基础概念

实体：现实世界中客观存在并可以被区别的事物。比如“一个学生”、“一本书”、“一门课”等等。值得强　　调的是这里所说的“事物”不仅仅是看得见摸得着的“东西”，它也可以是虚拟的，不如说“老师与学校的关系”。

- 属性：　“实体所具有的某一特性”，由此可见，属性一开始是个逻辑概念，比如说，“性别”是“人”的一个属性。在关系数据库中，属性又是个物理概念，属性可以看作是“表的一列”。
- 元组：表中的一行就是一个元组。
- 分量：元组的某个属性值。在一个关系数据库中，它是一个操作原子，即关系数据库在做任何操作的时候，属性是“不可分的”。否则就不是关系数据库了。
- 码：表中可以唯一确定一个元组的某个属性（或者属性组），如果这样的码有不止一个，那么大家都叫候选码，我们从候选码中挑一个出来做老大，它就叫主码。
- 全码：如果一个码包含了所有的属性，这个码就是全码。
- 主属性：一个属性只要在任何一个候选码中出现过，这个属性就是主属性。
- 非主属性：与上面相反，没有在任何候选码中出现过，这个属性就是非主属性。
- 外码：一个属性（或属性组），它不是码，但是它别的表的码，它就是外码。

数据库范式

1N:关系 R 中的属性都是不可分割的项.

2N:在 1N 的基础上,每个非主属性完全函数依赖于码.

3N:在 2N 的基础上,每一个非主属性既不部分依赖于码也不传递依赖于码.

1N

|　　消除非主属性对码的部分函数依赖

2N

| 消除非主属性对码的传递函数依赖

3N

| 消除主属性对码的部分和传递函数依赖

BCNF

| 消除非平凡且非函数依赖的多值依赖

4N

简单描述：

第三范式的要求如下：

1，每一列只有一个值

2，每一行都能区分。

3，每一个表都不包含其他表已经包含的非主关键字信息。

你说的两个表，如果每个都满足三范式，那么两个表也满足三范式。

数据库的设计范式是数据库设计所需要满足的规范，满足这些规范的数据库是简洁的、结构明晰的，同时，不会发生插入（insert）、删除（delete）和更新（update）操作异常。反之则是乱七八糟，不仅给数据库的编程人员制造麻烦，而且面目可憎，可能存储了大量不需要的冗余信息。

设计范式是不是很难懂呢？非也，大学教材上给我们一堆数学公式我们当然看不懂，也记不住。所以我们很多人就根本不按照范式来设计数据库。

实质上，设计范式用很形象、很简洁的话语就能说清楚，道明白。本文将对范式进行通俗地说明，并以笔者曾经设计的一个简单论坛的数据库为例来讲解怎样将这些范式应用于实际工程。

范式说明

第一范式（1NF）：数据库表中的字段都是单一属性的，不可再分。这个单一属性由基本类型构成，包括整型、实数、字符型、逻辑型、日期型等。

例如，如下的数据库表是符合第一范式：

字段 1	字段 2	字段 3	字段 4

?	?	?	?

而这样的数据库表是不符合第一范式的：

字段 1	字段 2	字段 3		字段 4
?	?	字段 3.1	字段 3.2	?

很显然，在当前的任何关系数据库管理系统（DBMS）中，傻瓜也不可能做出不符合第一范式的数据库，因为这些 DBMS 不允许你把数据库表的一列再分成二列或多列。因此，你想在现有的 DBMS 中设计出不符合第一范式的数据库都是不可能的。

第二范式（2NF）：数据库表中不存在非关键字段对任一候选关键字段的部分函数依赖(部分函数依赖指的是存在组合关键字中的某些字段决定非关键字段的情况），也即所有非关键字段都完全依赖于任意一组候选关键字。

假定选课关系表为 SelectCourse(学号, 姓名, 年龄, 课程名称, 成绩, 学分)，关键字为组合关键字(学号, 课程名称)，因为存在如下决定关系：

(学号, 课程名称) → (姓名, 年龄, 成绩, 学分)

这个数据库表不满足第二范式，因为存在如下决定关系：

(课程名称) → (学分)

(学号) → (姓名, 年龄)

即存在组合关键字中的字段决定非关键字的情况。

由于不符合 2NF，这个选课关系表会存在如下问题：

(1) 数据冗余：

同一门课程由 n 个学生选修，"学分"就重复 n-1 次；同一个学生选修了 m 门课程，姓名和年龄就重复了 m-1 次。

(2) 更新异常：

若调整了某门课程的学分，数据表中所有行的"学分"值都要更新，否则会出现同一门课程学分不同的情况。

(3) 插入异常：

假设要开设一门新的课程，暂时还没有人选修。这样，由于还没有"

学号"关键字，课程名称和学分也无法记录入数据库。

(4) 删除异常：

假设一批学生已经完成课程的选修，这些选修记录就应该从数据库表中删除。但是，与此同时，课程名称和学分信息也被删除了。很显然，这也会导致插入异常。

把选课关系表 SelectCourse 改为如下三个表：

学生：Student(学号, 姓名, 年龄)；

课程：Course(课程名称, 学分)；

选课关系：SelectCourse(学号, 课程名称, 成绩)。

这样的数据库表是符合第二范式的，消除了数据冗余、更新异常、插入异常和删除异常。

另外，所有单关键字的数据库表都符合第二范式，因为不可能存在组合关键字。

第三范式（3NF）：在第二范式的基础上，数据表中如果不存在非关键字段对任一候选关键字段的传递函数依赖则符合第三范式。所谓传递函数依赖，指的是如果存在"A → B → C"的决定关系，则 C 传递函数依赖于 A。因此，满足第三范式的数据库表应该不存在如下依赖关系：

关键字段 → 非关键字段 x → 非关键字段 y

假定学生关系表为 Student(学号, 姓名, 年龄, 所在学院, 学院地点, 学院电话)，关键字为单一关键字"学号"，因为存在如下决定关系：

(学号) → (姓名, 年龄, 所在学院, 学院地点, 学院电话)

这个数据库是符合 2NF 的，但是不符合 3NF，因为存在如下决定关系：

(学号) → (所在学院) → (学院地点, 学院电话)

即存在非关键字段"学院地点"、"学院电话"对关键字段"学号"的传递函数依赖。

它也会存在数据冗余、更新异常、插入异常和删除异常的情况，读者可自行分析得知。

把学生关系表分为如下两个表：

学生：(学号, 姓名, 年龄, 所在学院)；

学院：(学院, 地点, 电话)。

这样的数据库表是符合第三范式的，消除了数据冗余、更新异常、插入异常和删除异常。

鲍依斯-科得范式（BCNF）：在第三范式的基础上，数据库表中如果不存在任何字段对任一候选关键字段的传递函数依赖则符合 BCNF 范式。

假设仓库管理关系表为 StorehouseManage(仓库 ID, 存储物品 ID, 管理员 ID, 数量)，且有一个管理员只在一个仓库工作；一个仓库可以存储多种物品。这个数据库表中存在如下决定关系：

(仓库 ID, 存储物品 ID) →(管理员 ID, 数量)

(管理员 ID, 存储物品 ID) → (仓库 ID, 数量)

所以，(仓库 ID, 存储物品 ID)和(管理员 ID, 存储物品 ID)都是 StorehouseManage 的候选关键字，表中的唯一非关键字段为数量，它是符合第三范式的。但是，由于存在如下决定关系：

(仓库 ID) → (管理员 ID)

(管理员 ID) → (仓库 ID)

即存在关键字段决定关键字段的情况，所以其不符合 BCNF 范式。它会出现如下异常情况：

(1) 删除异常：

当仓库被清空后，所有"存储物品 ID"和"数量"信息被删除的同时，"仓库 ID"和"管理员 ID"信息也被删除了。

(2) 插入异常：

当仓库没有存储任何物品时，无法给仓库分配管理员。

(3) 更新异常：

如果仓库换了管理员，则表中所有行的管理员 ID 都要修改。

把仓库管理关系表分解为二个关系表：

仓库管理：StorehouseManage(仓库 ID, 管理员 ID)；

仓库：Storehouse(仓库 ID, 存储物品 ID, 数量)。

这样的数据库表是符合 BCNF 范式的，消除了删除异常、插入异常和更新异常。

BC 范式的条件有多种等价的表述：每个非平凡依赖的左边必须包含键码；每个决定因素必须包含键码。

第三范式：只检查非主属性部分依赖和传递依赖。
BC 范式： 既检查非主属性，又检查主属性部分依赖和传递依赖。
满足 BC 范式的关系都必然满足第三范式。

还可以这么说：若一个关系达到了第三范式，并且它只有一个候选码，或者它的每个候选码都是单属性，则该关系自然达到 BC 范式。

总结：

侯选关键字：又叫侯选码，惟一标识一行数据，其真子集不能是侯选关键字，一个表可以存在多个侯选关键字，如用户表的 username，userid
主关键字：又叫主键，主码，被选中的用来区分其它行的侯选关键字，一个表只有一个主关键字
部分依赖：(A,B)->C,D,如 A->C,则 C 部分依赖 A
传递依赖：A->B->C,则 C 传递依赖 A

1. 第一范式（1NF）：属性不可拆分 或 无重复的列
数据库的字段是单一属性，不可再分。
这个简单，就是一个属性不允许再分成多个属性来建立列。事实上，在目前的 DBMS 中是不可能拆分属性的，因为他们不允许这么做。
2. 第二范式（2NF）：完全函数依赖
任何非关键字段不能部分依赖任一侯选关键字（即必须完全依赖）。
先讲讲什么是部分函数依赖。
部分函数依赖，就是多个属性决定另一个属性，但事实上，这多个属性是有冗余的。例如，（学号，班级）->姓名，事实上，只需要学号就能决定姓名，因此班级是冗余的，应该去掉。
满足第二范式的数据库设计必须先满足第一范式。
因此第二范式的目标就是消除函数依赖关系中左边存在的冗余属性。
3.第三范式（3NF）：消除传递依赖

任何非关键字段不能传递依赖任一候选关键字
不依赖于其他非主属性（消除传递依赖）。
满足第三范式的数据库必须先满足第二范式。
也就是，数据库中的属性依赖仅能依赖于主属性，不存在于其他非主属性的关联。例如，图书，图书室的关系。图书包括编号、出版商、页码等信息，图书室包括图书室编号、所存图书（外键）。其中，图书室的表中不应该存储任何图书的具体信息（例如，出版商。。），而只能通过主键图书编号来获得对应图书的信息。

4.BC 范式（BCNF）：
（1）所有非主属性对每一个码都是完全函数依赖；
（2）所有的主属性对于每一个不包含它的码，也是完全函数依赖；
（3）没有任何属性完全函数依赖于非码的任意一个组合。
R 属于 3NF，不一定属于 BCNF，如果 R 属于 BCNF，一定属于 3NF。
5.第四范式（4NF）：
对于每一个 X->Y，X 都能找到一个候选码（　若关系中的某一属性组的值能唯一地表示一个元组，而其真子集不行，则称该属性组为候选码）。

1NF 消去对主码的部分函数依赖后=2NF。
2NF 消去对主码的传递函数依赖后=3NF。
3NF 消去对候选码（注意是候选码）的部分函数依赖和传递函数依赖后 = BCNF
bcnf 是对 3nf 的改进，即在 3nf 的基础的又把范围从主码扩大为候选码！

第一范式：1NF 是对属性的原子性约束，要求属性具有原子性，不可再分解；
第二范式：2NF 是对记录的惟一性约束，要求记录有惟一标识，即实体的惟一性；
第三范式：3NF 是对字段冗余性的约束，即任何字段不能由其他字段派生出来，它要求字段没有冗余。