数据库系统概论

An Introduction to Database System

第二章 关系数据库

中国人民大学信息学院

关系数据库简介

- 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

第二章 关系数据库

2.1 关系数据结构及形式化定义

- 2.1.1 关系
- 2.1.2 关系模式
- 2.1.3 关系数据库

2.1.1 关系

- 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

- 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

- 建立在集合代数的基础上

关系(续)

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
  - 长度指定长度的字符串集合
  - {‘男’，‘女’}
  - ……

2. 笛卡尔积（Cartesian Product）

- 笛卡尔积

  给定一组域D_1，D_2，…，D_n，这些域中可以有相同的。

  D_1，D_2，…，D_n的笛卡尔积为：

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

  - 所有域的所有取值的一个组合
  - 不能重复

笛卡尔积（续)

- 元组（Tuple）
  - 笛卡尔积中每一个元素（d_1，d_2，…，d_n）叫作一个*n*元组（n-tuple）或简称元组(Tuple)
  - (张清玫，计算机专业，李勇)、(张清玫，计算机专业，刘晨)等都是元组
- 分量（Component）
  - 笛卡尔积元素（d_1，d_2，…，d_n）中的每一个值d_i叫作一个分量
  - 张清玫、计算机专业、李勇、刘晨等都是分量

笛卡尔积（续)

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

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

- 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

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

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

$$R（D_1，D_2，\ldots，D_n）$$

- R：关系名
- n：关系的目或度（Degree）

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）

或**一元**关系

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) 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

简单的情况：候选码只包含一个属性

全码（All-key）

最极端的情况：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）

码(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）

主属性

候选码的诸属性称为主属性（Prime attribute）

不包含在任何候选码中的属性称为非主属性（ Non-Prime attribute）

或非码属性（Non-key attribute）

关系（续）

- *D*1，*D*2，…，*D*n的笛卡尔积的某个子集才有实际含义

例：表2.1 的笛卡尔积没有实际意义

取出有实际意义的元组来构造关系

关系：SAP(SUPERVISOR，SPECIALITY，POSTGRADUATE)

假设：导师与专业：1:1，　导师与研究生：1:n

主码：POSTGRADUATE（假设研究生不会重名）

SAP关系可以包含三个元组

{ (张清玫，计算机专业，李勇)，

(张清玫，计算机专业，刘晨)，

(刘逸，信息专业，王敏) }

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

关系（续）

8)基本关系的性质

① 列是同质的（Homogeneous）

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

③ 列的顺序无所谓,，列的次序可以任意交换

④ 任意两个元组的候选码不能相同

⑤ 行的顺序无所谓，行的次序可以任意交换

基本关系的性质(续)

⑥ 分量必须取原子值

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	信息专业	李勇	刘晨
刘逸	信息专业	王敏	

小表

2.1 关系数据结构

2.1.1 关系

2.1.2 关系模式

2.1.3 关系数据库

2.1.2 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，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）

=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$_2$，…，A$_n$)

- *R:* 关系名
- A_1，A_2，…，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

2.1 关系数据结构

2.1.1 关系

2.1.2 关系模式

2.1.3 关系数据库

2.1.3 关系数据库

- 关系数据库
  - 在一个给定的应用领域中，所有关系的集合构成一个关系数据库
- 关系数据库的型与值

2. 关系数据库的型与值

- 关系数据库的型: 关系数据库模式

  对关系数据库的描述。

- 关系数据库模式包括
  - 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: 关系模式在某一时刻对应的关系的集合，简称为关系数据库

第二章 关系数据库

2.2.1基本关系操作

- 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
  - 选择、投影、并、差、笛卡尔基是5种基本操作
- 关系操作的特点
  - 集合操作方式：操作的对象和结果都是集合，一次一集合的方式

2.2.2 关系数据库语言的分类

- 关系代数语言
  - 用对关系的运算来表达查询要求
  - 代表：ISBL
- 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
    - 代表：APLHA, QUEL
  - 域关系演算语言
    - 谓词变元的基本对象是域变量
    - 代表：QBE
- 具有关系代数和关系演算双重特点的语言
  - 代表：SQL（Structured Query Language）

第二章 关系数据库

2.1 关系数据结构及形式化定义

2.2 关系操作

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

2.3.1 关系的三类完整性约束

2.3.2 实体完整性

2.3.3 参照完整性

2.3.4 用户定义的完整性

2.3.1 关系的三类完整性约束

- 实体完整性和参照完整性：

  关系模型必须满足的完整性约束条件

  称为关系的两个不变性，应该由关系系统自动支持

- 用户定义的完整性：

  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

2.3 关系的完整性

2.3.1关系的三类完整性约束

2.3.2 实体完整性

2.3.3 参照完整性

2.3.4 用户定义的完整性

2.3.2 实体完整性

规则2.1 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：

主码（假设研究生不会重名）

不能取空值

实体完整性(续)

实体完整性规则的说明

(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。

(2) 现实世界中的实体是可区分的，即它们具有某种唯一性标识。

(3) 关系模型中以主码作为唯一性标识。

(4) 主码中的属性即主属性不能取空值。

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

2.3关系的完整性

2.3.1关系的三类完整性约束

2.3.2 实体完整性

2.3.3 参照完整性

2.3.4 用户定义的完整性

2.3.3 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

学生（学号，姓名，性别，专业号，年龄）

主码

专业（专业号，专业名）

主码

- 学生关系引用了专业关系的主码“专业号”。
- 学生关系中的“专业号”值必须是确实存在的专业的专业号，即专业关系中有该专业的记录。

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）

关系间的引用(续)

例3 学生实体及其内部的一对多联系

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	

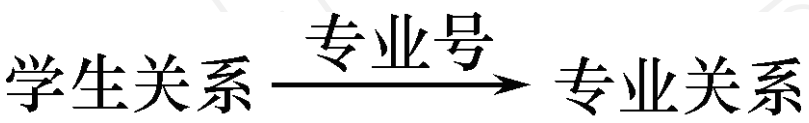
- “学号”是主码，“班长”是外码，它引用了本关系的“学号”
- “班长”必须是确实存在的学生的学号

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)

- ［例1］：学生关系的“专业号与专业关系的主码“专业号”相对应
  - “专业号”属性是学生关系的外码
  - 专业关系是被参照关系，学生关系为参照关系



(a)

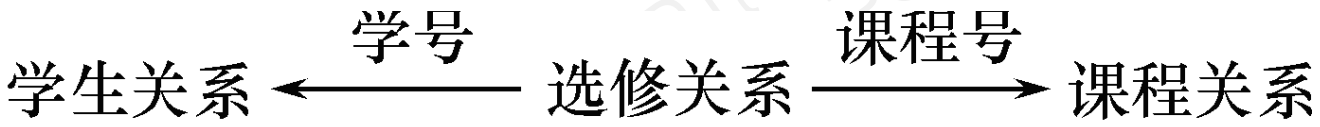
外码(续)

- ［例2］：

  选修关系的“学号” 与学生关系的主码“学号”相对应

  选修关系的“课程号”与课程关系的主码“课程号”相对应

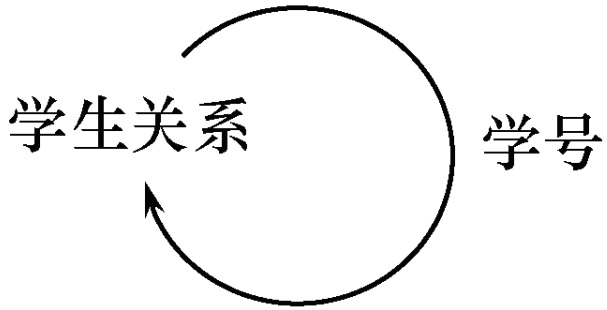
  - “学号”和“课程号”是选修关系的外码
  - 学生关系和课程关系均为被参照关系
  - 选修关系为参照关系



(b)

外码(续)

- ［例3］：“班长”与本身的主码“学号”相对应
  - “班长”是外码
  - 学生关系既是参照关系也是被参照关系



(c)

外码(续)

- 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

规则2.2 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)

[例1]：

学生关系中每个元组的“专业号”属性只取两类值：

（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一个不存在的专业

参照完整性规则(续)

〔例2〕：

选修（学号，课程号，成绩）

“学号”和“课程号”可能的取值：

（1）选修关系中的主属性，不能取空值

（2）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)

例3〕：

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，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关系的三类完整性约束

2.3.2 实体完整性

2.3.3 参照完整性

2.3.4 用户定义的完整性

2.3.4 用户定义的完整性

- 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)

- “课程号”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}

休息一会儿。。。