



## 第二章 关系数据库



## 第二章 关系数据库

---

**2.1 关系模型概述**

**2.2 关系数据结构**

**2.3 关系的完整性**

**2.4 关系代数**

**2.5 关系演算**

**2.6 小结**



## 第二章 课程内容矩阵

对第一章 数据模型的再认识

数据模型要素		数据结构、数据操作、数据完整性约束	学时
关系数据库	关系	域、二维表、元组、属性、键（码）、关系模式	3
	关系数据库	关系数据库模式、关系模式的实例	2
	关系操作	查询、插入、修改、删除、集合方式、记录方式	3
	关系完整性	实体完整性、参照完整性、用户定义完整性	3
	关系代数	代数表达式、并、差、交、笛卡儿积、选择、投影、连接、自然连接、 $\theta$ 连接、除法、外连接	3
	关系演算**	演算表达式、元组演算、域演算	2



# 关系数据库简介

## □ 系统而严格地提出关系模型的是美国IBM公司的 E.F.Codd

### ◆ 1970年提出关系数据模型

- E.F.Codd, “A Relational Model of Data for Large Shared Data Banks”, 《Communication of the ACM》,1970

### ◆ 之后，提出了关系代数和关系演算的概念

### ◆ 1972年提出了关系的第一、第二、第三范式

### ◆ 1974年提出了关系的BC范式



# 关系数据库简介

---

- 关系数据库应用数学方法来处理数据库中的数据
- 80年代后，关系数据库系统成为最重要、最流行的数据库系统



# 关系数据库简介

---

## □ 典型实验系统

- ◆ System R
- ◆ University INGRES

## □ 典型商用系统

- ◆ ORACLE
- ◆ SYBASE
- ◆ INFORMIX
- ◆ DB2
- ◆ INGRES



## 第二章 关系数据库

---

### 2.1 关系模型概述

### 2.2 关系数据结构

### 2.3 关系的完整性

### 2.4 关系代数

### 2.5 关系演算

### 2.6 小结



## 2.1 关系模型概述

---

### □ 关系数据库系统

- ◆ 是支持关系模型的数据库系统

### □ 关系模型的组成

- ◆ 关系数据结构
- ◆ 关系操作集合
- ◆ 关系完整性约束





# 1. 关系数据结构

---

## □ 单一的数据结构——关系

- ◆ 现实世界的实体以及实体间的各种联系均用关系来表示。

## □ 数据的逻辑结构——二维表

- ◆ 从用户角度，关系模型中数据的逻辑结构是一张二维表。



## 2. 关系操作集合

---

- 1) 常用的关系操作
- 2) 关系操作的特点
- 3) 关系数据语言的种类
- 4) 关系数据语言的特点



# 关系操作集合(续)

---

## □ 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



# 关系操作集合（续）

---

## □ 2) 关系操作的特点

- ◆ 集合操作方式，即操作的对象和结果都是集合。
  - 非关系数据模型的数据操作方式：一次一记录



## 关系操作集合（续）

---

### □ 3) 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



# 关系操作集合（续）

## □ 关系数据语言的种类（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



## 关系操作集合（续）

### □ 4) 关系数据语言的特点

- ◆ 关系语言是一种高度非过程化的语言
  - 存取路径的选择由DBMS的优化机制来完成
  - 用户不必用循环结构就可以完成数据操作
- ◆ 能够嵌入高级语言中使用
- ◆ 关系代数、元组关系演算和域关系演算三种语言在表达能力上完全等价



### 3. 关系的三类完整性约束

---

#### □ 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





## 第二章 关系数据库

---

**2.1** 关系模型概述

**2.2** 关系数据结构

**2.3** 关系的完整性

**2.4** 关系代数

**2.5** 关系演算

**2.6** 小结



## 2.2 关系数据结构

---

- 关系模型建立在集合代数的基础上
- 关系数据结构的基本概念
  - ◆ 关系
  - ◆ 关系模式
  - ◆ 关系数据库



## 2.2 关系数据库结构

---

- 2.2.1 关系
- 2.2.2 关系模式
- 2.2.3 关系数据库



## 2.2.1 关系

---

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



# 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：

$D_1 = \text{SUPERVISOR} = \{ \text{张清玫}, \text{刘逸} \}$

$D_2 = \text{SPECIALITY} = \{ \text{计算机专业}, \text{信息专业} \}$

$D_3 = \text{POSTGRADUATE} = \{ \text{李勇}, \text{刘晨}, \text{王敏} \}$

则  $D_1$ ,  $D_2$ ,  $D_3$  的笛卡尔积为：

$D_1 \times D_2 \times D_3 =$

{ (张清玫, 计算机专业, 李勇), (张清玫, 计算机专业, 刘晨),  
(张清玫, 计算机专业, 王敏), (张清玫, 信息专业, 李勇),  
(张清玫, 信息专业, 刘晨), (张清玫, 信息专业, 王敏),  
(刘逸, 计算机专业, 李勇), (刘逸, 计算机专业, 刘晨),  
(刘逸, 计算机专业, 王敏), (刘逸, 信息专业, 李勇),  
(刘逸, 信息专业, 刘晨), (刘逸, 信息专业, 王敏) }



## 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： $2 \times 2 \times 3 = 12$ ，即  $D_1 \times D_2 \times D_3$  共有

$2 \times 2 \times 3 = 12$  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。

在上例中，**12**个元组可列成一张二维表



表 2.1  $D_1$ ,  $D_2$ ,  $D_3$ 的笛卡尔积

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



### 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。

由于笛卡尔积不满足交换律，即

$$(d_1, d_2, \dots, d_n) \neq (d_2, d_1, \dots, d_n)$$

但关系满足交换律，即

$$(d_1, d_2, \dots, d_i, d_j, \dots, d_n) = (d_1, d_2, \dots, d_j, d_i, \dots, d_n) \quad (i, j = 1, 2, \dots, n)$$

解决方法：为关系的每个列附加一个属性名以取消关系元组的有序性



## 关系（续）

例 在表2.1 的笛卡尔积中取出有实际意义的元组来构造关系

关系：SAP(SUPERVISOR, SPECIALITY, POSTGRADUATE)

◆ 关系名，属性名

假设：导师与专业：n:1，导师与研究生：1:n

于是：SAP关系可以包含三个元组

{ (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）

---

## 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22 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) 码

#### 候选码 (Candidate key)

若关系中的某一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

在最简单的情况下，候选码只包含一个属性。

#### 全码 (All-key)

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**)



## 关系（续）

---

### 主码

若一个关系有多个候选码，则选定其中一个为主码（

**Primary key**）

候选码的诸属性称为主属性（**Prime attribute**）。

不包含在任何侯选码中的属性称为非码属性（**Non-key attribute**）



# 关系（续）

---

## 7) 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



## 8) 基本关系的性质

---

### ① 列是同质的 (Homogeneous)

每一列中的分量是同一类型的数据，来自同一个域

### ② 不同的列可出自同一个域

其中的每一列称为一个属性

不同的属性要给予不同的属性名



## 基本关系的性质(续)

上例中也可以只给出两个域：

人（PERSON）=张清玫，刘逸，李勇，刘晨，王敏

专业（SPECIALITY）=计算机专业，信息专业

SAP关系的导师属性和研究生属性都从**PERSON域中**取值为了避免混淆，必须给这两个属性取不同的属性名，而不能直接使用域名。

例如定义：

导师属性名为SUPERVISOR-PERSON（或SUPERVISOR）

研究生属性名为POSTGRADUATE-PERSON（或POSTGRADUATE）



## 基本关系的性质(续)

### ③ 列的顺序无所谓

列的次序可以任意交换

遵循这一性质的数据库产品(如**ORACLE**), 增加新属性时, 永远是插至最后一列但也有许多关系数据库产品没有遵循这一性质, 例如**FoxPro**仍然区分了属性顺序。



## 基本关系的性质(续)

### ④ 任意两个元组不能完全相同

由笛卡尔积的性质决定，但许多关系数据库产品没有遵循这一性质。

例如：

**Oracle, FoxPro**等都允许关系表中存在两个完全相同的元组，除非用户特别定义了相应的约束条件。





## 基本关系的性质(续)

---

### ⑤ 行的顺序无所谓

行的次序可以任意交换

遵循这一性质的数据库产品(如**ORACLE**),

插入一个元组时永远插至最后一行

但也有许多关系数据库产品没有遵循这一性质, 例如

**FoxPro**仍然区分了元组的顺序



## 基本关系的性质(续)

### ⑥ 分量必须取原子值

每一个分量都必须是不可分的数据项。

这是规范条件中最基本的一条

表2.3 非规范化关系

SUPERVISOR	SPECIALITY	POSTGRADUATE	
		PG1	PG2
张清玫 刘逸	信息专业 信息专业	李勇 王敏	刘晨



## 2.2 关系数据结构

---

### 2.2.1 关系

### 2.2.2 关系模式

### 2.2.3 关系数据库



## 2.2.2 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：

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

**$R$**  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

**dom** 属性向域的映象集合

**$F$**  属性间的数据依赖关系集合



## 定义关系模式 (续)

---

例:

导师和研究生出自同一个域——人，  
取不同的属性名，并在模式中定义属性向域的映象，即说明它们分别出自哪个域：

**dom (SUPERVISOR-PERSON)**

**dom (POSTGRADUATE-PERSON)**



## 定义关系模式 (续)

关系模式通常可以简记为

$R(U)$  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

**关系模式和关系往往统称为关系**

通过上下文加以区别

---



## 2.2 关系数据结构

---

### 2.2.1 关系

### 2.2.2 关系模式

### 2.2.3 关系数据库



## 2.2.3 关系数据库

---

1. 关系数据库

2. 关系数据库的型与值



# 1. 关系数据库

---

在一个给定的应用领域中，**所有实体及实体之间联系的关系的集合**构成一个关系数据库。



## 2. 关系数据库的型与值

---

关系数据库也有型和值之分

关系数据库的型称为关系数据库模式，是对关系数据库的描述

若干域的定义

在这些域上定义的若干关系模式

关系数据库的值是这些关系模式在某一时刻对应的关系的集合，通常简称为关系数据库



## 第二章 关系数据库

---

2.1 关系模型概述

2.2 关系数据结构

2.3 关系的完整性

2.4 关系代数

2.5 关系演算

2.6 小结



## 2.3 关系的完整性

关系模型的完整性规则是对关系的某种约束条件。

关系模型中三类完整性约束：

实体完整性

参照完整性

用户定义的完整性

**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。



# 关系的完整性(续)

---

**2.3.1 实体完整性**

**2.3.2. 参照完整性**

**2.3.3. 用户定义的完整性**





## 2.3.1 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

**SAP(SUPERVISOR, SPECIALITY, POSTGRADUATE)**

**POSTGRADUATE** 属性为主码

(假设研究生不会重名)，则其不能取空值



## 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) 主码中的属性即主属性不能取空值。

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。



# 关系的完整性

---

2.3.1 实体完整性

2.3.2. 参照完整性

2.3.3. 用户定义的完整性



## 2.3.2 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。

**例1 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，专业号，年龄）

专业（专业号，专业名）



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

专 业 号	专 业 名
0 1	信 息
0 2	数 学
0 3	计 算 机







## 关系间的引用(续)

---

### 例2 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
8 0 1	张 三	女	0 1	1 9
8 0 2	李 四	男	0 1	2 0
8 0 3	王 五	男	0 1	2 0
8 0 4	赵 六	女	0 2	2 0
8 0 5	钱 七	男	0 2	1 9

## 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)

### 例3 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	20	
803	王五	男	01	20	802
804	赵六	女	02	20	805
805	钱七	男	02	19	





## 2. 外码 (Foreign Key)

---

设 $F$ 是基本关系 $R$ 的一个或一组属性，但不是关系 $R$ 的码。如果 $F$ 与基本关系 $S$ 的主码 $K_s$ 相对应，则称 $F$ 是基本关系 $R$ 的外码

基本关系 $R$ 称为参照关系 (Referencing Relation) 基本关系 $S$ 称为被参照关系 (Referenced Relation) 或目标关系 (Target Relation)。



## 外码(续)

### 说明

- 关系 $R$ 和 $S$ 不一定是不同的关系
- 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
- 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



### 3. 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





# 关系的完整性(续)

---

- 2.3.1 实体完整性
- 2.3.2. 参照完整性
- 2.3.3. 用户定义的完整性



## 2.3.3 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}



# 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