

# 数据库原理

## The Theory of Database System

### 第二章 关系数据库



中国矿业大学计算机学院



中国矿业大学数据库原理精品课程

# 第二章 关系数据库

## 2.1 关系

## 2.2 关系代数

## 2.3 查询优化

## 2.4 关系系统

## 2.5 关系演算



# 2.1 关系

## 2.1.1 关系定义

## 2.1.2 关系操作

## 2.1.3 关系完整性约束

## 2.1.4 关系模式



## 2.1.1 关系定义

- 1. 域 (Domain)
- 2. 笛卡尔积 (Cartesian Product)
- 3. 关系 (Relation)
- 4. 码 (Key)



# 1. 域 (Domain)

➤ 域是一组具有相同数据类型的值的集合。  
例：

- 整数
- 实数
- 介于某个取值范围的整数
- 长度指定长度的字符串集合
- {‘男’， ‘女’ }
- 介于某个取值范围的日期



## 2. 笛卡尔积 (Cartesian Product)

### ➤ 1) 笛卡尔积

给定一组域  $D_1, D_2, \dots, D_n$ , 这些域中可以有相同的。  $D_1, D_2, \dots, D_n$  的笛卡尔积为:

$$D_1 \times D_2 \times \dots \times D_n = \{ (d_1, d_2, \dots, d_n) \mid d_i \in D_i, i=1, 2, \dots, n \}$$

➤ 所有域的所有取值的一个组合

➤ 不能重复



# 笛卡尔积 (续)

例 给出三个域:

$D_1 = \{\text{孙悟空, 宋江, 林黛玉}\}$

$D_2 = \{\text{男, 女}\}$

$D_3 = \{\text{西游记, 水浒传, 红楼梦}\}$

则  $D_1, D_2, D_3$  的笛卡尔积为:

$D_1 \times D_2 \times D_3 = \{(\text{孙悟空, 男, 西游记}), (\text{宋江, 男, 西游记}), (\text{林黛玉, 男, 西游记}), (\text{孙悟空, 女, 西游记}), (\text{宋江, 女, 西游记}), (\text{林黛玉, 女, 西游记}), (\text{孙悟空, 男, 水浒传}), (\text{宋江, 男, 水浒传}), (\text{林黛玉, 男, 水浒传}), (\text{孙悟空, 女, 水浒传}), (\text{宋江, 女, 水浒传}), (\text{林黛玉, 女, 水浒传}), (\text{孙悟空, 男, 红楼梦}), (\text{宋江, 男, 红楼梦}), (\text{林黛玉, 男, 红楼梦}), (\text{孙悟空, 女, 红楼梦}), (\text{宋江, 女, 红楼梦}), (\text{林黛玉, 女, 红楼梦})\}$



# 笛卡尔积（续）

## ➤ 2) 元组 (Tuple)

- 笛卡尔积中每一个元素  $(d_1, d_2, \dots, d_n)$  叫作一个  $n$  元组 (n-tuple) 或简称元组。

## ➤ 3) 分量 (Component)

- 笛卡尔积元素  $(d_1, d_2, \dots, d_n)$  中的每一个值  $d_i$  叫作一个分量。





# 笛卡尔积（续）

## ➤ 4) 基数（Cardinal number）

若 $D_i$  ( $i=1, 2, \dots, n$ ) 为有限集，其基数为 $m_i$  ( $i=1, 2, \dots, n$ )，则 $D_1 \times D_2 \times \dots \times D_n$ 的基数 $M$ 为：

$$M = \prod_{i=1}^n m_i$$

在上例中，基数： $3 \times 2 \times 3 = 18$ ，即  
 $D_1 \times D_2 \times D_3$ 共有 $3 \times 2 \times 3 = 18$ 个元组。



# 笛卡尔积（续）

## ➤ 5) 笛卡尔积的表示方法

- 笛卡尔积可表示为一个二维表。表中的每行对应一个元组，表中的每列对应一个域。



# 3. 关系 (Relation)

## 1) 关系

$D_1 \times D_2 \times \dots \times D_n$  的子集叫作在域  $D_1, D_2, \dots, D_n$  上的关系, 表示为

$$R(D_1, D_2, \dots, D_n)$$

$R$ : 关系名

$n$ : 关系的目或度 (Degree)



# 关系（续）

注意：

关系是笛卡尔积的有限子集。无限关系在数据库系统中是无意义的。

小说名	人物名	性 别
西游记	孙悟空	男
水浒传	宋江	男
红楼梦	林黛玉	女



# 关系（续）

## 2) 元组

关系中的每个元素是关系中的元组，通常用 $t$ 表示。

## 3) 单元关系与二元关系

当 $n=1$ 时，称该关系为单元关系（Unary relation）。

当 $n=2$ 时，称该关系为二元关系（Binary relation）。



# 关系（续）

## 4) 关系的表示

关系也是一个二维表，表的每行对应一个元组，表的每列对应一个域。

表 2.2 SAP 关系

SUPERVISOR	SPECIALITY	POSTGRADUATE
张清玫	信息专业	李勇
张清玫	信息专业	刘晨
刘逸	信息专业	王敏



# 关系（续）

## 5) 属性

关系中不同列可以对应相同的域，为了加以区分，必须对每列起一个名字，称为属性（Attribute）。

$n$ 目关系必有 $n$ 个属性。



# 关系（续）

## 6) 三类关系

### 基本关系（基本表或基表）

实际存在的表，是实际存储数据的逻辑表示

### 查询表

查询结果对应的表

### 视图表

由基本表或其他视图表导出的表，是虚表，不对应实际存储的数据





# 7) 基本关系的性质

- 关系是一个二维表，表的每一行对应一个元组，表的每一列有一个属性名且对应一个域。
- 列是同质的，即每一列的值来自同一域。每列的属性名是不同的。
- 关系所有域都应是原子数据的集合。
- 关系中任意两个元组不能完全相同。
- 关系中行的排列顺序、列的排列顺序是无关紧要的。
- 每个关系都有称之为关键字的属性集唯一标识各元组。



# 关系的性质

- 1) 不能有完全相同的两列，列的顺序无关
- 2) 不能有完全相同的两行，行的顺序无关
- 3) 分量必须取原子值



## 4. 码 (Key)

### ➤ 候选码 (Candidate key)

- 若关系中的某一**最小属性组**的值能**唯一地标识**一个元组，则称该属性组为候选码。
- 在最极端的情况下，关系模式的所有属性组是这个关系模式的候选码，称为全码 (All-key)



# 码（续）

## ➤ 主码（Primary Key）

- 若一个关系有多个候选码，则选定其中一个为主码。
- 包含在任何一个候选码中的属性称为主属性（Prime attribute）。
- 不包含在任何侯选码中的属性称为非主属性（Non-key attribute）



# 码（续）

## ➤ 外码（Foreign Key）

- 设 $F$ 是基本关系 $R$ 的一个或一组属性，但不是 $R$ 的码。 $K_s$ 是基本关系 $S$ 的主码。如果 $F$ 与 $K_s$ 相对应，则称 $F$ 是 $R$ 的外码。
- 并称基本关系 $R$ 为参照关系（Referencing Relation），基本关系 $S$ 为被参照关系（Referenced Relationship）。



# 例如：学生关系和专业关系

学生 (学生编号, 姓名, 性别, 年龄, 专业编号, 身份证号码)

专业 (专业编号, 专业名称, 专业负责人)



# 外码(续)

## 说明

- 关系  $R$  和  $S$  不一定是不同的关系
- 外码并不一定要与相应的主码同名  
当外码与相应的主码属于不同关系时，往往取相同的名字，以便于识别
- 目标关系  $S$  的主码  $K_s$  和参照关系的外码  $F$  必须定义在同一个（或一组）域上

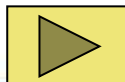


# 关系间的引用(续)

## 例1 学生实体及其内部的领导联系(一对多)

学生（学号，姓名，性别，专业号，年龄，班长）

学号	姓名	性别	专业号	年龄	班长
801	张三	女	01	19	802
802	李四	男	01	19	802
803	王五	男		20	
804	赵六	女	02	20	805
805	钱七	男	02	19	805





# 关系间的引用(续)

例2 学生实体、专业实体以及专业与学生间的一对多联系

学生（学号，姓名，性别，专业，年龄）

专业（专业号，专业名）

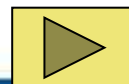


## 学生（学号，姓名，性别，专业，年龄）

学号	姓名	性别	专业	年龄
801	张三	女	01	19
802	李四	男	01	20
803	王五	男		20
804	赵六	女	02	20
805	钱七	男	02	19

## 专业（专业号，专业名）

专业号	专业名
01	信息
02	数学
03	计算机



# 关系间的引用(续)

## 例3 学生、课程、学生与课程之间的多对多联系

学生（学号，姓名，性别，专业号，年龄）

课程（课程号，课程名，学分）

选修（学号，课程号，成绩）



## 学生

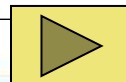
学号	姓名	性别	专业号	年龄
801	张三	女	01	19
802	李四	男	01	20
803	王五	男	01	20
804	赵六	女	02	20
805	钱七	男	02	19

## 课程

课程号	课程名	学分
01	数据库	4
02	数据结构	4
03	编译	4
04	PASCAL	2

## 学生选课

学号	课程号	成绩
801	04	92
801	03	78
801	02	85
802	03	82
802	04	90
803	04	88



# 关系间的引用(续)

在关系数据库中，表与表的联系就是通过**公共属性**实现的，这个公共属性是一个表的主码和另外一个表的外码。



# 关系间的引用(续)

教师（工号，姓名，学院，系部，职称）

课程（课程号，课程名，学分，学时）

开课（工号，课程号，学年，学期）

教室（编号，名称，容纳人数，是否多媒体）

排课（？）

学生（学号，姓名，学院，专业）

选课（？）



# 2.1 关系

2.1.1 关系定义

2.1.2 关系操作

2.1.3 关系完整性约束

2.1.4 关系模式



## 2.1.2 关系操作

### ➤ 1) 常用的关系操作

#### – 查询

- 选择、投影、连接、除、并、交、差

#### – 数据更新

- 插入、删除、修改

#### – 查询的表达能力是其中最主要的部分





# 关系操作（续）

## ➤ 2) 关系数据语言的种类

### – 关系代数语言

- 用对关系的运算来表达查询要求
- 典型代表：ISBL



# 关系操作（续）

- 关系演算语言：用谓词来表达查询要求
  - 元组关系演算语言
    - 谓词变元的基本对象是元组变量
    - 典型代表：APLHA, QUEL
  - 域关系演算语言
    - 谓词变元的基本对象是域变量
    - 典型代表：QBE
- 具有关系代数和关系演算双重特点的语言
  - 典型代表：SQL



# 2.1 关系

2.1.1 关系定义

2.1.2 关系操作

2.1.3 关系完整性约束

2.1.4 关系模式



## 2.1.3 关系完整性约束

### ➤ 实体完整性

- 通常由关系系统自动支持

### ➤ 参照完整性

- 早期系统不支持，目前大型系统能自动支持

### ➤ 用户定义的完整性

- 反映应用领域需要遵循的约束条件，体现了具体领域中的语义约束
- 用户定义后由系统支持



# 关系完整性约束（续）

- 实体完整性和参照完整性是关系模型必须满足的完整性约束条件，被称作是关系的两个不变性，应该由关系系统自动支持。



# 关系完整性约束（续）

- 1、实体完整性
- 2、参照完整性
- 3、用户定义的完整性



# 1、实体完整性

## 实体完整性规则（Entity Integrity）

若属性 $A$ 是基本关系 $R$ 的主属性，则属性 $A$ 不能取空值。

例：学生(学号，姓名，性别，出生年月，籍贯)

“学号” 为主属性，则其不能取空值

考虑：假设学生不会重名呢？



# 实体完整性(续)

关系模型必须遵守实体完整性规则的原因

- (1) 实体完整性规则是针对基本关系而言的。一个基本表通常对应现实世界的一个实体集或多对多联系。
- (2) 现实世界中的实体和实体间的联系都是可区分的，即它们具有某种唯一性标识。
- (3) 相应地，关系模型中以主码作为唯一性标识。





# 实体完整性(续)

## 关系模型必须遵守实体完整性规则的原因(续)

### (4) 主属性不能取空值。

空值就是“不知道”或“无意义”的值。

主属性取空值，就说明存在某个不可标识的实体，即存在不可区分的实体，这与第（2）点相矛盾，因此这个规则称为实体完整性。



# 实体完整性(续)

## 注意

实体完整性规则规定基本关系的**所有主属性**都不能取空值。

例：选修（学号，课程号，成绩）

“学号、课程号”为主码，则两个属性都不能取空值。



# 关系完整性约束（续）

- 1、实体完整性
- 2、参照完整性
- 3、用户定义的完整性



## 2、参照完整性

- (1) 关系间的引用
- (2) 外码
- (3) 参照完整性规则



# (1) 关系间的引用

- 在关系模型中实体及实体间的联系都是用关系来描述的，因此可能存在着关系与关系间的引用。
- 外码体现了关系与关系之间的联系。



## (2) 参照完整性规则

若属性（或属性组） $F$ 是基本关系 $R$ 的外码，它与基本关系 $S$ 的主码 $K_S$ 相对应（基本关系 $R$ 和 $S$ 不一定是不同的关系），则对于 $R$ 中每个元组在 $F$ 上的值必须为：

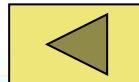
- 或者取空值（ $F$ 的每个属性值均为空值）
- 或者等于 $S$ 中某个元组的主码值。



# 参照完整性规则(续)

学生关系中每个元组的“专业号”属性只取下面两类值：

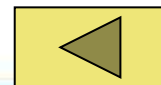
- (1) 空值，表示尚未给该学生分配专业
- (2) 非空值，这时该值必须是专业关系中某个元组的“专业号”值，表示该学生不可能分配到一个不存在的专业中



# 参照完整性规则(续)

选修（学号，课程号，成绩）

“学号”和“课程号”是选修关系中的主属性  
按照实体完整性和参照完整性规则，它们  
只能取相应被参照关系中已经存在的主码值



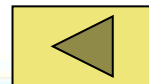


# 参照完整性规则(续)

学生（学号，姓名，性别，专业号，年龄，班长）

“班长”属性值可以取两类值：

- （1）空值，表示该学生所在班级尚未选出班长；
- （2）非空值，这时该值必须是本关系中某个元组的学号值



# 关系完整性约束（续）

- 1、实体完整性
- 2、参照完整性
- 3、用户定义的完整性



### 3、用户定义的完整性

- 用户定义的完整性是针对某一具体关系数据库的约束条件，反映某一具体应用所涉及的数据必须满足的语义要求。
- 关系模型应提供定义和检验这类完整性的机制，以便使用统一的系统的方法处理它们，而不要由应用程序承担这一功能。



# 用户定义的完整性(续)

例:

课程(课程号, 课程名, 学分)

- “课程名” 属性必须取唯一值
- 非主属性 “课程名” 也不能取空值
- “学分” 属性只能取值{1, 2, 3, 4}



# 2.1 关系

2.1.1 关系定义

2.1.2 关系操作

2.1.3 关系完整性约束

2.1.4 关系模式



## 2.1.4 关系模式

1. 什么是关系模式
2. 定义关系模式
3. 关系模式与关系



# 1. 什么是关系模式

关系模式（Relation Schema）是型  
关系是值

关系模式是对关系的描述

- 元组集合的结构

  - 属性构成

  - 属性来自的域

  - 属性与域之间的映象关系

- 元组语义以及完整性约束条件

- 属性间的数据依赖关系集合



## 2. 定义关系模式

关系模式可以形式化地表示为：

$R(U, D, \text{dom}, I, F)$

$R$  关系名

$U$  组成该关系的属性名集合

$D$  属性组  $U$  中属性所来自的域

$\text{dom}$  属性向域的映象集合

$I$  一组完整性约束条件

$F$  属性间的数据依赖关系集合





# 定义关系模式 (续)

关系模式通常可以简记为

$R(U, F)$  或  $R(A_1, A_2, \dots, A_n)$

$R$  关系名

$A_1, A_2, \dots, A_n$  属性名

注：域名及属性向域的映象常常直接说明为  
属性的类型、长度



# 3. 关系模式与关系

## ➤ 关系模式

对关系的描述

静态的、稳定的

## ➤ 关系

关系模式在某一时刻的状态或内容  
动态的、随时间不断变化的



# 小结

## ➤ 关系数据结构

### — 关系

- 域
- 笛卡尔积
- 关系
  - 关系，属性，元组
  - 候选码，主码，主属性
  - 基本关系的性质

### — 关系模式

### — 关系数据库



## ➤ 关系的数据操作集合

### — 查询

- 选择、投影、连接、除、并、交、差

### — 数据更新

- 插入、删除、修改



## ➤ 关系的完整性约束

- 实体完整性
- 参照完整性
  - 外码
- 用户定义的完整性

