。  由于R1中不存在非主属性对候选关键字的传递依赖，所以关系模式R1是3NF。

(2) 在关系模式R2={C#,TN,TS}，候选关键字是C#，主属性是C#，非主属性是TN,TS，**函数依赖为C#→TN，TN→TS。由于R2中存在非主属性对候选关键字的传递依赖C# TS，所以关系模式R2不是3NF。**

可以把关系**R2无损联接地分解成两个3NF**的关系模式：

ρ={R3,R4}，R3={C#,TN},R4={TN,TS}。

快速理解：请参见

<http://www.php100.com/html/webkaifa/PHP/PHPyingyong/2009/0830/3245.html>