**5 逻辑数据库**

**5.1由概念数据库模式生成初始关系数据库模式的方法**

在数据库设计中，从概念模型（如 E-R 图）生成初始关系数据库模式是关键步骤。概念模型更注重数据及其关系的抽象表示，而关系模式是可直接实现的表结构。以下是将概念数据库模式转化为关系数据库模式的具体方法。

**1. 普通实体集的变换**

**规则**

* 对于每个普通实体集，将其转换为一个关系（表）。
* 实体的每个属性成为关系的列。
* 实体的主键属性成为关系的主键。

**示例**

**实体集**：学生(Student) 包含属性 学号(SID)、姓名(Name)、年龄(Age)。

* 转化后的关系模式：

**Student(SID, Name, Age)**

其中，SID 是主键。

**2. 弱实体集的变换**

**规则**

* 弱实体集依赖于强实体集，其转化规则如下：
  1. 创建一个关系（表），包含弱实体集的所有属性。
  2. 添加依赖的强实体集的主键属性到该关系中。
  3. 将弱实体的主键设置为其本地主键和强实体主键的组合。

**示例**

**弱实体**：订单明细(OrderDetail) 依赖于强实体订单(Order)。

* 属性：
  + **Order 主键：OrderID**
  + OrderDetail 属性：DetailID、Quantity
* 转化后的关系模式：

**OrderDetail(OrderID, DetailID, Quantity)**

其中，(OrderID, DetailID) 是复合主键。

**3. 多值属性的变换**

**规则**

* 将多值属性拆分为单独的关系。
* 该关系包含：
  1. 原实体的主键属性。
  2. 多值属性作为新关系的普通属性。
  3. 新关系的主键是实体主键和多值属性的组合。

**示例**

**实体**：学生(Student)，其电话(Phone) 是多值属性。

* 转化后的关系模式：

Student(SID, Name, Age)

StudentPhone(SID, Phone)

其中：

* + SID 是 StudentPhone 的主键，Phone 是普通属性。

**4. 实体间联系的变换**

**规则**

1. **一对一 (1:1) 联系**：
   * 可将关系中一个实体的主键作为外键添加到另一个实体对应的关系中。
   * 或者创建一个新的关系来表示联系，包含两实体的主键。

**示例**： 联系：国家(Nation) 和 首都(Capital) 一一对应。

* + 转化方式一（外键）：

**Nation(NationID, Name, CapitalID)**

**Capital(CapitalID, Name)**

* + 转化方式二（新关系）：

Nation(NationID, Name)

Capital(CapitalID, Name)

NationCapital(NationID, CapitalID)

1. **一对多 (1:N) 联系**：
   * 在 "多" 一端的实体对应关系中添加 "一" 一端的主键作为外键。

**示例**： 联系：学生(Student) 和 课程(Course)，一个学生可选修多门课程。

* + 转化后的关系模式：

Student(SID, Name, Age)

Course(CID, CName, SID)

其中，SID 是外键。

1. **多对多 (M:N) 联系**：
   * 创建一个新关系表示联系，包含两实体的主键及其联系的其他属性。

**示例**： 联系：学生(Student) 和 课程(Course) 的选课关系。

* + 转化后的关系模式：

**Student(SID, Name, Age)**

**Course(CID, CName)**

**Enroll(SID, CID, Grade)**

其中，(SID, CID) 是 Enroll 的复合主键。

**5. 确定函数依赖集**

**定义**

* 函数依赖表示属性之间的关系，定义哪些属性由其他属性唯一决定。
* 函数依赖是规范化和消除数据冗余的重要基础。

**规则**

* 识别每个关系中的函数依赖集。
* 以主键为基准，确保每个非主键属性函数依赖于主键。
* 在发现部分依赖或传递依赖时，通过分解关系模式消除冗余。

**示例**

关系 Student(SID, Name, Major, Advisor)：

* 函数依赖：
  + SID → Name, Major, Advisor （学号决定学生信息）。
  + Major → Advisor （专业决定导师）。
* 如果有冗余，则分解为两张表：

Student(SID, Name, Major)

MajorAdvisor(Major, Advisor)

**总结**

1. **普通实体集**：转化为关系，属性为列，主键保持。
2. **弱实体集**：包含依赖强实体的主键，并将组合键作为主键。
3. **多值属性**：拆分为单独关系，主键包括实体主键和多值属性。
4. **实体间联系**：根据联系的类型（1:1、1:N、M:N），选择适当的方式转化。
5. **函数依赖**：识别依赖关系，分解不符合规范化要求的模式。

**函数依赖的公理系统：Armstrong 公理**

**1. 为什么需要函数依赖的公理系统**

在关系数据库规范化中，**函数依赖**描述了属性之间的约束关系。仅知道给定的函数依赖集合是不够的，还需要能够推导出这个集合所隐含的所有函数依赖。

* **目标**：通过已知的函数依赖集合 FFF，推导出 FFF 所蕴涵的所有函数依赖。
* **工具**：Armstrong 公理系统是一个**有效且完备**的公理系统，可以用于推导函数依赖的所有蕴涵关系。

**2. Armstrong 公理系统的三条基本规则**

Armstrong 公理系统包括以下三条**推导规则**：

**1. 自反性 (Reflexivity)**

* **描述**：如果 Y⊆X，则 X→Y。
* **含义**：一个属性集总能决定它自己的子集。
* **示例**：
  + X={A,B}，Y={A}，则 {A,B}→{A}。

**2. 增广性 (Augmentation)**

* **描述**：如果 X→Y，则 XZ→YZ（其中 Z 是任意属性集）。
* **含义**：可以对函数依赖的两边同时添加相同的属性集 Z，依赖关系依然成立。
* **示例**：
  + {A}→{B}，则 {A,C}→{B,C}。

**3. 传递性 (Transitivity)**

* **描述**：如果 X→Y且 Y→Z，则 X→Z。
* **含义**：如果 X可以决定Y，而 Y又可以决定 Z，则 X必然可以决定Z。
* **示例**：
  + {A}→{B}，{B}→{C}，则 {A}→{C}。

**3. Armstrong 公理的完备性**

Armstrong 公理是**完备的**：

* **完备性**：通过这三条规则，可以推导出一个给定函数依赖集合所蕴涵的所有函数依赖。
* **有效性**：仅能推导出集合 FFF 所实际蕴涵的函数依赖。