南京大学 NANJING UNIVERSITY

奋进行动 团结奋斗 争先进位 坐言起行 应势而动

数 据 管 理 基 础

第2章 关系数据库

（关系数据结构、关系操作、关系的完整性）

智能软件与工程学院

关系数据库

- 1970，IBM公司的 E. F. Codd（81年图灵奖） 提出关系数据模型

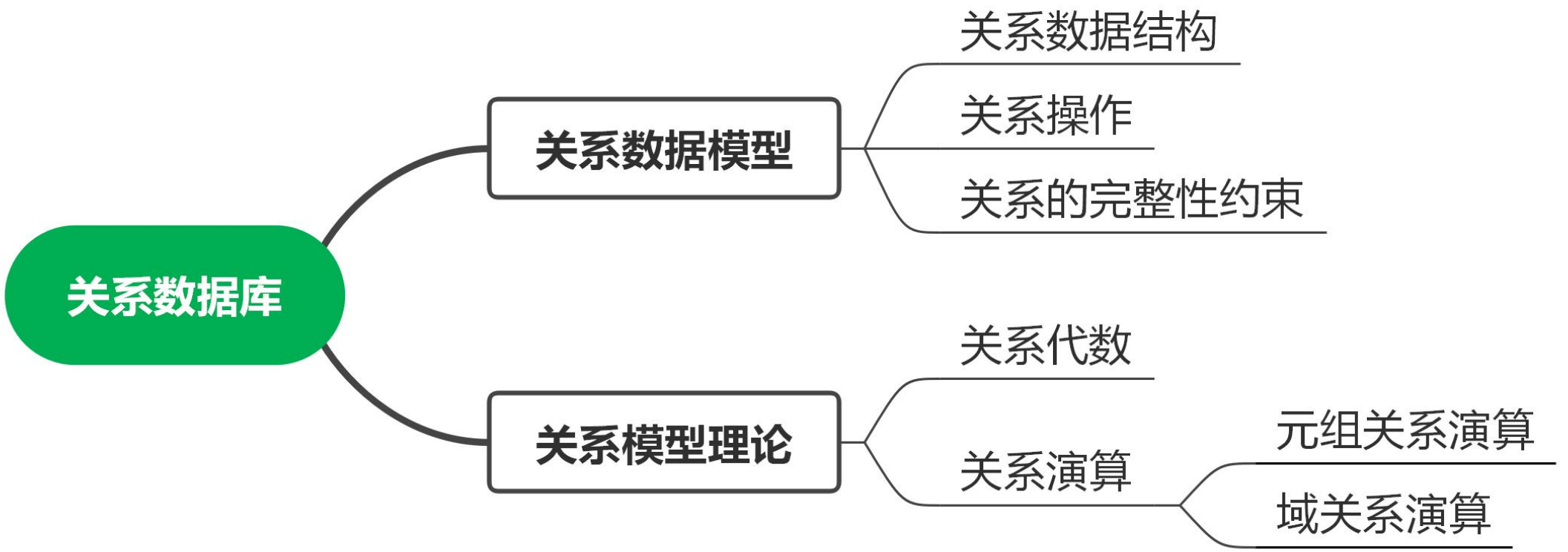
  The Relational Model of Data for Large Shared Data Banks.

- 课程内容
  - 第2章：关系数据模型（关系模型的数据结构、关系操作和数据完整性；关系代数和关系演算）
  - 第3章：关系数据库子语言SQL
  - 第4和5章：数据安全性、数据完整性
  - 第6章：关系数据库规范化理论
  - 第7章：数据库设计
  - 第8章：数据库编程
  - 第10和11章：事务处理（数据库恢复、并发控制）

关系数据库中名词术语之间的对应关系

关系模型理论	关系数据库管理系统（SQL)	文件系统
关系 relation	表 table	文件 set of records
属性 attribute	列 column	字段 field
元组 tuple	行 row	记录 record
关系模式 schema	表头 table heading	记录型 type of record
码 key 候选码 candidate key	主码 primary key 唯一码 unique	

❑本章知识框架


关系数据库
关系数据模型
关系数据结构
关系操作
关系的完整性约束
关系模型理论
关系代数
关系演算
元组关系演算
域关系演算

第2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

2.1.4 关系模型的存储结构

2.1 关系数据结构及形式化定义

- 关系模型 仅有单一的数据结构 -- 关系
  - 现实世界中的实体以及实体间的各种联系均用关系来表示
- 关系模型 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
- 关系模型 是建立在集合代数的基础上

- 要理解关系数据模型，首先需要理解下述三个概念
  - 域 (Domain)
  - 笛卡尔积 (Cartesian Product)
  - 关系 (Relation)

域（Domain）

[定义2.1] ‘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：[18,60]
  - 指定长度的字符串集合：char(25)
  - 枚举数据类型：{‘男’,‘女’}
- 例如：
  - D_1 = SUPERVISOR = { 张清玫，刘逸 }
  - D_2 = SPECIALITY = {计算机专业，信息专业}
  - D_3 = POSTGRADUATE = {李勇，刘晨，王敏}
- 每一个域有一个‘域名’，同一个域中的元素互不相同。

笛卡尔积 1

[定义2.2] 笛卡尔积（Cartesian Product）

- 给定一组域D_1，D_2，…，D_n，允许其中某些域是相同的。
- D_1，D_2，…，D_n的笛卡尔积可表示为：$D_1 \times D_2 \times \ldots \times D_n$
- 笛卡尔积的运算结果也是一个集合，其中的每个元素都是一个具有如下形式的‘n元组’：$(d_1, d_2, \ldots, d_n)$，其中 $d_i \in D_i (i=1, 2, \ldots, n)$
- 由所有符合上述要求的‘n元组’组成该笛卡儿积的运算结果，即：

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

- 笛卡尔积的运算结果
  - 所有域的所有取值的一个组合
  - 不存在重复的 n 元组

笛卡尔积 2

❑ 元组（Tuple）

- 笛卡尔积中每一个元素 $(d_1, d_2, \ldots, d_n)$ 叫作一个‘n元组’(n-tuple)或‘n元有序组’，简称‘元组’
- 例如：
  - (张清玫，计算机专业，李勇)
  - (张清玫，计算机专业，刘晨)

❑分量（Component）

- 笛卡尔积元素 $(d_1, d_2, \ldots, d_n)$ 中的每一个值 d_i 叫作一个‘分量’
- 例如：张清玫、计算机专业、李勇、刘晨等
- 一个‘n元组’含有 n 个‘分量’，按照书写顺序建立‘分量’与‘域’的对应关系，即：$d_i \in D_i \ (i = 1, 2, \cdots, n)$

笛卡尔积 3

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

- **笛卡尔积的结果表示方法**
  - **集合表示**：列举出结果集中的所有元组
  - **二维表表示**：
    - 将笛卡尔积的运算结果表示为一张二维表
    - 表中的每一行对应一个元组，每一列对应一个域
    - 为了区分每一列所对应的域，需要在二维表中添加一个‘表头’(table heading -- 二维表第一行)，用于显示每一列对应域的域名

笛卡尔积 4 - 集合表示

- 例如，给出3个域：
  - D_1 = 导师集合SUPERVISOR = {张清玫，刘逸}
  - D_2 = 专业集合SPECIALITY = {计算机专业，信息专业}
  - D_3 = 研究生集合POSTGRADUATE = {李勇，刘晨，王敏}

- D_1, D_2, D_3的笛卡尔积（其基数为2×2×3＝12）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 5 - 二维表表示

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

笛卡尔积 6 - 思考

- 例如，给出3个域：
  - D_1 = 导师集合SUPERVISOR = {张清玫，刘逸}
  - D_2 = 专业集合SPECIALITY = {计算机专业，信息专业}
  - D_3 = 研究生集合POSTGRADUATE = {李勇，刘晨，王敏}

  D_1, D_2, D_3的笛卡尔积可表示为 $D_1 \times D_2 \times D_3$

- 问：

  1. 在上述笛卡尔积 $D_1 \times D_2 \times D_3$ 中，是否存在相同的‘域’？

  2. 请给出一个笛卡尔积的例子，其中使用到了相同的域。

关系 1

[定义2.3] **关系**（Relation）

- 给定一个域的序列 $D_1, D_2, \ldots, D_n$（其中可能存在相同的域），笛卡尔积 $D_1 \times D_2 \times \ldots \times D_n$ 的子集叫做在域 $D_1, D_2, \ldots, D_n$ 上的关系，表示为 $R(D_1, D_2, \ldots, D_n)$，其中：
  - R 是关系名，用于区分不同的关系
  - n 是关系的‘目’或‘度’（Degree）
    - 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）
    - 当 $n > 2$ 时，称该关系为**n-元**关系 或 **多元**关系

元组 （Tuple）

- 关系中的每个元素是关系中的**元组**，通常用 t 表示

关系 2

❑属性（Attribute）

- 关系也可以被表示为一张‘二维表’，表中的每一行（不包括第一行）对应关系中的一个元组，表中的每一列对应一个域。
- 在一个关系中，不同列可以对应相同的域。为了加以区分，关系所对应的二维表中的每一列，被称为是该关系中的一个‘**属性**’。
- 关系中的每一个属性都有一个名字，称为‘**属性名**’。在同一个关系中，属性名互不相同。
- 在关系对应的二维表中，表中的第一行被称为是二维表的‘表头’，里面填写的是各个属性的属性名。除第一行表头外，其他的每一行都是关系中的一个元组。
- *n*目关系必有*n*个属性

关系 3

❑关系的表示

在域$D_1, D_2, \ldots, D_n$上的n目关系R可以被表示为 $R(A_1, A_2, \ldots, A_n)$, 其中：

- R是关系名
- $A_1, A_2, \ldots, A_n$是属性名
- 例如：表2.2

表2.2 SAP关系

SUPERVISOR	SPECIALITY	POSTGRADUATE
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

❑元组分量的表示

- 若 t 是关系R中的一个元组，那么用$t[A_i]$表示元组t在属性A_i上的取值（元组分量，也称为属性值），且 $t[A_i] \in D_i\ (i = 1, 2, \ldots, n)$
- 或者，用 $t[i]$ 表示元组t在第 i 列上的取值，且 $t[i] \in D_i\ (i = 1, 2, \cdots, n)$

关系 4

- **码(Key)、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，而其所有的真子集都不能，则称该属性组为关系的‘候选码’，简称‘码’。
  - 简单的情况：一个候选码只包含一个属性；
  - 最极端的情况：由关系中的所有属性构成的属性组是这个关系的候选码，称为‘全码’（All-key）。
  - 每一个关系中都存在‘候选码’
  - 候选码是语义范畴的概念，是现实世界中普遍存在的客观规律，不能仅根据当前关系中的元组集合来确定其候选码的组成。

关系 5

❑思考：

1. 如表2.2所示的关系，其候选码是什么？

表2.2 SAP关系

SUPERVISOR	SPECIALITY	POSTGRADUATE
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

2. 请设计一个候选码只包含单个属性的关系。

3. 请设计一个有多个候选码的关系。

4. 请设计一个候选码是全码的关系。

关系 6

- **主码（Primary key）**
  - 在一个关系中，可以选择一个候选码作为该关系的‘主码’。
  - ‘主码’是关系数据库管理系统（SQL）中才有的概念。当我们在创建关系对应的‘基表’时，可以为基表定义‘主码’也可以不定义‘主码’。在一张基表中，最多只能定义一个主码。
  - 在关系模型理论中，只有‘候选码’，不需要为关系定义‘主码’。

- **主属性 与 非主属性/非码属性**

  在一个关系中
  - 候选码中的诸属性称为该关系的‘主属性’（Prime attribute）
  - 不包含在任何候选码中的属性称为该关系的‘非主属性’（Non-Prime attribute）或‘非码属性’（Non-key attribute）

关系 7 - 关系的类别

- 关系可以有三种类型
  - 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。

- 说明：
  ① ‘表’（包括基本表、基表、查询表、视图表）的概念，一般只在关系数据库管理系统（即SQL语言）中使用；
  ② 关系模型是一种逻辑数据模型，不涉及数据的物理存储，在关系模型理论中，其数据结构只有‘关系’；
  ③ 在关系模型上进行数据操纵，其操作结果也构成一个‘关系’（也可以通俗地称为‘结果关系’、‘临时关系’或‘中间关系’）。

关系 8 - 关系的性质

- 按照定义2.2和定义2.3，关系可以是一个无限集合，且定义关系的笛卡尔积不满足交换律。

关系：笛卡尔积 $D_1 \times D_2 \times \ldots \times D_n$ 的子集

- 当使用‘关系’来作为关系数据模型的数据结构时，需要添加以下的约束：
  - 必须是一个有限子集
  - 笛卡尔积满足交换律（或者说：关系中的列/属性满足‘无序性’）
    - 给定一组域 $D_1, D_2, \cdots, D_n$，不论以何种顺序来计算它们的笛卡尔积，其结果是等价的
    - 即：$D_1 \times \ldots \times D_i \times D_{i+1} \times \ldots \times D_n$ 和 $D_1 \times \ldots \times D_i \times D_{i+1} \times \ldots \times D_n$ 等价，且等价具有传递性
- 经过以上约定和扩充后，关系模型数据结构‘关系’具有以下性质

关系 9 - 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。（属性名的唯一性）

③ 列的无序性（属性的无序性）
- 列的排列顺序无所谓，列与列之间的次序可以任意交换；
- 可通过属性名指定需要访问的列。

④ 行的唯一性（元组的唯一性）
- 任意两个元组在‘候选码’属性上的取值不能相同。

⑤ 行的无序性（元组的无序性）
- 行的排列顺序无所谓，行与行之间的次序可以任意交换；
- 可通过指定行在列上的取值来确定需要访问的行。

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。

关系 10

- 在关系模型理论中，每一个关系都必须满足前述的6个条件。或者说，只有同时满足这6个条件的二维表才能被称为‘关系’。
- 但是，在许多实际关系数据库产品中，基本表并不完全具有这6条性质。例如：
  - 有些数据库产品仍然区分属性顺序和元组顺序
  - 有些数据库产品可以根据行号(rowid)来访问元组
  - 有些数据库产品支持非规范化关系（如对象关系数据库等）

表2.3　非规范化关系

SUPERVISOR	SPECIALITY	POSTGRADUATE		
		PG1	PG2	PG3...
张清玫	计算机专业	李勇	刘晨	
刘逸	信息专业	王敏		

POSTGRADUATE是一个多值属性（集合）

扩展阅读：Relational Model (1)

Codd: A Relational Model of Data for Large Shared Data Banks

❑ relation

Give a list of sets S_1, S_2, ..., S_n (not necessarily distinct), R is a *relation* on these *n* sets if it is a set of *n-tuples* each of which has its first element from S_1, its second element from S_2, and so on.

$$R = \{(v_1, v_2, \ldots, v_n) \mid v_j \in S_j \ \ for \ \ j = 1,2,\ldots,n\}$$

We shall refer to $(v_1, v_2, \ldots, v_n)$ as a *n-tuple*, S_j as the *j-th domain* of R.

More concisely, R is a subset of the Cartesian product

$$R \subseteq S_1 \times S_2 \times \ldots \times S_n$$

As defined above, R is said to have *degree n*.

Relations of degree 1 are often called *unary*, degree 2 *binary*, degree 3 *ternary*, and degree n *n-ary*.

扩展阅读：Relational Model (2)

Codd: A Relational Model of Data for Large Shared Data Banks

An *n-ary* relation *R* has the following properties:

① Each row represents an *n-tuple* of R.

② The ordering of rows is *immaterial*.

③ All rows are *distinct*.

④ The ordering of columns is *significant* -- it corresponds to the ordering S_1, S_2, ..., S_n of the domains on which R is defined.

⑤ The significance of each column is partially conveyed by *labeling it with the name of the corresponding domain*.

扩展阅读：Relational Rules (3)

Patrick O'Neil and Elizabeth O'Neil. Database: Principles, Programming and Performance.

Relational Rules

Rule 1. First Normal Form Rule

- Can't have composite fields and multi-valued fields.

Rule 2. Access Rows by Content Only Rule

- No order to the rows
- No order to the columns

Rule 3. The Unique Row Rule

- Two rows can't be same in all attributes at once. So that a relation is an unordered SET of tuples.

扩展阅读：Relational Model (4)

师伯乐等: 数据库系统教程

- n-元关系
  - 是一个元数为n的元组集合，每个元组有n个属性值。
  - 如果一个关系的元组个数是无限的，那么称为无限关系；一个关系的元组个数是有限的，则称为有限关系。由于计算机存储系统的限制，如不特别声明仅研究有限关系。
- 尽管在关系、二维表、数据文件三者之间有类似之处，但它们又有区别。严格地说，关系是一种规范化了的二维表。
- 在关系数据模型中，对关系作了下列规范性限制：
  ① 关系中每一个属性值都是不可分解的，也不允许出现重复组（多值）；
  ② 关系中不允许出现相同的元组；
  ③ 由于关系是一个集合，因此不考虑元组间的顺序；
  ④ 元组中的属性理论上也是无序的，但使用时按习惯考虑列的顺序。

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

2.1.4 关系模型的存储结构

关系模式 1

- 关系的型与值
  - 关系是元组的集合
  - 关系模式是关系的‘型’，元组集合是关系的‘值’

- 关系模式（Relation Schema）
  - 关系模式是对关系的描述
  - 元组集合的结构
    - 属性构成（*关系模式的核心*）
    - 属性来自的域
    - 属性与域之间的映象关系
  - 完整性约束条件：关系中的元组分量和元组需要满足的约束条件

关系模式 2

- 关系的关系模式一般是稳定不变的，但关系的值通常会随着时间的变化而变化。
  - 关系中一个确定的元组集合被称为是该关系的一个‘值’，也被称为是一个‘关系实例’(relation instance)，它描述了关系在某个确定时刻的取值。
  - 例如：南京大学所有在读本科生构成学生关系students
    - students 是关系名，其属性构成在创建之后一般不会再变化
    - 所有在校就读本科生对应的元组集合，构成当前students实例
    - 每年的毕业季结束后，会从students关系中删除所有已毕业学生所对应的元组，得到新的students实例
    - 每年秋季新生报道结束后，所有新生元组被加入到students关系中，得到另一个新的students实例

关系模式 3

- 关系模式的形式化表示：$R(U, D, \mathrm{DOM}, F)$
  - R：关系名
  - U：组成该关系的属性名集合
  - D：U 中属性所来自的域
  - DOM：属性向域的映象集合（描述各个属性对应的域）
  - F：属性间数据的依赖关系的集合（关系上的完整性约束条件）

- 关系模式通常可以简记为 $R(U)$ 或 $R(A_1, A_2, \ldots, A_n)$
  - R 是关系名，U 是属性名的集合
  - $A_1, A_2, \ldots, A_n$ 是关系中所有属性的属性名
  - 注：域名及属性向域的映象常常直接说明为属性的类型、长度等

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笼统称为关系**
  - 通过上下文加以区别

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

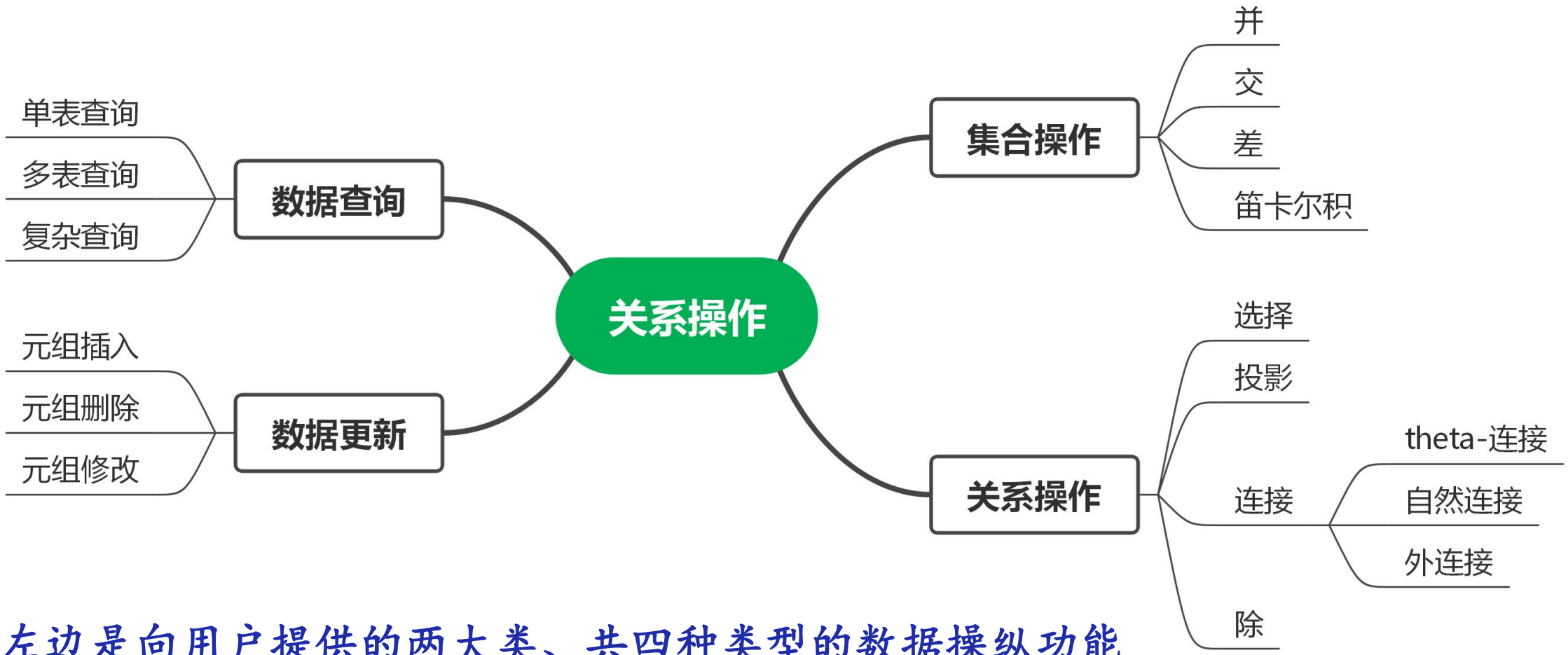
- ➢关系数据库的型
  - ●关系数据库模式，是对关系数据库的描述
  - ●包括所有关系的关系模式，以及所有其他数据库对象的定义信息
- ➢关系数据库的值
  - ●关系模式在某一时刻对应的关系的集合，通常称为关系数据库

2.1.4 关系模型的存储结构

- 关系模型的物理组织
  - 有的关系数据库管理系统中一张表对应一个操作系统文件，将物理数据组织交给操作系统完成
  - 有的关系数据库管理系统从操作系统那里申请若干个大的文件，自己划分文件空间，组织表、索引等存储结构，并进行存储管理

- 关系模型是一种逻辑数据模型，不涉及数据的物理存储组织。因此，在本门课程以及大学初级数据库课程中，一般不会过多介绍数据库物理存储实现。

基本的关系操作

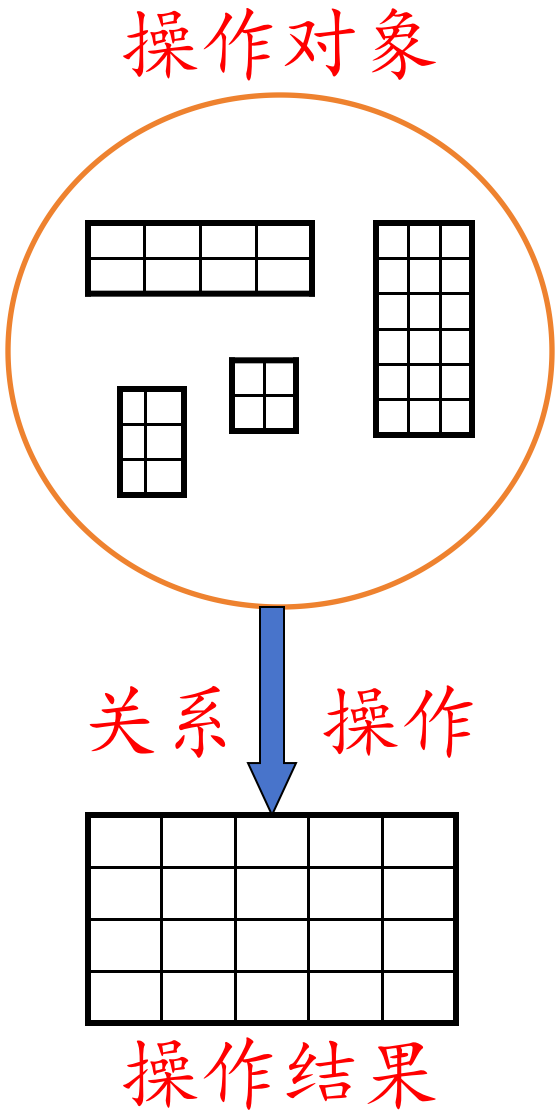


- 左边是向用户提供的两大类、共四种类型的数据操纵功能
- 右边是用来实现或表示用户的数据操纵请求的关系操作（符）
- 选择、投影、并、差、笛卡尔积是5种基本操作
- 关系操作的特点：操作的对象和结果都是关系（集合）

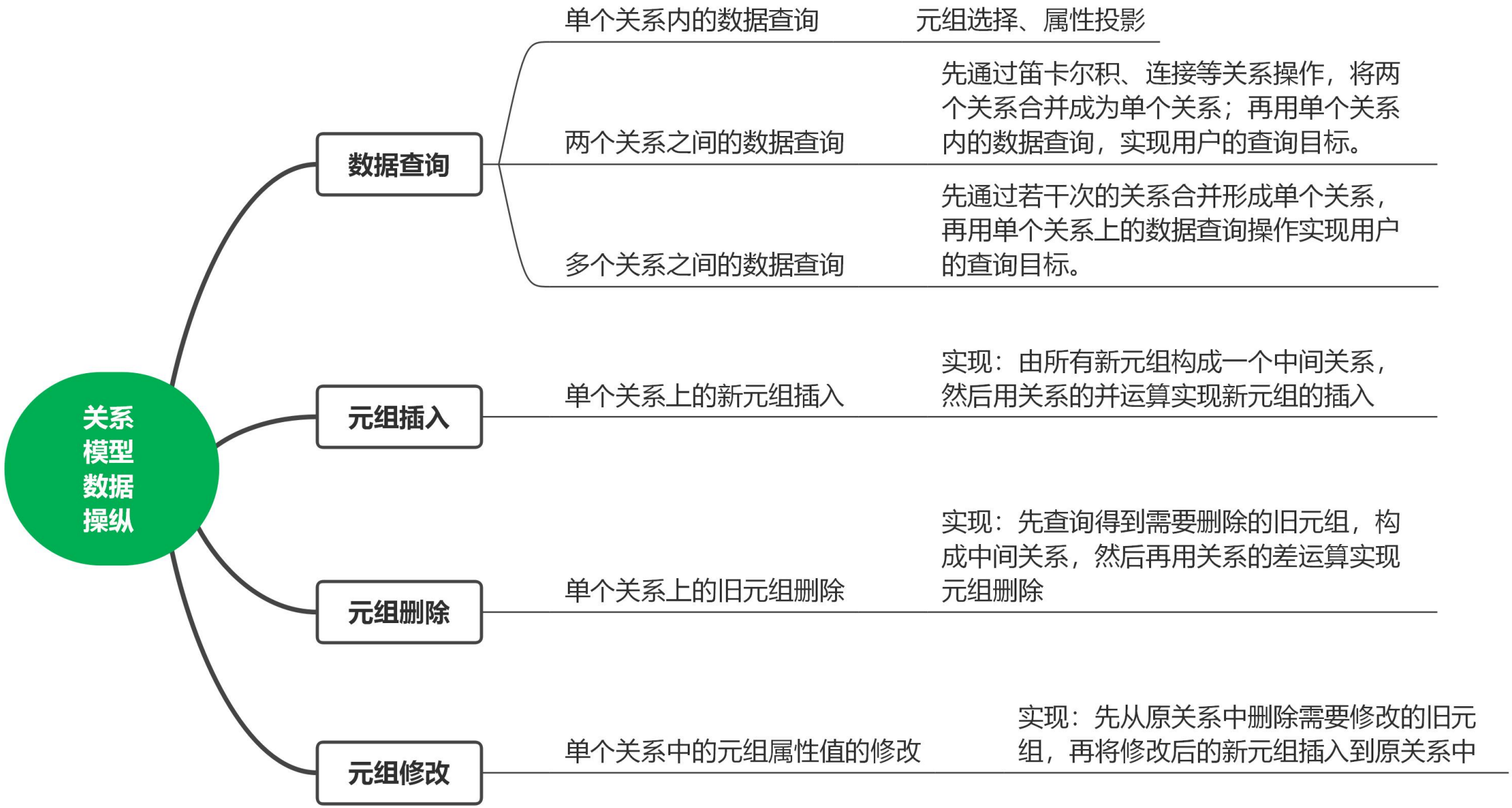
对关系操作的总结

- 操作对象是关系
- 操作结果也构成一个关系

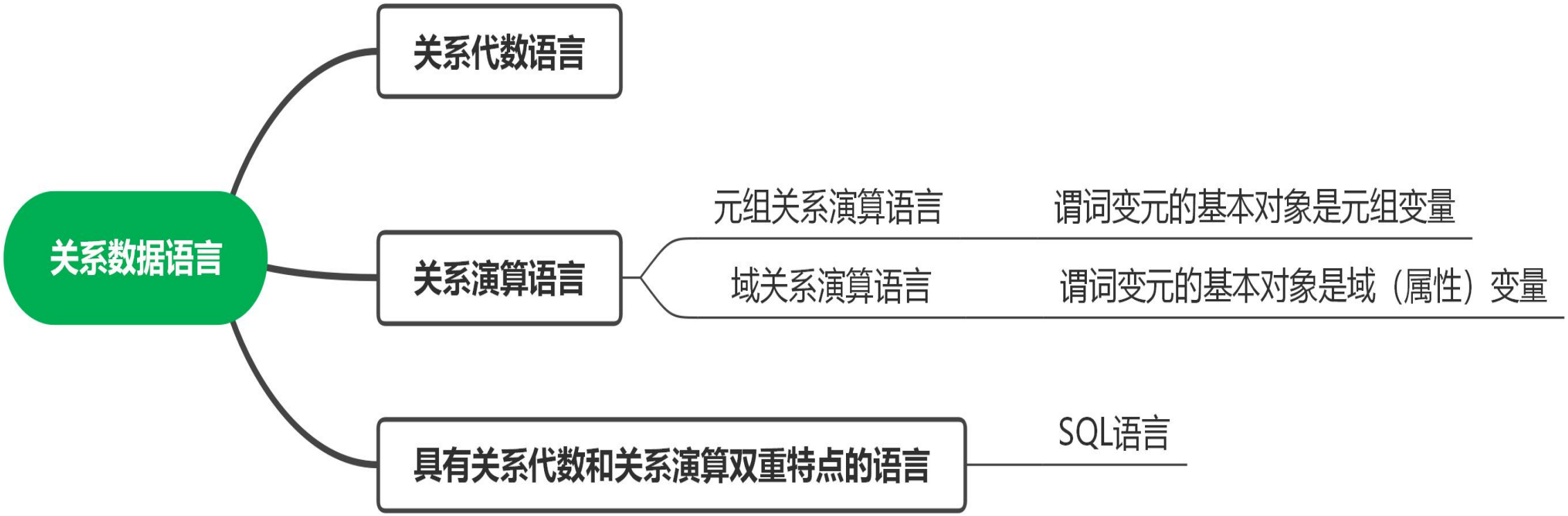
- 关系模型上的五种基本关系操作
  - （元组）选择
  - （属性）投影
  - （两个关系的）笛卡尔积
  - （两个关系的）并
  - （两个关系的）差



关系模型上的数据操纵功能



关系数据语言的分类


关系数据语言
关系代数语言
关系演算语言
元组关系演算语言
谓词变元的基本对象是元组变量
域关系演算语言
谓词变元的基本对象是域（属性）变量
具有关系代数和关系演算双重特点的语言
SQL语言

第2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

关系的三类完整性约束

- 关系的完整性是对关系的某种约束条件，是关系的值及其随时间变化时应该满足的一些约束条件。关系在任何时刻都要满足这些约束要求。

- 在关系模型中，可以定义三类完整性约束
  - ① 实体完整性
  - ② 参照完整性
    - 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”，由关系系统自动支持。
  - ③ 用户定义的完整性
    - 用户定义的完整性是特定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。

2.3.1 实体完整性

❑规则2.1 实体完整性规则

- ➢实体完整性（Entity Integrity）是指关系中元组（二维表中的行）的唯一性。
- ➢在关系数据库管理系统中，实体完整性规则是指：若属性A是基本关系（基表）的主码中的属性，则属性A不能取‘空值’(null)
- ➢在关系模型理论中，关系的性质④表明，关系满足实体完整性约束
- ➢在关系数据库管理系统中，创建基表时可以定义主码也可以不定义主码。
  - ●如果没有为基表定义主码，那么就允许在基表中出现重复的行（元组），这样的基表就不满足实体完整性约束；
  - ●如果为基表定义主码，DBMS将自动保证“主码中的所有属性都不能取空值”以及“主码值的唯一性”。

2.3.1 实体完整性

- 空值 就是“不知道”或“不存在”或“无意义”的值
- 实体完整性约束的例子
  - 学生（学号，姓名，性别，专业号，年龄）
    - “学号”是学生关系的主码
    - 在学生关系中，学号属性上不允许出现空值
  - 选修（学号，课程号，成绩）
    - “学号、课程号”是选修关系的主码
    - 在选修关系中，“学号”和“课程号”两个属性都不能取空值

2.3.1 实体完整性

❑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（表）而言的。一个基本表通常对应现实世界中的一个实体集。

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. 在关系数据库管理系统中，以‘主码’作为基本表中元组（实体）的唯一性标识。

4. 主码中的属性不能取空值。
   - 主码中的属性取空值，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’。

2.3.2 参照完整性 - 关系间的引用 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，自然存在着关系与关系间的引用。

❑例1 ：（实体之间的关系引用）

学生实体、专业实体以及专业与学生间的一对多联系，学生关系引用了专业关系的主码“专业号”。

- ➢学生（学号，姓名，性别，专业号，年龄）
- ➢专业（专业号，专业名）

2.3.2 参照完整性 - 关系间的引用 2

❑例2（联系与相关实体之间的关系引用）

学生、课程、学生与课程之间的多对多联系，选修关系引用了学生关系的主码“学号”和课程关系的主码“课程号”。

- 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）

❑例3：（实体集内部的关系引用）

学生实体及其内部的领导联系(一对多)，学生关系引用了自身的主码“学号”

- 学生（学号，姓名，性别，专业号，年龄，班长）

2.3.2 参照完整性 - 外码 1

❑定2.5 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）

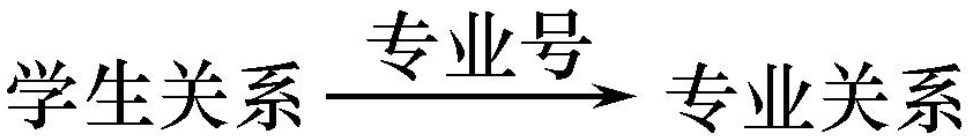
❑其中

- 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必须定义在同一个域上
- 外码并不一定要与相对应的主码同名
  - 当外码与相对应的主码属于不同关系时，往往取相同的名字，以便于识别

2.3.2 参照完整性 - 外码 2

- 在例1中，学生关系的“专业号”与专业关系的主码“专业号”相对应
  - 学生（学号，姓名，性别，专业号，年龄）
  - 专业（专业号，专业名）

- “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)

2.3.2 参照完整性 - 外码 3

- 例2中，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

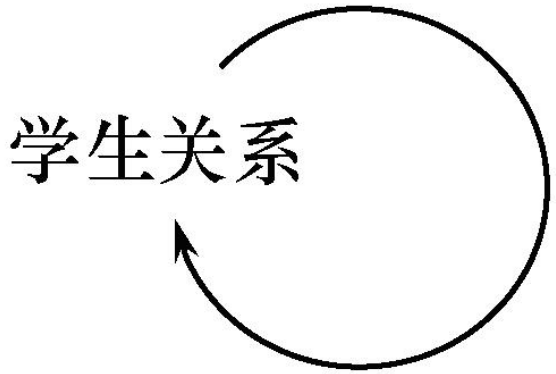
(b)

2.3.2 参照完整性 - 外码 4

- 例3中，“班长”与本身的主码“学号”相对应

  学生（学号，姓名，性别，专业号，年龄，班长）

  - “班长”是外码
  - 学生关系既是参照关系也是被参照关系



(c)

班长

标注在箭头上的是参照关系中的‘外码’

❑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

2.3.2 参照完整性 - 参照完整性规则 2

- 例1中，学生关系中每个元组的“专业号”属性只取两类值：
  - 空值，表示尚未给该学生分配专业
  - 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

(a) 学生关系

学号	姓名	性别	专业号	年龄
801	张君	女	01	19
802	李明	男	01	20
803	王强	男	01	20
804	赵蕾	女	02	20
805	钱虎	男	02	19

(b) 专业关系

专业号	专业名
01	信息
02	数学
03	计算机

2.3.2 参照完整性 - 参照完整性规则 3

- 例2中，选修（学号，课程号，成绩）
  - “学号”和“课程号”可能的取值 ：
    - 选修关系中的主属性，不能取空值
    - 只能取相应被参照关系中已经存在的主码值

(a) 学生关系

学号	姓名	性别	专业号	年龄
801	张君	女	01	19
802	李明	男	01	20
803	王强	男	01	20
804	赵蕾	女	02	20
805	钱虎	男	02	19

(c) 课程关系

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

(d) 选修关系

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

2.3.2 参照完整性 - 参照完整性规则 4

- 例3中，学生（学号，姓名，性别，专业号，年龄，班长）
  - 这是学生关系内部的引用，外码“班长”属性值可以取两类值：
    - 空值，表示该学生所在班级尚未选出班长
    - 非空值，该值必须是本关系中某个元组的学号值

(e) 加入班长外码后的学生关系

学号	姓名	性别	专业号	年龄	班长
801	张君	女	01	19	802
802	李明	男	01	20	802
803	王强	男	01	20	802
804	赵蕾	女	02	20	805
805	钱虎	男	02	19	805

2.3.3 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约束的机制，以便用统一的系统的方法处理它们，而不是由应用程序来承担这一功能。

- 例：课程（课程号，课程名，学分）

  在课程关系中，可以定义以下的用户定义的完整性约束：

  - “课程号”属性值的唯一性约束（除了主码，其他候选码的定义方式）
    - 如果已经将‘课程号’定义为课程关系的主码，就不需要再定义该属性上的唯一性约束
  - 非主属性“课程名”的非空约束（不能取空值）
  - 取值范围约束：“学分”属性只能取值{1，2，3，4}

数据完整性约束（示例）

学生(学号, 姓名, 性别, 身份证号, 院系代码, 专业代码, 录取大类代码)

- （实体完整性）主码：学号
- （参照完整性）3个外码：院系代码，专业代码，录取大类代码
- （用户定义的完整性）
  - 唯一性约束：身份证号
  - 非空约束
    - 主属性：身份证号
    - 非主属性：姓名，性别，录取大类代码
    - 主码属性自带非空约束，其他属性（没有定义非空约束的院系代码和专业代码）允许取空值
  - 取值范围约束
    - ‘性别’属性的取值范围是｛男，女｝

总结

- 关系数据库系统是目前使用最广泛的数据库系统，关系模型、基于关系模型的关系数据库系统是本门课程的重点。
- 关系数据库系统与其他非关系数据库系统的区别，就是关系数据库系统只有‘关系’（或‘表’、‘二维表’）这一种数据结构。
- 数据模型是区分不同类型数据库管理系统的依据。要学习关系数据库，首先要学懂弄通什么是关系数据模型。
- 本课件重点讲解关系模型的基本概念，核心知识点包括：
  - 关系模型的数据结构、关系操作、关系的完整性
  - 关系、关系模式、关系数据库 以及三者之间的联系和区别
  - ‘关系’和‘二维表’（表）的联系和区别

习言道 | 学以致用、用以促学、学用相长

复习思考题（1）

1. 试述关系模型的三个组成部分。
2. 在关系数据模型中，定义并理解下述术语，说明它们之间的联系和区别：
   ① 域，笛卡尔积，关系，属性，元组，分量
   ② 表/二维表，列，行，表头
   ③ 码，候选码，全码，主码，外码
   ④ 主属性，非主属性
   ⑤ 关系模式，关系，关系数据库
3. 被称为‘关系’的二维表需要满足哪些性质？
4. 关系数据库管理系统(SQL语言)中的‘表’和关系模型中的‘关系’有什么区别？
5. 关于‘关系’的定义
   ① 为什么要对关系及关系中的属性进行命名？
   ② 在一个关系中，是否允许存在同名的属性？
   ③ 在一个关系中，不同的属性是否允许对应同一个域(domain)？
   ④ 如何理解关系中的两个‘无序性’：元组的无序性 & 属性的无序性？
   ⑤ 在一个n元关系中，元组为什么也被称为是‘n元有序组’？
   ⑥ 什么是关系的‘型’？什么是关系的‘值’？

复习思考题（2）

6. 请回答以下问题：
   ① 在不同的关系之间，是否允许存在同名的属性？
   ② 在不同的关系之间，是否允许存在值域相同的属性？
   ③ 来自两个不同关系中的同名属性，它们的值域是否一定也相同？
   ④ 来自两个不同关系中的不同名属性，它们的值域是否一定也不相同？

7. 为什么要为关系定义码(Key)？请利用关系和码的定义，给出下述定理的描述性证明：每一个关系中都有码。

8. 请按照下述要求分别写出一个例子关系，并说明理由（只需要写出关系模式和语义约束，不需要写出元组集合）
   ① 设计一个关系，存在由多个属性组合构成的码；
   ② 设计一个关系，有多个不同的候选码；
   ③ 设计一个关系，由关系中的所有属性构成该关系的码。

复习思考题（3）

9. 设有右图所示的关系 T_1 及其元组集合，在仅有这四个元组的情况下，请找出关系 T_1 的所有候选码。

T_1

A	B	C	D
a_1	b_1	c_1	d_1
a_2	b_3	c_1	d_2
a_3	b_4	c_2	d_2
a_4	b_2	c_2	d_1

10. 设有右图所示的关系 T_2 及其元组集合，在仅有这四个元组的情况下，请找出关系 T_2 的所有候选码。

T_2

A	B	C	D	E
a_1	b_1	c_1	d_1	e_1
a_2	b_1	c_1	d_1	e_2
a_3	b_1	c_2	d_1	e_1
a_4	b_2	c_1	d_1	e_1

11. 请仿照第9题的关系 T_1，自己创建一个新的关系 R_1，要求：
   （1）关系 R_1 中含有 A, B, C, D 四个属性和四个元组；
   （2）只有一个候选码 $\{A, B, C\}$.

12. 请仿照第10题的关系 T_2，自己创建一个新的关系 R_2，要求：
   （1）关系 R_2 中含有 A, B, C, D, E 五个属性和五个元组；
   （2）只有一个候选码 $\{A, B, C, D\}$.

复习思考题（4）

13. 数据库系统向用户提供了哪四种数据操纵方式？在关系数据模型中，这四种操纵方式又可以被分解为哪五种基本数据操纵功能？分别对应着哪五种关系操作？

14. 关系模型中的完整性约束规则有哪些？请简要叙述每一条完整性约束规则内容。

15. 请举例说明，在一个关系中，既有不允许取空值的候选码，也有允许取空值的候选码。

16. 请举例说明，在参照完整性中，什么情况下在外码属性上允许取空值。