著作权归https://www.pdai.tech所有。 链接：<https://www.pdai.tech/md/db/sql/sql-db-theory-concept.html>

**关系型数据库设计理论**

在上文了解数据库如何工作后，本节介绍如何将一个**关系模型**（基于表的数据模型）合理的转化为**数据表**和**关系表**，以及确定**主外健**的。这便是数据库设计理论基础，包括术语，函数依赖，范式等理论基础。@pdai

* [SQL DB - 关系型数据库设计理论](#sql-db---%e5%85%b3%e7%b3%bb%e5%9e%8b%e6)
  + [重要的术语](#%e9%87%8d%e8%a6%81%e7%9a%84%e6%9c%af%e8)
  + [函数依赖](#%e5%87%bd%e6%95%b0%e4%be%9d%e8%b5%96)
  + [异常](#%e5%bc%82%e5%b8%b8)
  + [范式](#%e8%8c%83%e5%bc%8f)
    - [1. 第一范式 (1NF)](#1-%e7%ac%ac%e4%b8%80%e8%8c%83%e5%bc%8f-)
    - [2. 第二范式 (2NF)](#2-%e7%ac%ac%e4%ba%8c%e8%8c%83%e5%bc%8f-)
    - [3. 第三范式 (3NF)](#3-%e7%ac%ac%e4%b8%89%e8%8c%83%e5%bc%8f-)
  + [参考文章](#%e5%8f%82%e8%80%83%e6%96%87%e7%ab%a0)

[**¶**](#重要的术语) **重要的术语**

关系模型是一种基于表的数据模型，以下为关系学生信息，该表有很多不足之处，本文研究内容就是如何改进它：

下面是一些重要术语：

* **属性（attribute）**：列的名字，上图有学号、姓名、班级、兴趣爱好、班主任、课程、授课主任、分数。
* **依赖（relation）**：列属性间存在的某种联系。
* **元组（tuple）**：每一个行，如第二行 （1301，小明，13班，篮球，王老师，英语，赵英，70） 就是一个元组
* **表（table）**：由多个属性，以及众多元组所表示的各个实例组成。
* **模式（schema）**：这里我们指逻辑结构，如 学生信息（学号，姓名，班级，兴趣爱好，班主任，课程，授课主任，分数） 的笼统表述。
* **域（domain）**：数据类型，如string、integer等，上图中每一个属性都有它的数据类型（即域）。
* **键（key）**：由关系的一个或多个属性组成，任意两个键相同的元组，所有属性都相同。需要保证表示键的属性最少。一个关系可以存在好几种键，工程中一般从这些候选键中选出一个作为**主键（primary key）**。
* **候选键（prime attribute）**：由关系的一个或多个属性组成，候选键都具备键的特征，都有资格成为主键。
* **超键（super key）**：包含键的属性集合，无需保证属性集的最小化。每个键也是超键。可以认为是**键的超集**。
* **外键（foreign key）**：如果某一个关系A中的一个（组）属性是另一个关系B的键，则该（组）属性在A中称为外键。
* **主属性（prime attribute）**：所有候选键所包含的属性都是主属性。
* **投影（projection）**：选取特定的列，如将关系学生信息投影为学号、姓名即得到上表中仅包含学号、姓名的列
* **选择（selection）**：按照一定条件选取特定元组，如选择上表中分数>80的元组。
* **笛卡儿积（交叉连接Cross join）**：第一个关系每一行分别与第二个关系的每一行组合。
* **自然连接（natural join）**：第一个关系中每一行与第二个关系的每一行进行匹配，如果得到有交叉部分则合并，若无交叉部分则舍弃。
* **连接（theta join）**：即加上约束条件的笛卡儿积，先得到笛卡儿积，然后根据约束条件删除不满足的元组。
* **外连接（outer join）**：执行自然连接后，将舍弃的部分也加入，并且匹配失败处的属性用NULL代替。
* **除法运算（division）**：关系R除以关系S的结果为T，则T包含所有在R但不在S中的属性，且T的元组与S的元组的所有组合在R中。

[**¶**](#函数依赖) **函数依赖**

通过函数依赖关系，来帮助你确定表中的合理主外健等；这里只是简介，有这么个概念就可以了，因为大多数情况你不用那些所谓的推倒关系，你也是可以凭借**直觉**设计出来的。

记 A->B 表示 A 函数决定 B，也可以说 B 函数依赖于 A。

如果 {A1，A2，... ，An} 是关系的一个或多个属性的集合，该集合函数决定了关系的其它所有属性并且是最小的，那么该集合就称为键码。

对于 A->B，如果能找到 A 的真子集 A'，使得 A'-> B，那么 A->B 就是部分函数依赖，否则就是完全函数依赖。

对于 A->B，B->C，则 A->C 是一个传递函数依赖。

[**¶**](#异常) **异常**

介绍 不符合范式的关系，会产生很多异常，为了引出**范式**的内容。

以下的学生课程关系的函数依赖为 Sno, Cname -> Sname, Sdept, Mname, Grade，键码为 {Sno, Cname}。也就是说，确定学生和课程之后，就能确定其它信息。

| **Sno** | **Sname** | **Sdept** | **Mname** | **Cname** | **Grade** |
| --- | --- | --- | --- | --- | --- |
| 1 | 学生-1 | 学院-1 | 院长-1 | 课程-1 | 90 |
| 2 | 学生-2 | 学院-2 | 院长-2 | 课程-2 | 80 |
| 2 | 学生-2 | 学院-2 | 院长-2 | 课程-1 | 100 |
| 3 | 学生-3 | 学院-2 | 院长-2 | 课程-2 | 95 |

不符合范式的关系，会产生很多异常，主要有以下四种异常:

* 冗余数据: 例如 学生-2 出现了两次。
* 修改异常: 修改了一个记录中的信息，但是另一个记录中相同的信息却没有被修改。
* 删除异常: 删除一个信息，那么也会丢失其它信息。例如删除了 课程-1 需要删除第一行和第三行，那么 学生-1 的信息就会丢失。
* 插入异常: 例如想要插入一个学生的信息，如果这个学生还没选课，那么就无法插入。

[**¶**](#范式) **范式**

范式理论是为了解决以上提到四种异常。

高级别范式的依赖于低级别的范式，1NF 是最低级别的范式。

[**¶**](#1-第一范式-1nf) **1. 第一范式 (1NF)**

属性不可分。

[**¶**](#2-第二范式-2nf) **2. 第二范式 (2NF)**

每个非主属性完全函数依赖于键码。

可以通过分解来满足。

**分解前**

| **Sno** | **Sname** | **Sdept** | **Mname** | **Cname** | **Grade** |
| --- | --- | --- | --- | --- | --- |
| 1 | 学生-1 | 学院-1 | 院长-1 | 课程-1 | 90 |
| 2 | 学生-2 | 学院-2 | 院长-2 | 课程-2 | 80 |
| 2 | 学生-2 | 学院-2 | 院长-2 | 课程-1 | 100 |
| 3 | 学生-3 | 学院-2 | 院长-2 | 课程-2 | 95 |

以上学生课程关系中，{Sno, Cname} 为键码，有如下函数依赖:

* Sno -> Sname, Sdept
* Sdept -> Mname
* Sno, Cname-> Grade

Grade 完全函数依赖于键码，它没有任何冗余数据，每个学生的每门课都有特定的成绩。

Sname, Sdept 和 Mname 都部分依赖于键码，当一个学生选修了多门课时，这些数据就会出现多次，造成大量冗余数据。

**分解后**

关系-1

| **Sno** | **Sname** | **Sdept** | **Mname** |
| --- | --- | --- | --- |
| 1 | 学生-1 | 学院-1 | 院长-1 |
| 2 | 学生-2 | 学院-2 | 院长-2 |
| 3 | 学生-3 | 学院-2 | 院长-2 |

有以下函数依赖:

* Sno -> Sname, Sdept
* Sdept -> Mname

关系-2

| **Sno** | **Cname** | **Grade** |
| --- | --- | --- |
| 1 | 课程-1 | 90 |
| 2 | 课程-2 | 80 |
| 2 | 课程-1 | 100 |
| 3 | 课程-2 | 95 |

有以下函数依赖:

* Sno, Cname -> Grade

[**¶**](#3-第三范式-3nf) **3. 第三范式 (3NF)**

非主属性不传递函数依赖于键码。

上面的 关系-1 中存在以下传递函数依赖:

* Sno -> Sdept -> Mname

可以进行以下分解:

关系-11

| **Sno** | **Sname** | **Sdept** |
| --- | --- | --- |
| 1 | 学生-1 | 学院-1 |
| 2 | 学生-2 | 学院-2 |
| 3 | 学生-3 | 学院-2 |

关系-12

| **Sdept** | **Mname** |
| --- | --- |
| 学院-1 | 院长-1 |
| 学院-2 | 院长-2 |