关系规范化理论

逻辑设计问题：针对一个具体问题，应该如何构造一个适合于它的数据模式，即应该构造几个关系模式，每个关系由哪些属性组成等。

关系模型的形式化定义：三元组 R<U,F>

R：R为关系名，它是符号化的元组语义；

U：U为关系的属性集合，一组属性U；

F： F为属性集U的数据依赖集，属性组U上的一组数据依赖F

当且仅当U上的一个关系r满足F时，r称为关系模式R<U,F>的一个关系。

数据依赖是一个关系内部属性与属性之间的一种约束关系。这种约束关系是通过属性间值的相等与否体现出来的数据间相互联系。（函数依赖FD、多值依赖MVD）

数据依赖：通过一个关系中属性间 值得相等与否体现出来的数据间的相互关系。

属性间的这种函数依赖关系类似于数学中的y=f(x)，自变量x确定之后，相应的函数值y也就唯一地确定了。

关系模式可能存在的问题：

教学（学号，姓名，年龄，性别，系名，系主任，课程名，成绩） 码（学号，课程名）

数据冗余大：浪费大量的存储空间 （数据冗余应尽可能少，控制数据冗余）

更新异常：面临数据不一致的危险

插入异常：如果一个系刚成立，尚无学生，就无法把这个系及其系主任的信息存入数据库。

一个新系没有招生时，系名和系主任名无法插入到数据库中

删除异常：如果某个系的学生全部毕业了，在删除该系学生信息的同时，把这个系及其系主任的信息也删掉了。

当一个系的学生都毕业了而又没有招新生时，删除了全部学生记录，随之也删除了系名和系主任名

通常按照属性间依赖情况来区分关系规范化程度。

用函数依赖的概念定义码：

设K为 *R<U*,*F>* 中的属性或属性组，若 （每个属性），则 K 称为 R 的候选码。若候选码多于一个，则选定其中的一个作为主码，通常称之为码。

K满足两个条件：

K完全函数决定该关系的所有其他属性；

K的任何真子集都不能完全函数决定R的所有其他属性，K必须是最小的；

主属性：包含在任何一个候选码中的属性。

非主属性或非码属性：不包含在任何码中的属性。

最简单的情况，单个属性是码。最极端的情况，整个属性组是码，称为全码。

由于码能唯一确定一个元组，所以关系的码函数决定该关系的所有属性；

一个关系中的所有属性都函数依赖于该关系的码。

关系模式R中属性或属性组 X并非R的码，但X是另一个关系模式的码，则称X是R的外部码，也称外码。

主码与外码提供了一个表示关系间联系的手段。

第一范式： 如果一个关系模式R的所有属性都是不可分的基本数据项，则R∈1NF。

（元组的每个分量必须是不可分的数据项。）

（关系中的属性不能是组合属性，必须是基本项。）

第二范式：若R∈1NF，且每一个非主属性完全函数依赖于码，则R∈2NF。

（每一行被码唯一标识）

一个关系模式R不属于2NF，就会产生以下几个问题

关系模式S-L-C（Sno,Sdept,Sloc,Cno,Grade）,码为（Sno,Cno）

1插入异常。未选课的学生不能插入

假如要插入一个学生Sno=S7,Sdep=PHY,Sloc=BLD2,但该生还未选课，即这个学生无Cno，这样的元组就插不进S-L-C中，因为插入元组时必须给定码值，而此时码值的一部分为空，因而学生的固有信息无法插入。

2删除异常。放弃选修一门课时，只选修这一门课的学生将被删除

假定某个学生只选一门课，如S4就选了一门课C3。现在C3这门课他也不选了，那么C3这个数据项就要删除。而C3是主属性，删除了C3 ，整个元组就必须跟着删除，使得S4的其他信息也被删除了.从而造成删除异常，即不应删除的信息也删除了。

3修改复杂。

某个学生从数学系(MA)转到计算机科学系(CS)，这本来只需修改此学生元组中的Sdept分量。但因为关系模式S-L-C中还含有系的住处Sloc属性，学生转系将同时改变住处，因而还必须修改元组中的Sloc分量。另外，如果这个学生选修了k门课，Sdept，Sloc重复存储了k次，不仅布储冗余度大，而且必须无遗漏地修改k个元组中全部Sdept，Sloc信息，造成修改的复杂化。

第三范式：若R∈3NF，则每一个非主属性既不部分函数依赖于码，也不传递函数依赖于码。

（要求一个数据库表中不能包含其他表中已包含的非码信息）

一个关系模式R若不是3NF，就会产生与2NF相类似的问题。

所谓函数依赖，是指关系中属性或属性组的值可以决定其他属性的值。

当存在部分函数依赖时，就会产生数据冗余。

**关系数据库设计理论：关系规范化理论 + 关系模式分解方法**

范式（Normal Form）是指规范化的关系模式。

数据模型：关系模型

数据模式：有了数据模式，才能构造复杂的数据结构来建立数据之间的内在联系与复杂关系，从而构成数据的全局结构模式。

先有数据模型，才能据其讨论相应数据模式，有了数据模式，就能依据该模式得到相应的实例。

关系模式：关系名(属性1，属性2，……，属性n)

关系模型：用二维表的形式表示实体和实体间联系的数据模型。

数据库设计要按“一事一做”的原则：一个关系描述一个实体或实体间的一种关系。

数据冗余：重复的数据多；

更新异常：有些被修改，有些没有修改；

插入异常：希望插入的信息由于不能满足数据完整性的某种要求，而不能正常地被插入到数据库的问题；

删除异常：在删除某种信息的同时，把其他信息也删除了；

函数依赖反映了同一关系中属性间一一对应的约束。

函数依赖是属性或属性之间一一对应的关系。

数据依赖是同一关系中属性间的相互依赖和相互制约。

1NF：如果关系模式R，其所有的属性均为简单属性，即每个属性都是不可再分的，则称R属于第一范式。

2NF：若R∈1NF，且每一个非主属性完全依赖于码，则R∈2NF。

3NF：若R∈3NF，则每一个非主属性既不部分函数依赖于码，也不传递函数依赖于码。

BCNF：关系模式*R<U*,*F>* 中，若每一个决定因素都包含码，则*R<U*,*F>* ∈BCNF。

（所有的主属性对每一个不包含它的码都是完全函数依赖）

（没有任何属性完全函数依赖于非码的任何一组属性）

如果R属于BCNF，由于R排除了任何属性对码的传递依赖与部分依赖，所以R一定属于3NF。但是，若R∈3NF，则R未必属于BCNF。

3NF只强调非主属性对码的完全直接依赖，这样可能出现主属性对码的部分依赖和传递依赖。

死锁：

如果事务T1和T2都需要数据R1和R2，他们在操作时：T1封锁了数据R1，T2封锁了数据R2；然后T1又请求封锁R2，T2又请求封锁R1；因T2已封锁R2，故T1等待T2释放R2上的锁，同理，因T1已封锁R1，故T2等待T1释放R1上的锁；由于T1和T2都没有获得全部必要的数据，所以他们都不会结束，只能继续等待。这样就出现了T1 在等待T2，而T2又在等待T1的局面，T1和T2两个事务水远不能结束，形成死锁。