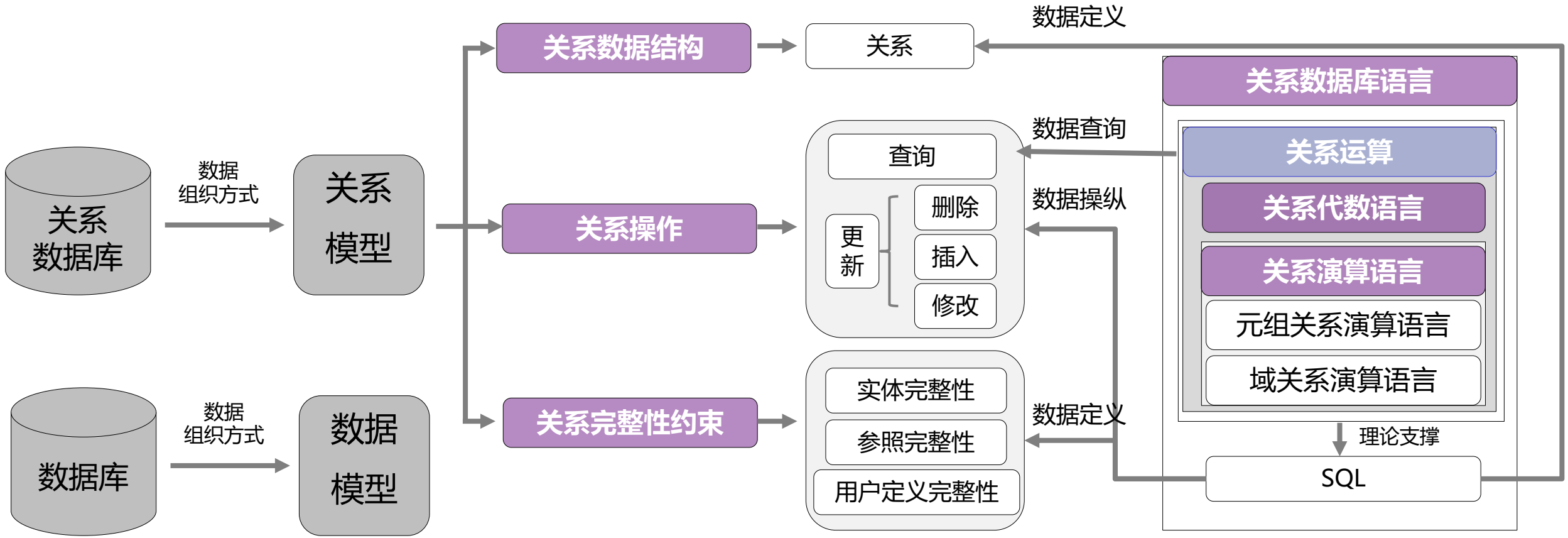
第2章 关系模型

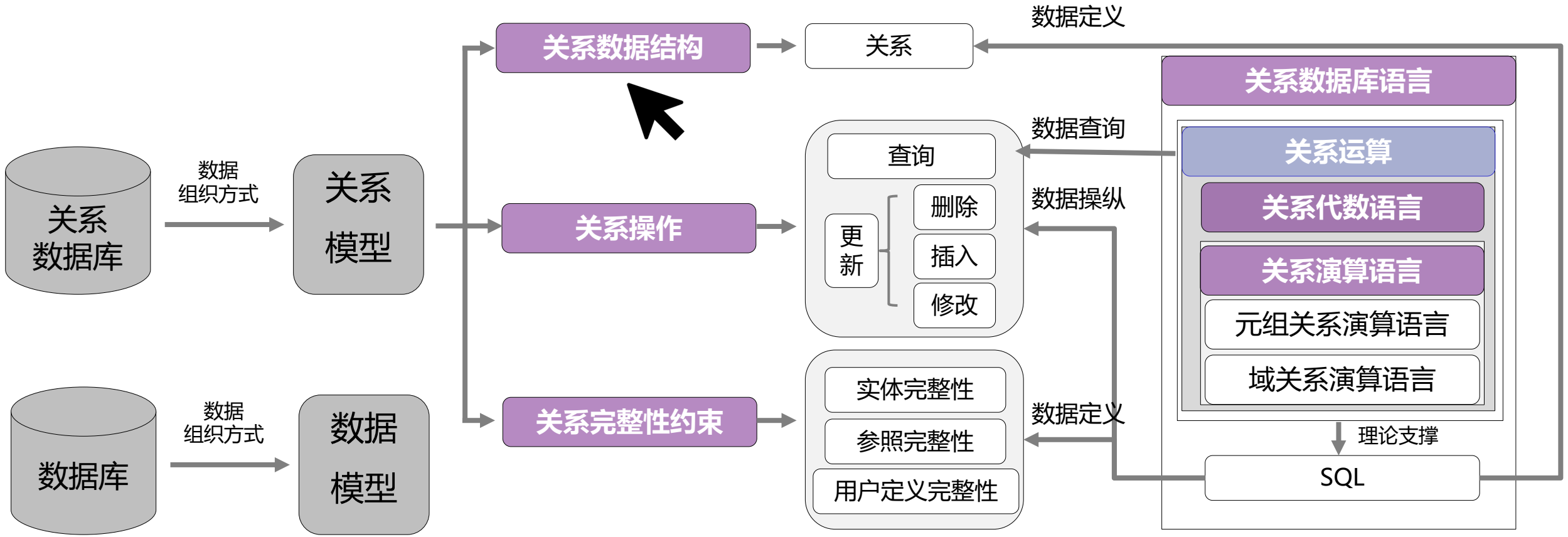
清华大学计算机系 李国良

liguoliang@tsinghua.edu.cn
http://dbgroup.cs.tsinghua.edu.cn/ligl

本章知识框架


关系数据结构
关系
数据定义
关系数据库语言
关系数据库
数据
组织方式
关系
模型
关系操作
查询
数据查询
关系运算
更新
删除
插入
修改
数据操纵
关系代数语言
关系演算语言
元组关系演算语言
域关系演算语言
理论支撑
SQL
数据库
数据
组织方式
数据
模型
关系完整性约束
实体完整性
参照完整性
用户定义完整性
数据定义

目录


关系数据结构
关系
数据定义
关系数据库语言
关系运算
关系代数语言
关系演算语言
元组关系演算语言
域关系演算语言
理论支撑
SQL
关系
数据库
数据
组织方式
关系
模型
关系操作
查询
更新
删除
插入
修改
数据查询
数据操纵
数据库
数据
组织方式
数据
模型
关系完整性约束
实体完整性
参照完整性
用户定义完整性
数据定义

关系数据库基本概念

- **学生选课关系数据库示例**
  - 包含三个Student、Course、SC三个表（关系）。

Student表

Sno (学号)	Sname (姓名)	Sgender (性别)	Sage (年龄)	Sdept (所在系)
2021310721	李博	男	17	CS
2021310722	赵宇	男	19	CS
2021310723	张敏	女	18	CS
2021310724	王勇	男	18	MA
2021310725	刘佳	女	17	MA

SC表

Sno (学号)	Cno (课程号)	Grade (成绩)
2021310721	5	98
2021310722	1	87
2021310723	1	92
2021310723	5	76
2021310724	7	84
2021310725	4	95

Course表

Cno (课程号)	Cname (课程名)	Cpno (先修课)	Ccredit (学分)
1	数据库	2	4
2	数据结构与算法	6	4
3	操作系统	2	3
4	高等数学		4
5	软件工程	6	2
6	程序设计		3
7	数值分析	4	2

关系数据库基本概念（续）

关系数据库

- 关系数据库是表的集合：
  - 每个表有唯一的名字
  - 每个表可以有多个列
  - 每个列有唯一的名字
  - 表中的一行代表的是一系列值之间的联系，即表是关系的集合
- **表**的概念和数学上**关系**的概念密切相关，这正是关系数据库名称的由来

- 常见的关系数据库
  - Oracle 、Microsoft SQL Server、IBM DB2、 GaussDB、MySQL、SQLite

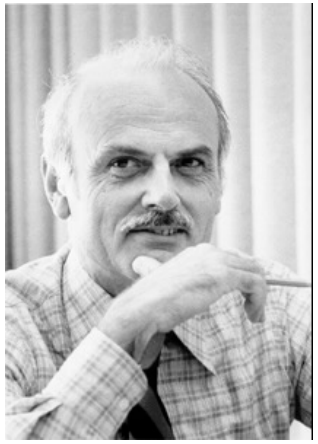
- **关系数据库**采用**关系模型**作为数据的组织方式

Student表

Sno (学号)	Sname (姓名)	Sgender (性别)	Sage (年龄)	Sdept (所在系)
2021310721	李博	男	17	CS
2021310722	赵宇	男	19	CS
2021310723	张敏	女	18	CS
2021310724	王勇	男	18	MA
2021310725	刘佳	女	17	MA

关系模型

- 1970年，IBM公司的**埃德加•科德**（Edgar F. Codd）首次提出了关系模型的概念。
  随后，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年提出了关系的巴斯科德范式
- 1981年，ACM授予埃德加·科德图灵奖，以表彰他在关系数据库研究方面的杰出贡献。



埃德加·科德

关系数据库基本概念（续）

- 关系模型的形象化类比

关系数据库基本概念（续）

➢ 关系模型术语与日常生活术语对比

关系模型术语	一般表格的术语
关系模式	表头（表格列名的集合）
关系名	表名
关系	二维表
元组	行/记录
属性	列
属性名	列名
属性值	列值
分量	一条记录中的一个列值

关系数据库基本概念（续）

- **基本概念**
  - **关系数据库**是由**表**的集合构成，**表**在关系理论里面又称为**关系**
  - 每个**表**都有唯一的名字，称之为**关系名**
  - **表**中的每一列则被称为**属性**
  - 表格中的每一行则被称为**元组**，即关系是元组的集合
  - 元组中一个属性值，在关系中称为**分量**
  - 一个有*n*个属性的关系称为***n*元关系**
- **关系**实际上就是一个二维表
  - 表格的每一行是一个元组
  - 表格的每一列是一个属性

属性

Student表

Sno (学号)	Sname (姓名)	Sgender (性别)	Sage (年龄)	Sdept (所在系)
2021310721	李博	男	17	CS
2021310722	赵宇	男	19	CS
2021310723	**张敏**	**女**	**18**	**CS**
2021310724	王勇	男	18	MA
2021310725	刘佳	女	17	MA

元组

分量

关系数据库基本概念（续）

- **关系模式**
  - **关系模式**是对关系的描述，由关系名和其属性集合组成，是相对不变的。记做$R(A_1, A_2, \cdots, A_n)$。
  - Student关系可以写为Student(Sno, Sname, Sgender, Sage, Sdept)
- **关系实例**
  - **关系实例**是关系元组集合，表示特定实例，会根据业务状态发生变化。
    - ◆ Student关系的实例有5个元组，对应5个学生。如果有新学生，则其元组数会增加。

Sno	Sname	Sgender	Sage	Sdept
2021310721	李博	男	17	CS
2021310722	赵宇	男	19	CS
2021310723	张敏	女	18	CS
2021310724	王勇	男	18	MA
2021310725	刘佳	女	17	MA

关系数据库基本概念（续）

➢ 基本概念

- 关系中每个属性都存在一个合法的取值集合，是该属性的**域**
  - ◆ 例如，Student关系中的Sage属性，其每一个取值通常是大于或等于0的
- 关系中每一个属性中的值都应该来自同一个域
  - ◆ 例如，Sage属性值应该是整数或者**空值**，不能出现日期数据或者其它类型的数据。

Sno (学号)	Sname (姓名)	Sgender (性别)	Sage (年龄)	Sdept (所在系)
2021310721	李博	男	17	CS
2021310722	赵宇	男	19	CS
2021310723	张敏	女	18	CS
2021310724	王勇	男	18	MA
2021310725	刘佳	女	17	MA

关系数据库基本概念（续）

- **超键：**如果某一属性或者某属性集合可以唯一地标识不同的元组，则称这样的属性或者属性集为**超键**。
- **候选键：**不含有多余属性的超键：一个超键去除任意一个属性就不再是超键了。
  - Student关系的5个属性中，(Sno, Sname)是一个超键，但不是候选键；Sno是候选键；
  - 如果Sname属性的取值是唯一的，则它也是候选键。
- 候选键的各个属性称为**主属性**，不包含在任何侯选键中的属性称为**非主属性**。
  - Student关系主属性是Sno。

Sno (学号)	Sname (姓名)	Sgender (性别)	Sage (年龄)	Sdept (所在系)
2021310721	李博	男	17	CS
2021310722	赵宇	男	19	CS
2021310723	张敏	女	18	CS
2021310724	王勇	男	18	MA
2021310725	刘佳	女	17	MA

关系数据库基本概念（续）

- 在某些情况，需要同时使用两个或者两个以上的属性才能唯一标识不同的元组，这种由多个属性所构成的键称为**复合键**。
  - 例如，在SC关系中，如果只用Sno或者Cno属性都不能唯一地标识不同的元组，但如果同时使用Sno和Cno属性，则可以唯一地标识不同的元组，即可唯一地区分不同学生在不同课程上的成绩。SC关系中的Sno和Cno属性的组合是其复合键。

复合键

SC表

Sno（学号）	Cno（课程号）	Grade（成绩）
2021310721	5	98
2021310722	1	87
2021310723	1	92
2021310723	5	76
2021310724	7	84
2021310725	4	95

关系数据库基本概念（续）

- 一个关关系可能有一个或多个候选键，在定义关系数据库关系表的时候，一般选择一个最合适的候选键作为**主键**，主键的不同取值必须唯一标识不同的元组。
  - 例如，Student关系的5个属性中，Sno属性可以唯一地标识不同的学生，可以选择Sno作为Student关系的主键。

Student表

Sno **(学号)**	Sname (姓名)	Sgender (性别)	Sage (年龄)	Sdept (所在系)
2021310721	李博	男	17	CS
2021310722	赵宇	男	19	CS
2021310723	张敏	女	18	CS
2021310724	王勇	男	18	MA
2021310725	刘佳	女	17	MA

关系数据库基本概念（续）

- SC关系的Sno和Cno属性分别是Student和Course关系的主键，这种一个关系模式R1中的属性是另一个关系模式R2的主键，这样的属性在R1上称作参照R2的外键，即**外键**。

Student表

Sno (学号)	Sname (姓名)	Sgender (性别)	Sage (年龄)	Sdept (所在系)
2021310721	李博	男	17	CS
2021310722	赵宇	男	19	CS
2021310723	张敏	女	18	CS
2021310724	王勇	男	18	MA
2021310725	刘佳	女	17	MA

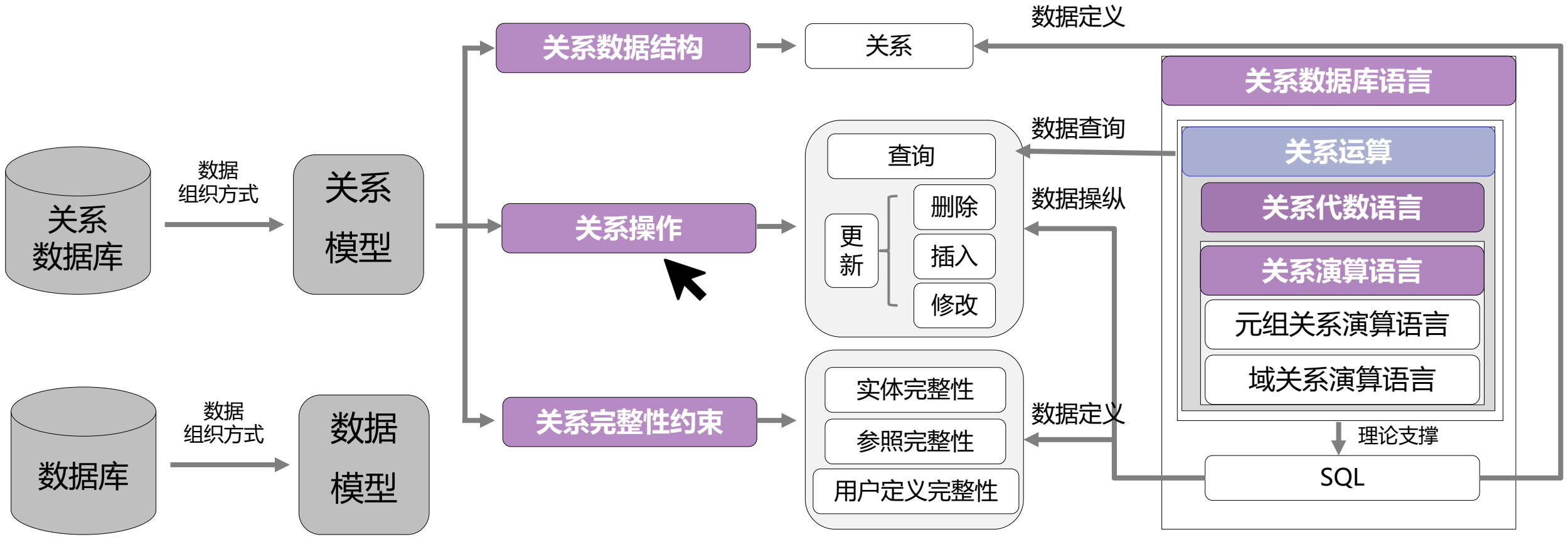
Course表

Cno (课程号)	Cname (课程名)	Cpno (先修课)	Ccredit (学分)
1	数据库	2	4
2	数据结构与算法	6	4
3	操作系统	2	3

SC表

Sno (学号)	Cno (课程号)	Grade (成绩)
2021310721	5	98
2021310722	1	87
2021310723	1	92
2021310723	5	76

目录


关系数据库
数据
组织方式
关系
模型
数据库
数据
组织方式
数据
模型
关系数据结构
关系操作
关系完整性约束
关系
查询
更新
删除
插入
修改
实体完整性
参照完整性
用户定义完整性
数据定义
数据查询
数据操纵
数据定义
关系数据库语言
关系运算
关系代数语言
关系演算语言
元组关系演算语言
域关系演算语言
理论支撑
SQL

关系操作

- 基于关系模型，常见的**关系操作**主要包括了：
  - 查询操作（Query）和数据更新操作。
  - 数据更新包括了删除（Delete）、插入（Insert）和修改（Update）
- 查询操作是关系操作的核心部分。
- 关系操作可以通过关系运算进行表示。
- 关系运算是埃德加·科德博士所提出的关系模型的一部分，本质上是一种形式化的关系查询语言，可以用于从数据库中查询数据。

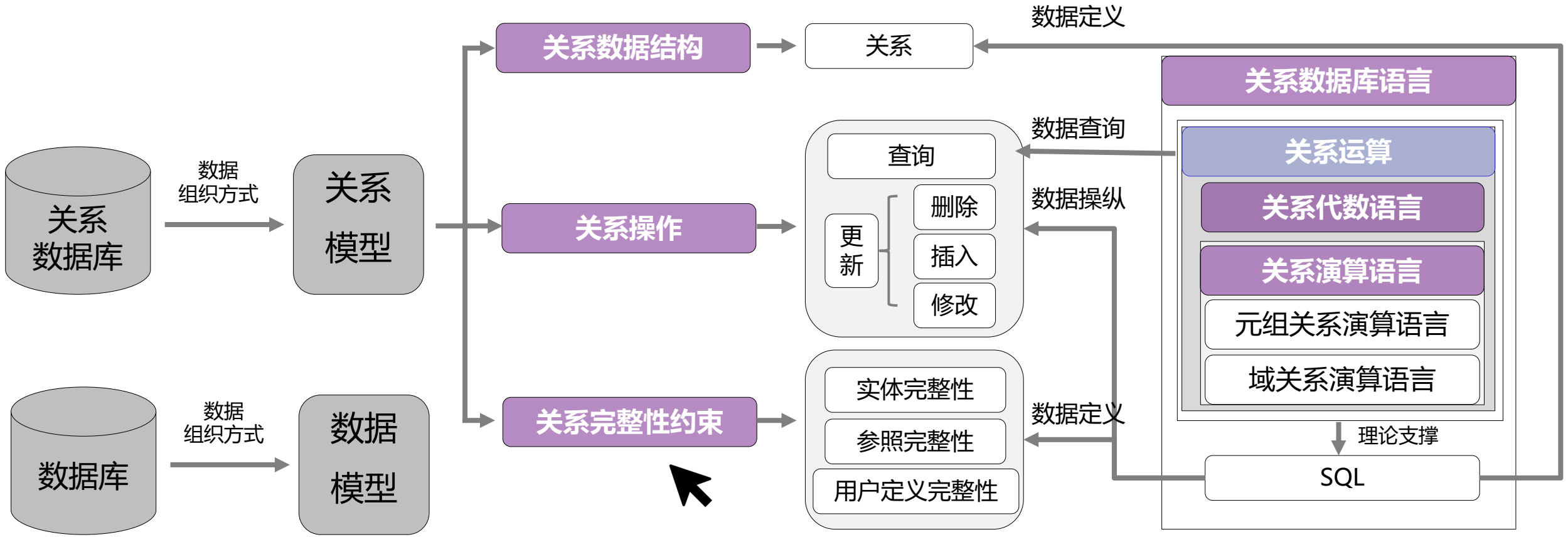
关系数据库语言

- **关系数据库语言**包括了关系运算和SQL
- 关系运算包含
  - 关系代数（relational algebra）
    - ◆关系代数是一种过程化查询语言，通过描述对关系的运算来表达查询、获取数据；
  - 关系演算则是非过程化查询语言，通过描述想要获取的数据的信息来获取数据（不需要给出运算过程）
    - ◆关系演算可以分为元组关系演算和域关系演算两种语言。
- 为了方便用户查询处理关系数据，定义了结构化查询语言SQL 来操作处理关系数据

关系运算和SQL的关系

- SQL、关系代数和关系演算都是在关系模型基础上用于表达数据操作的语言
- SQL是用户与关系数据库直接交互的途径
- SQL是一种介于关系代数和关系演算之间的语言，它和关系演算一样是非过程化语言
- 关系代数是过程化的，是SQL的理论基础
- 执行SQL需要通过关系代数理论进行优化

目录


关系数据结构
关系
数据定义
关系数据库语言
关系运算
关系代数语言
关系演算语言
元组关系演算语言
域关系演算语言
理论支撑
SQL
数据查询
查询
数据操纵
更新
删除
插入
修改
关系操作
关系完整性约束
数据定义
实体完整性
参照完整性
用户定义完整性
关系数据库
数据组织方式
关系模型
数据库
数据组织方式
数据模型

关系完整性约束

- 现实世界中仍然有许多事物很难用现有的方法进行建模。因此，我们在信息世界中建模这些事物的时候，通常需要附加一定的**约束。**
- 在关系模型中，有三类完整性约束：
  - 实体完整性约束（entity integrity constraint）
  - 参照完整性约束（referential integrity constraint）
  - 用户定义完整性约束（user-defined integrity constraint）

常见的约束类型

- **完整性约束**
  - 实体完整性约束（主键约束）
  - 参照完整性约束（外键约束）
  - 用户定义完整性约束
- **其它约束**
  - 非空约束
  - 唯一约束
  - 域约束

常见的约束类型（续）

➢ 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实体和参照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

实体完整性约束

➢ 实体完整性约束

- 规范设计的关系数据库中的每个元组都应该是唯一且可区分的。例如，在学生关系数据库中，不应该出现两个表示学生实体的相同元组，并且在学生关系数据库中的学生实体之间是可以相互区分的。
- 实体完整性约束可以保证关系数据库的上述特性，**实体完整性约束主要是通过在关系表中实施主键取值约束**，来保证关系中的每个元组可以被唯一识别。

实体完整性约束（续）

➢ 实体完整性约束规则

- 如果键K是关系R的主键，则K不能取空值；
- 如果关系R的主键K是复合键，则构成复合键的多个属性均不能取空值；
- 如果键K是关系R的主键，则K的取值不能在R中重复。

➢ 实体完整性检查规则

- 检查关系R的主键K的值是否唯一，如果不唯一或者为空，则拒绝插入或者修改元组数据；
- 如果关系R的主键K是复合键，检查K的各个属性是否为空。如果有一个为空或者复合键不唯一，则拒绝插入或修改元组数据。

实体完整性约束（续）

- **实体完整性约束示例**

复合主键的属性取值为空

Sno（学号）	Cno（课程号）	Grade（成绩）	Note(备注)
2021310721	5	98	
2021310722		87	
2021310723	1	92	有进步

违反实体完整性约束规则（2）的SC关系表

主键的取值重复

Sno（学号）	Cno（课程号）	Grade（成绩）	Note(备注)
2021310721	5	98	
2021310722	1	87	
2021310722	1	92	有进步

违反实体完整性约束规则（3）的SC关系表

Sno（学号）	Cno（课程号）	Grade（成绩）	Note(备注)
2021310721	5	98	
2021310722	1		
2021310723	1	92	有进步

符合实体完整性约束的SC关系表

参照完整性约束

- 实体完整性约束可以看做是实体集内部之间的约束，参照完整性约束更多地指代**实体集之间关系的约束**。
- 例如，考虑下图中学校与学生之间**一对多的“录取”关系**
  - 一所学校可以录取多名学生，如果某一学生被某所学校录取，那么录取该学生的学校必须出现在学校实体集中。
  - 一个实体集中的属性取值参照另一个相关实体集。即，实体集之间存在相互引用、相互约束的情况。

参照完整性约束（续）

➢ **参照完整性约束规则**：

- **（1）删除规则**：如果一个实体E被另一个实体F引用，可以禁止删除该实体E。例如，某所学校录取了1000名学生，则不能将该学校删除；如果要删除该学校，则必须同时把隶属于该学校的1000名学生的信息也删除。（被参照实体后删除）
- **（2）插入规则**：如果一个实体F引用了另一个实体E，那么实体E必须存在数据库中。例如，某位学生被某所学校录取，那么该学校必须已经存在学校实体集中；否则，需要先将学校实体集中的信息存到数据库后，才能将该学生的信息插入到数据库中。（被参照实体先插入）

用户定义完整性约束

➢ 用户定义完整性约束：

- 用户定义完整性约束则是用户根据具体的数据库应用场景，设置的具体的约束条件，用户定义完整性约束可以反映数据的特殊语义要求。
  - ◆例如，在Student关系中，假设学校在录取学生的时候明确表明每位学生都必要有姓名和年龄信息，则Student关系中的Sname和Sage属性均不能取空值。

Student表

Sno (学号)	Sname (姓名)	Sgender (性别)	Sage (年龄)	Sdept (所在系)
2021310721	李博	男	17	CS
2021310722	赵宇	男	19	CS
2021310723	张敏	女	18	CS
2021310724	王勇	男	18	MA
2021310725	刘佳	女	17	MA

其它约束

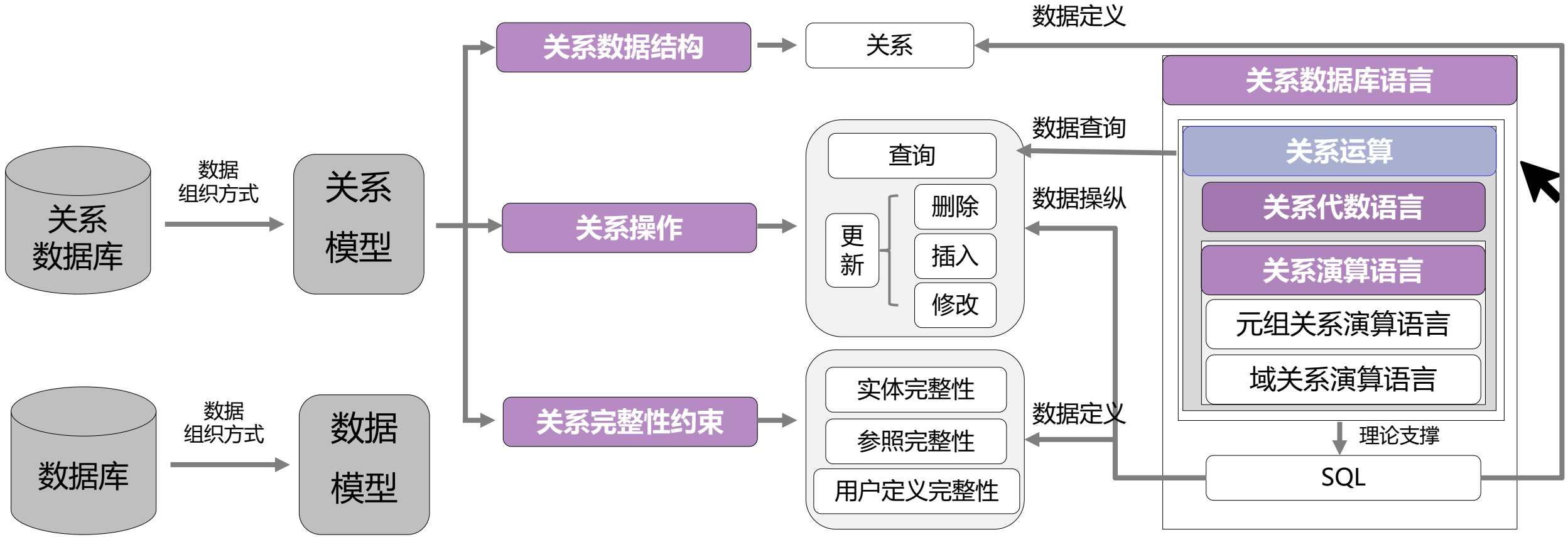
- **非空约束**（not null constraint）
  - 指关系中某一属性的取值不能为空值。
  - 例如，如果对Student关系中的Sage属性增加了非空约束，则在增加一个新的学生元组的时候，其Sage属性必须要有一个非空的取值，否则数据库系统将会报错。

- **唯一约束**（unique constraint）
  - 指关系中某一属性的取值不能重复的约束。
  - 例如给Student关系中的Sdept属性加上唯一约束后，则表明学生的系别不能重复出现。
  - 唯一约束与主键约束都可以保证关系中某一属性的取值不会重复出现。
  - 不同的是，主键约束只能作用于一个关系中的一个属性且该属性的取值不能为空值，而唯一约束可以作用于一个关系中的多个属性，且该属性的取值可以为空值但不能重复。

其它约束（续）

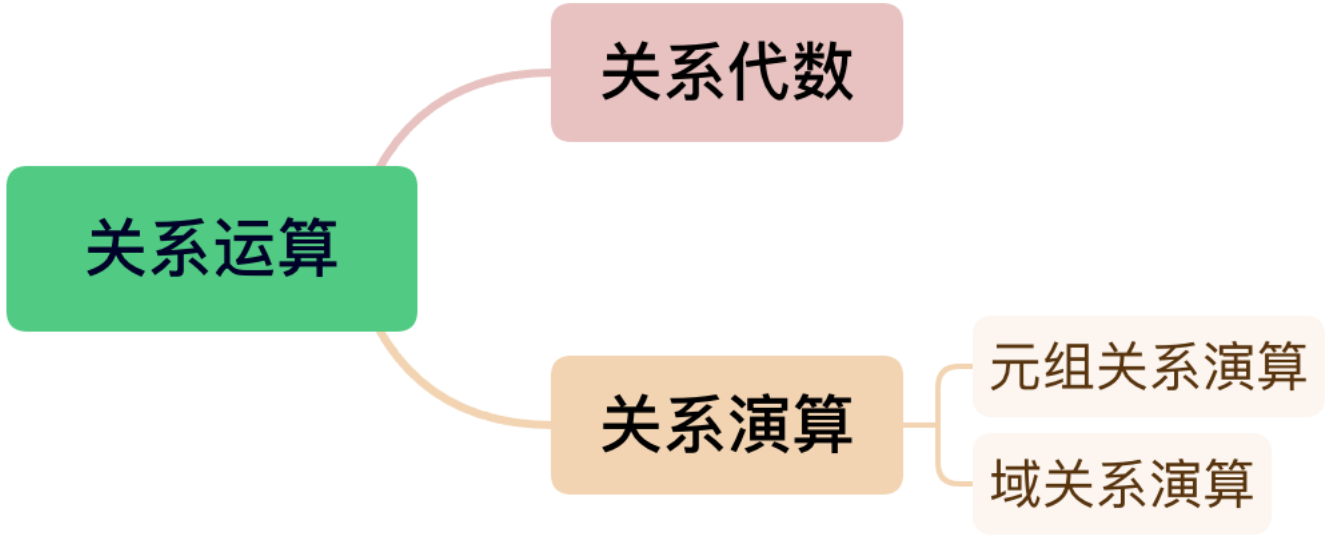
- **自增长约束**（auto_increment constraint）
  - 指一个关系中的属性的取值自增长，例如一个元组的ID
  - 一般用于自增列
- **默认值约束**（default constraint）
  - 指一个关系中的属性的取值为默认值，例如默认值为NULL或者0
- **检查约束**（check constraint）
  - 指一个关系中的属性的取值必须满足一个指定条件的约束
  - 例如，在Student关系中，Sage属性的取值范围是大于等于0

目录


关系数据结构
关系
数据定义
关系数据库语言
关系运算
关系代数语言
关系演算语言
元组关系演算语言
域关系演算语言
理论支撑
SQL
关系
数据库
数据
组织方式
关系
模型
关系操作
查询
更新
删除
插入
修改
数据查询
数据操纵
数据库
数据
组织方式
数据
模型
关系完整性约束
实体完整性
参照完整性
用户定义完整性
数据定义

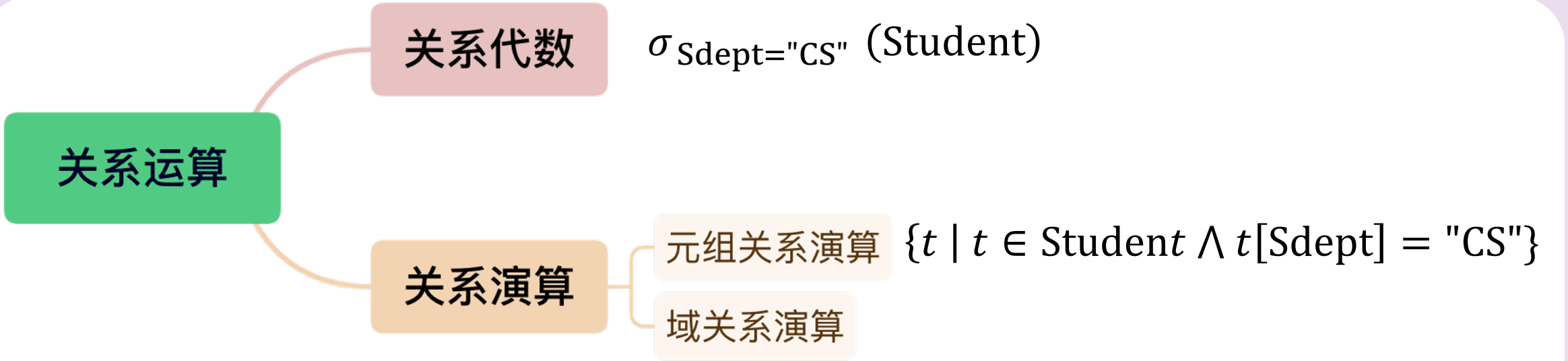
关系运算

- 关系运算是埃德加·科德博士所提出的关系模型的一部分，本质上是一种形式化的关系查询语言，可以用于从数据库中查询数据。



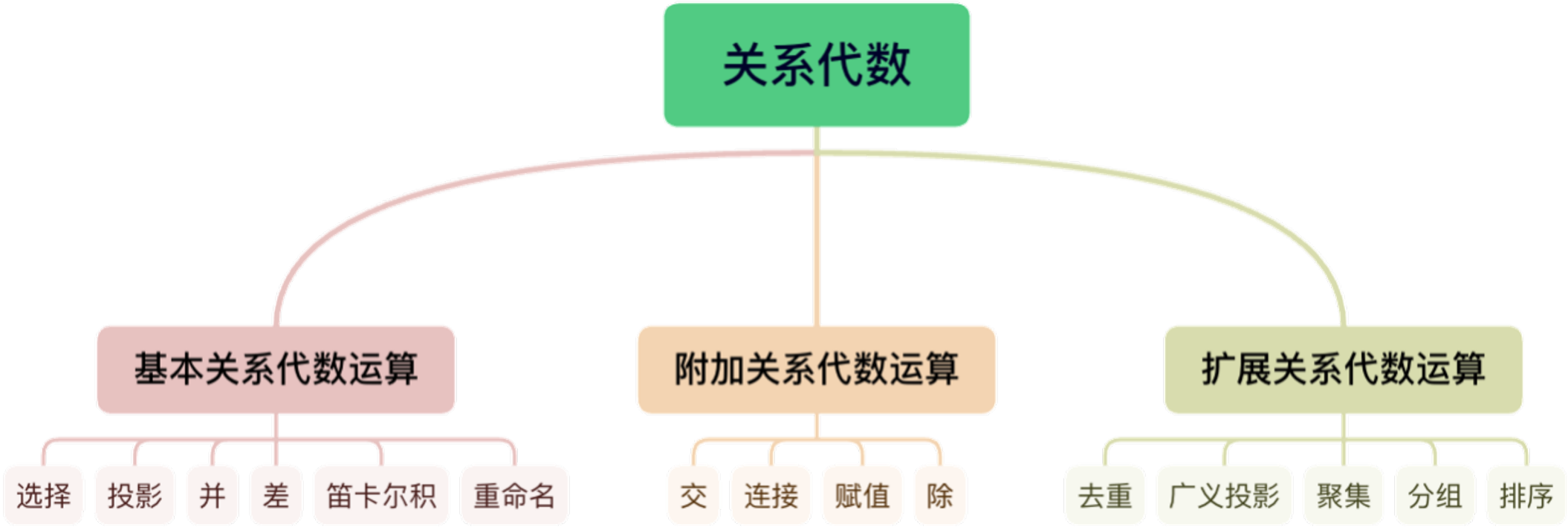
关系运算

- 关系运算是埃德加·科德博士所提出的关系模型的一部分，本质上是一种形式化的关系查询语言，可以用于从数据库中查询数据。
- **查询计算机系的学生信息：**



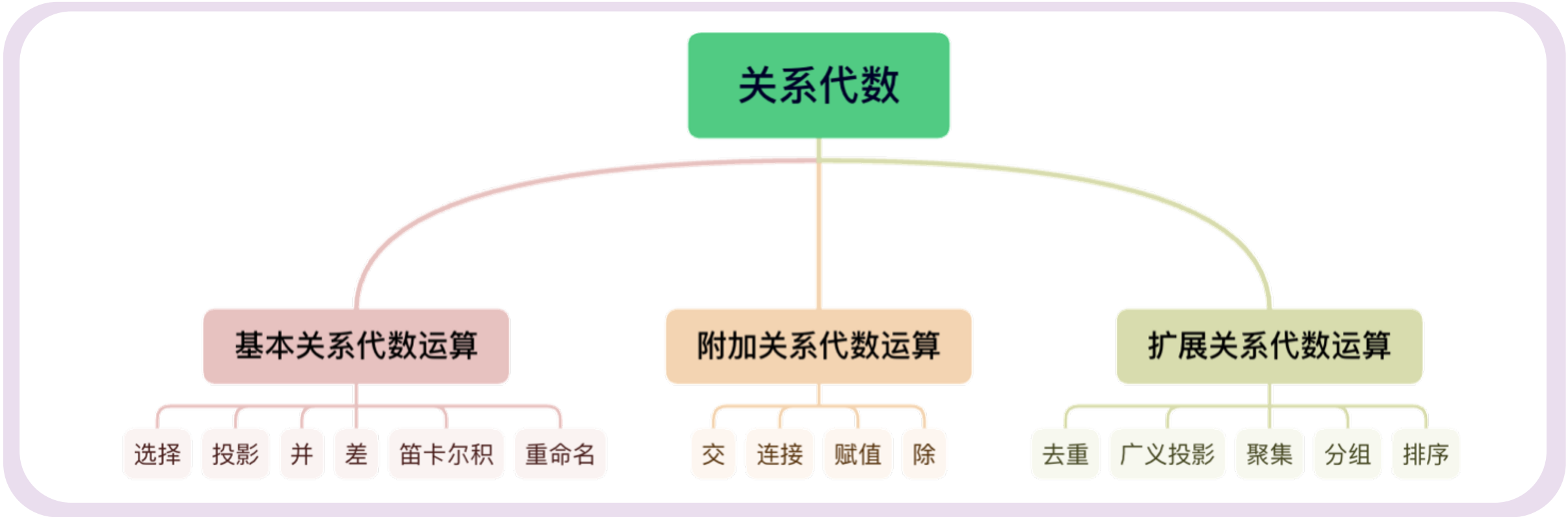
关系代数

- 关系代数定义了一个关系数据的运算的集合
- 关系运算以一个或者两个关系为输入；输出一个新的关系作为运算结果。
- 关系是一个以元组为元素的多重集合（multiset）- 可能包含重复元素
- 关系代数运算本质上是对多重集合的运算



基本关系代数运算

- **基本关系代数运算**包括选择、投影、并、差、笛卡尔积和重命名。
- 选择、投影、重命名为一元运算；并、差、笛卡尔积为二元运算。
- **基本关系代数运算**可以用来表达**任何传统关系代数运算。**



选择运算

- 选择运算（σ）可以从关系R中获取满足条件的元组：

$$\sigma_p(R) = \{t | t \in R \wedge p(t) = \text{True}\}$$

- p 为选择谓词
  - p是由逻辑运算符与（∧）、或（∨）、非（¬）连接的若干原子表达式构成的公式
  - 原子表达式的形式为：$X\ \theta\ Y$
    - X，Y：属性名、常量，或者函数值
    - θ：比较运算符，包括=、>、<、≥、≤、≠

选择运算

- 【例2.1】查询计算机系所有学生的信息。

$$\sigma_{\text{Sdept="CS"}}(\text{Student})$$

- 查询结果为：

Sno	Sname	Sgender	Sage	Sdept
2021310721	李博	男	17	CS
2021310722	赵宇	男	19	CS
2021310723	张敏	女	18	CS

选择运算

- 【例2.2】查询计算机系年龄大于等于18的学生的信息，以及所有数学系的学生信息。

$$\sigma_{(\text{Sdept}=\text{"CS"} \wedge \text{Sage}\geq 18) \vee \text{Sdept}=\text{"MA"}}(\text{Student})$$

- 查询结果为：

Sno	Sname	Sgender	Sage	Sdept
2021310722	赵宇	男	19	CS
2021310723	张敏	女	18	CS
2021310724	王勇	男	18	MA
2021310725	刘佳	女	17	MA

投影运算

- 投影运算（Π）可以从关系R中获取某些列组成新的关系：

$$\Pi_{A_1,A_2,\ldots,A_k}(R) = \{t[A_1, A_2, \ldots, A_k] \mid t \in R\}$$

- $A_1, A_2, \ldots, A_k$为R中的属性列
- 返回R中元组在$A_1, A_2, \ldots, A_k$列上的值
- 删除重复元组

投影运算

- 【例2.3】查询计算机系所有学生的学号、姓名。

$$\Pi_{\text{Sno, Sname}}\left(\sigma_{\text{Sdept="CS"}}\left(\text{Student}\right)\right)$$

- 查询结果为：

Sno	Sname
2021310721	李博
2021310722	赵宇
2021310723	张敏

并运算

- 并运算（∪）返回两个关系R和S中元组取并集的结果：

$$R \cup S = \{ t \mid t \in R \vee t \in S \}$$

- R和S中属性个数要相同
- R和S中的属性应存在一一对应关系
- R中每个属性的域和S中对应属性的域要相同

- Union：去重
- Union all：不去重

并运算

- 【例2.4】查询所有17或18岁学生的信息。

$$\sigma_{\text{Sage}=17}(\text{Student}) \cup \sigma_{\text{Sage}=18}(\text{Student})$$

- 查询结果为：

Sno	Sname	Sgender	Sage	Sdept
2021310721	李博	男	17	CS
2021310723	张敏	女	18	CS
2021310724	王勇	男	18	MA
2021310725	刘佳	女	17	MA

差运算

- 差运算（−）返回在关系R中但是不在关系S中的元组集合：

$$R - S = \{ t \mid t \in R \wedge t \notin S \}$$

- R和S中属性个数要相同
- R和S中的属性应存在一一对应关系
- R中每个属性的域和S中对应属性的域要相同

差运算

- 【例2.5】查询计算机系中的未成年同学的信息。

$$\sigma_{\text{Sdept="CS"}}(\text{Student}) - \sigma_{\text{Sage}\geq 18}(\text{Student})$$

- 查询结果为：

Sno	Sname	Sgender	Sage	Sdept
2021310721	李博	男	17	CS

笛卡尔积运算

- 笛卡尔积运算（×）返回关系R中元组和关系S中的元组做笛卡尔积的结果：

$$R \times S = \{(t, q) \mid t \in R \ \wedge q \in S\}$$

- (t, q)为R中元组t和S中元组q拼接在一起得到的元组
- $R \times S$中有$|R| \times |S|$个元组

笛卡尔积运算

- 【例2.6】R中有3个元组，S中有2个元组，$R\times S$中有$3\times 2=6$个元组。

A	B
a	1
a	2
b	3

(a) R

C	D
x	4
y	2

(b) S

A	B	C	D
a	1	x	4
a	1	y	2
a	2	x	4
a	2	y	2
b	3	x	4
b	3	y	2

(c) $R\times S$

重命名运算

- 重命名运算（ρ）将关系R重命名为关系S：

$$\rho_{S(A_1,A_2,\ldots,A_n)}(R)$$

- 同时将各属性按照从左到右的顺序重命名为$A_1, A_2, \ldots, A_n$
- $\rho_S(R)$：只修改关系名，不修改属性名

重命名运算

- 【例2.7】将学生选课表重命名为StudentCourse表，同时将Grade属性重命名为Score属性

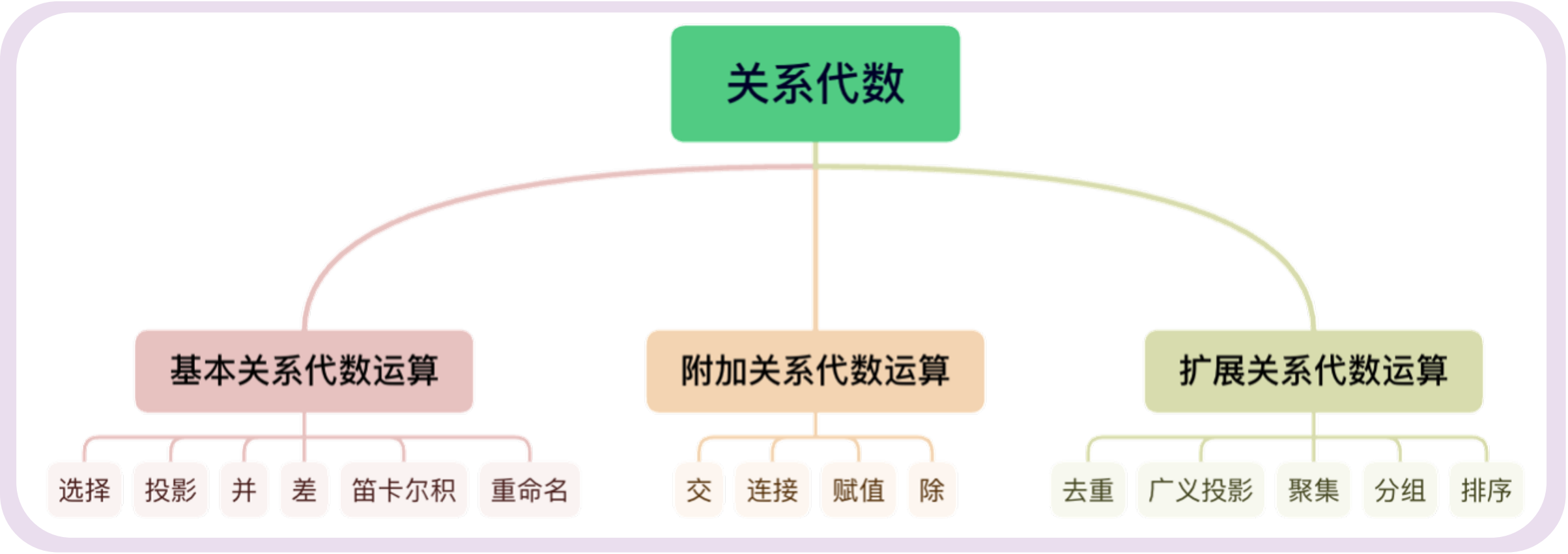
$$\rho_{\text{StudentCourse (Sno,Cno,Score)}}(\text{SC})$$

- 查询结果为：

Sno	Cno	Score
2021310721	5	98
2021310722	1	87
2021310723	1	92
2021310723	5	76
2021310724	7	84
2021310725	4	95

附加关系代数运算

- 基本关系代数运算写出来的表达式比较冗长，因此定义了附加关系代数运算
- 附加关系代数运算是由基本关系代数运算导出的运算，可以简化表达式
- 附加关系代数运算和基本关系代数运算的**表达能力相同**



交运算

- 交运算（∩）返回两个关系R和S中元组取交集的结果：

$$R \cap S = \{ t \mid t \in R \wedge t \in S \}$$

- R和S中属性个数要相同
- R和S中的属性应存在一一对应关系
- R中每个属性的域和S中对应属性的域要相同

- 交运算（∩）可以通过差运算（－）来表示：

$$R \cap S = R - (R - S)$$

交运算

- 【例2.8】查询计算机系年龄大于等于18的学生的信息：

$$\sigma_{\text{Sdept="CS"}}(\text{Student}) \cap \sigma_{\text{Sage}\geq 18}(\text{Student})$$

- 查询结果为：

Sno	Sname	Sgender	Sage	Sdept
2021310722	赵宇	男	19	CS
2021310723	张敏	女	18	CS

连接运算

- 连接运算（$\bowtie$）返回关系R和S笛卡尔积运算结果中满足一定条件的元组：

$$R \bowtie_p S = \{(t,q) \mid t \in R \ \wedge q \in S \ \wedge \ p\big((t,q)\big) = \text{True}\}$$

- p 为选择谓词
  - p是由逻辑运算符与（$\wedge$）、或（$\vee$）、非（$\neg$）连接的若干原子表达式构成的公式
  - 原子表达式的形式为：$R.X \ \theta \ S.Y$
    - X是R的属性，Y是S的属性，X和Y所属域具有相同的数据类型
    - θ：比较运算符，包括=、>、<、≥、≤、≠

- 连接运算（$\bowtie$）可通过组合笛卡尔积运算（$\times$）和选择运算（σ）来表示

连接运算

- 【例2.9】对学生表和学生选课表进行连接运算，连接的条件为学生表中的Sno列和学生选课表中的Sno列的值相等

$$\text{Student} \bowtie_{\text{Student.Sno}=\text{SC.Sno}} \text{SC}$$

- 查询结果为：

Student.Sno	Student.Sname	Student.Sgender	Student.Sage	Student.Sdept	SC.Sno	SC.Cno	SC.Grade
2021310721	李博	男	17	CS	2021310721	5	98
2021310722	赵宇	男	19	CS	2021310722	1	87
2021310723	张敏	女	18	CS	2021310723	1	92
2021310723	张敏	女	18	CS	2021310723	5	76
2021310724	王勇	男	18	MA	2021310724	7	84
2021310725	刘佳	女	17	MA	2021310725	4	95

- 当θ是“=”时，该连接运算称为**等值连接**，例2.9即为等值连接

连接运算

- **自然连接**是一种特殊的等值连接
- **自然连接**将连接条件指定为R和S中属性名相同的列做等值连接，因此p可省略
- 【例2.10】对学生表和学生选课表进行自然连接运算：

- 查询结果为：

$$\text{Student} \bowtie \text{SC}$$

Sno	Sname	Sgender	Sage	Sdept	Cno	Grade
2021310721	李博	男	17	CS	5	98
2021310722	赵宇	男	19	CS	1	87
2021310723	张敏	女	18	CS	1	92
2021310723	张敏	女	18	CS	5	76
2021310724	王勇	男	18	MA	7	84
2021310725	刘佳	女	17	MA	4	95

连接运算

- **外连接**是连接运算的扩展，可以处理缺失值
- **左外连接**（R⟕S）会保留左边关系R的所有元组，对于R中的元组，若在S中没有在同名属性上取值相同的元组，会用空值来填充S中的属性
- **右外连接**（R⟖S）会保留右边关系S的所有元组，对于S中在R中不存在同名属性上取值相同的元组，会用空值来填充R中的属性
- **全外连接**（R⟗S）的查询结果是左外连接和右外连接查询结果的并集
- 【例2.11】左外连接、右外连接和全外连接运算示例

A	B
a	1
b	1
b	3

(a) R

B	C
2	i
1	j
4	k

(b) S

A	B	C
a	1	j
b	1	j
b	3	NULL

(c) $R \bowtie S$

A	B	C
NULL	2	i
a	1	j
b	1	j
NULL	4	k

(d) $R \bowtie S$

A	B	C
a	1	j
b	1	j
b	3	NULL
NULL	2	i
a	1	j
b	1	j
NULL	4	k

(e) $R \bowtie S$

赋值运算

- 赋值运算（←）将←右侧的关系代数表达式结果赋值给←左侧的关系变量：

$$T \leftarrow E$$

- T为临时关系变量，E为关系代数表达式

- 【例2.12】对学生表和学生选课表进行连接运算，连接的条件为学生表中的Sno列和学生选课表中的Sno列的值相等，并将连接结果赋值给关系变量result

$$\text{temp} \leftarrow \text{Student} \times \text{SC}$$

$$\text{result} \leftarrow \sigma_{\text{Student.Sno=SC.Sno}}(\text{temp})$$

- 赋值运算可以分解复杂的关系代数表达式，使查询变得简单

除运算

- 设$R(A_1, A_2, \ldots, A_m, B_1, B_2, \ldots, B_n)$和$S(B_1, B_2, \ldots, B_n)$是两个关系，则$R \div S$的属性为$A_1, A_2, \ldots, A_m$，且：

$$R \div S = \{ t \mid t \in \Pi_{A_1, A_2, \ldots, A_m}(R) \wedge (\forall q \in S, (t, q) \in R) \}$$

- 除运算（$\div$）会返回R中在属性$A_1, A_2, \ldots, A_m$上的元组t，其中元组t和关系S中的任意元组q的组合都会出现在关系R中
- 如果S中有R中没有的属性，则无法进行除运算

- 除运算（$\div$）可以用投影运算（Π）和笛卡尔积运算（$\times$）表示：

$$R \div S = \Pi_{A_1, A_2, \ldots, A_m}(R) - \Pi_{A_1, A_2, \ldots, A_m}\left(\left(\Pi_{A_1, A_2, \ldots, A_m}(R) \times S\right) - R\right)$$

例如SC ÷ Course，找到选择所有课程的同学

除运算

- 【例2.13】除运算示例

A	B
a	1
a	2
a	3
b	1
b	2

(a) R

B
1
2
3

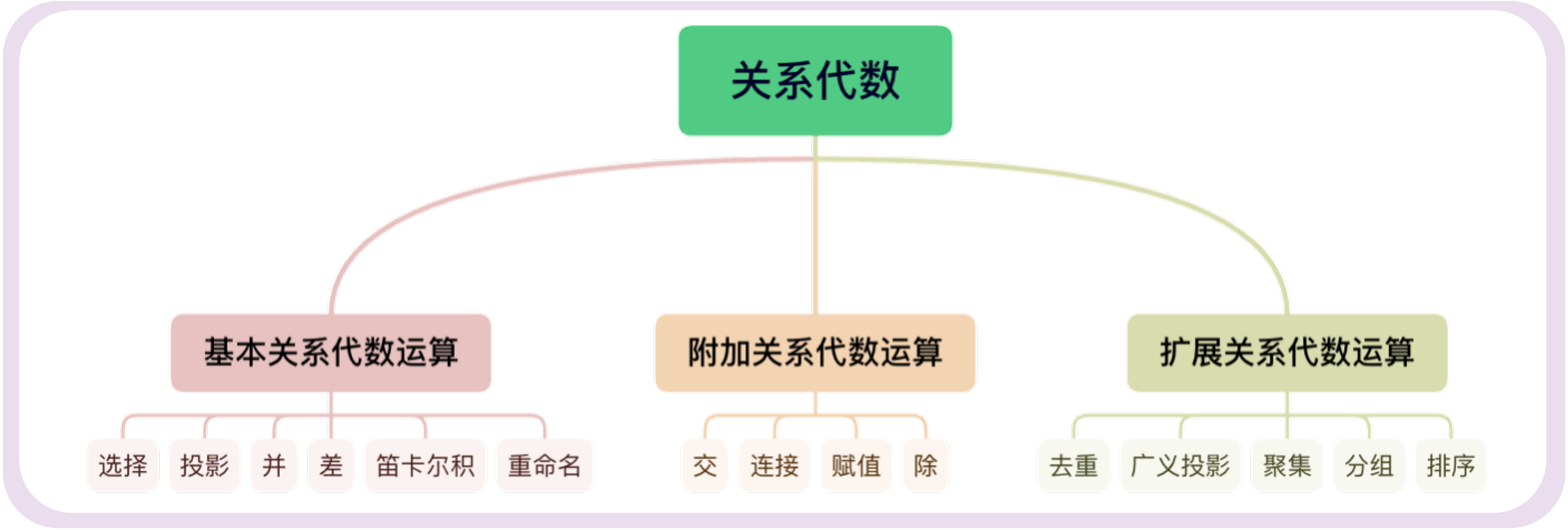
(b) S

A
a

(c) $R \div S$

扩展关系代数运算

- **基本关系代数运算代数和附加关系代数运算**不能表达SQL中的某些运算
- **扩展关系代数运算**定义了使用**基本关系代数运算和附加关系代数运算无法实现的运算**



去重运算

- 去重运算（δ）可以将关系R中的重复元组去除，并返回去除重复元组后的关系：

$$\delta(R)$$

- 【例2.14】查询所有系的信息

$$\delta\left(\Pi_{\text{Sdept}}(\text{Student})\right)$$

- 查询结果为：

Sdept
CS
MA

广义投影运算

- 广义投影运算（Π）允许在投影列表中使用算术运算和字符串函数等来对投影运算进行扩展：

$$\Pi_{F_1,F_2,\ldots,F_k}(R)$$

- R为关系
- $F_1, F_2, \ldots, F_k$ 为包含常量、变量（R 中列）、运算符（算术运算符，逻辑运算符，关系运算符）、函数等的多个表达式
- 例如2023-age得到出生年份

广义投影运算

- 【例2.15】查询所有学生的学号，姓名，出生年份。

$$\Pi_{Sno,Sname,2021-Sage}(Student)$$

- 查询结果为：

Sno	Sname	2021-Sage
2021310721	李博	2004
2021310722	赵宇	2002
2021310723	张敏	2003
2021310724	王勇	2003
2021310725	刘佳	2004

聚集运算

- 聚集运算（$\mathcal{G}$）可以查询关系R按某些列的值聚集在一起的结果：

$$\mathcal{G}_{F_1(A_1),F_2(A_2),\ldots,F_k(A_k)}(R)$$

- $A_1, A_2, \ldots, A_k$为R中的属性列
- F_i为作用在属性A_i上的聚集函数（$1 \leq i \leq k$）
- 常见的聚集函数包括count, sum, avg, min, max等
- 例如求课程平均分

聚集运算

- 【例2.16】查询学生总人数。

$$\mathcal{G}_{\mathrm{count(Sno)}}(\mathrm{Student})$$

查询结果为：5

聚集运算

- 【例2.17】查询学生总人数，以及学生平均年龄。

$$\mathcal{G}_{\text{count(Sno), avg(Sage)}}(\text{Student})$$

- 查询结果为：

count(Sno)	avg(Sage)
5	17.8

分组运算

- 分组运算首先对关系R中的元组按照某些列的值进行分组，然后在各组上应用聚集运算：

$$_{G_1,G_2,\ldots,G_l}\mathcal{G}_{F_1(A_1),F_2(A_2),\ldots,F_k(A_k)}(R)$$

- $G_1, G_2, \ldots, G_l$是用来分组的一系列属性，为R中的列，在这些列上取值都相同的元组将被分到同一组
- $A_1, A_2, \ldots, A_k$为R中的属性列，F_i为作用在属性A_i上的聚集函数（$1 \leq i \leq k$）
- 查询结果中会包含$G_1, G_2, \ldots, G_l$和$F_1(A_1), F_2(A_2), \ldots, F_k(A_k)$列

分组运算

➢ 【例2.18】查询选课学生总人数。

$$\mathcal{G}_{\text{count\_distinct(Sno)}}(\text{SC})$$

- 其中count_distinct函数在对Sno列去重后进行计数，查询结果为5。
- 也可以先对Sno列使用去重运算，再使用count函数进行计数，可以得到相同结果：

$$\mathcal{G}_{\text{count(Sno)}}(\delta(\Pi_{\text{Sno}}(\text{SC})))$$

分组运算

- 【例2.19】查询所有选课学生的学号及平均分。

$$A_{\text{Sno}}\,\mathcal{G}_{\text{avg(Grade)}}(\text{SC})$$

- 查询结果为：

Sno	avg(Grade)
2021310721	98
2021310722	87
2021310723	84
2021310724	84
2021310725	95

分组运算

- 【例2.20】查询各个系的男生和女生人数。

$$_{A}\,_{\text{Sdept,Sgender}}\,\mathcal{G}_{\text{count(Sno)}}(\text{Student})$$

- 查询结果为：

Sdept	Sgender	count(Sno)
CS	男	2
CS	女	1
MA	男	1
MA	女	1

排序运算

- 排序运算（τ）将关系R中的元组按照一列或多列的值排序：

$$\tau_{A_1,A_2,\ldots A_k}(R)$$

- $A_1, A_2, \ldots A_k$是用来排序的列
- 首先将R中的元组按照A_1的值排序，对于A_1列取值相同的元组，按照A_2的值排序，以此类推

排序运算

- 【例2.21】将学生表按照年龄排序，对于年龄相同的元组，按照学号排序

$$\tau_{\text{Sage,Sno}}(\text{Student})$$

- 查询结果为：

Sno	Sname	Sgender	Sage	Sdept
2021310721	李博	男	17	CS
2021310725	刘佳	女	17	MA
2021310723	张敏	女	18	CS
2021310724	王勇	男	18	MA
2021310722	赵宇	男	19	CS

关系代数表达式

- 关系代数表达式可以是以下形式（其中E_1和E_2均为关系代数表达式）：

（1）(E_1)；

（2）$\sigma_p(E_1)$；

（3）$\Pi_{A_1,A_2,\ldots,A_k}(E_1)$；

（4）$E_1 \cup E_2$；

（5）$E_1 - E_2$；

（6）$E_1 \times E_2$；

（7）$\rho_{S(A_1,A_2,\ldots,A_n)}(E_1)$；

（8）$E_1 \cap E_2$；

（9）$E_1 \bowtie_p E_2$；

（10）$E_1 \div E_2$；

（11）$\delta(E_1)$；

（12）$\Pi_{F_1,F_2,\ldots,F_k}(E_1)$；

（13）$\mathcal{G}_{F_1(A_1),F_2(A_2),\ldots,F_k(A_k)}(E_1)$；

（14）$_{G_1,G_2,\ldots,G_l}\mathcal{G}_{F_1(A_1),F_2(A_2),\ldots,F_k(A_k)}(E_1)$。

- 关系代数表达式中各运算符号的计算顺序为从左到右，括号具有最高优先级

关系代数表达式

- 【例2.22】查询选修了1号课程的学生学号及成绩。
- $\Pi_{\text{Sno, Grade}}(\sigma_{\text{Cno="1"}}(\text{SC}))$
- 该表达式的执行步骤为：
  - 步骤①：执行选择运算，选择学生选课表中课程号为1的元组。即 $\sigma_{\text{Cno="1"}}(\text{SC})$。
  - 步骤①：执行投影运算，获取步骤1中得到的学生选课表中课程号为1的元组的学号、成绩信息。即执行表达式 $\Pi_{\text{Sno, Grade}}(\sigma_{\text{Cno="1"}}(\text{SC}))$。
- 查询结果为：

Sno	Grade
2021310722	87
2021310723	92

关系代数表达式

- 【例2.23】查询选修了1号课程的学生学号、姓名及成绩

$$\Pi_{\text{Sno, Sname, Grade}}(\sigma_{\text{Cno="1"}}(\text{SC} \bowtie \text{Student}))$$

- 该表达式的执行步骤为：

步骤①：执行自然连接运算，即执行表达式 $\text{SC} \bowtie \text{Student}$

步骤②：执行选择运算，即执行表达式 $\sigma_{\text{Cno="1"}}(\text{SC} \bowtie \text{Student})$

步骤③：执行投影运算，即执行表达式 $\Pi_{\text{Sno, Sname, Grade}}(\sigma_{\text{Cno="1"}}(\text{SC} \bowtie \text{Student}))$

关系代数表达式

- 【例2.23】查询选修了1号课程的学生学号、姓名及成绩

$$\Pi_{\text{Sno, Sname, Grade}}((\sigma_{\text{Cno="1"}}(\text{SC})) \bowtie \text{Student})$$

- 该表达式的执行步骤为：

  - 步骤①：执行选择运算，即执行表达式$\sigma_{\text{Cno="1"}}(\text{SC})$
  - 步骤②：执行自然连接运算，即执行表达式 $(\sigma_{\text{Cno="1"}}(\text{SC})) \bowtie \text{Student}$
  - 步骤③：执行投影运算，即执行表达式 $\Pi_{\text{Sno, Sname, Grade}}((\sigma_{\text{Cno="1"}}(\text{SC})) \bowtie \text{Student})$

关系代数表达式

- 【例2.23】查询选修了1号课程的学生学号、姓名及成绩。

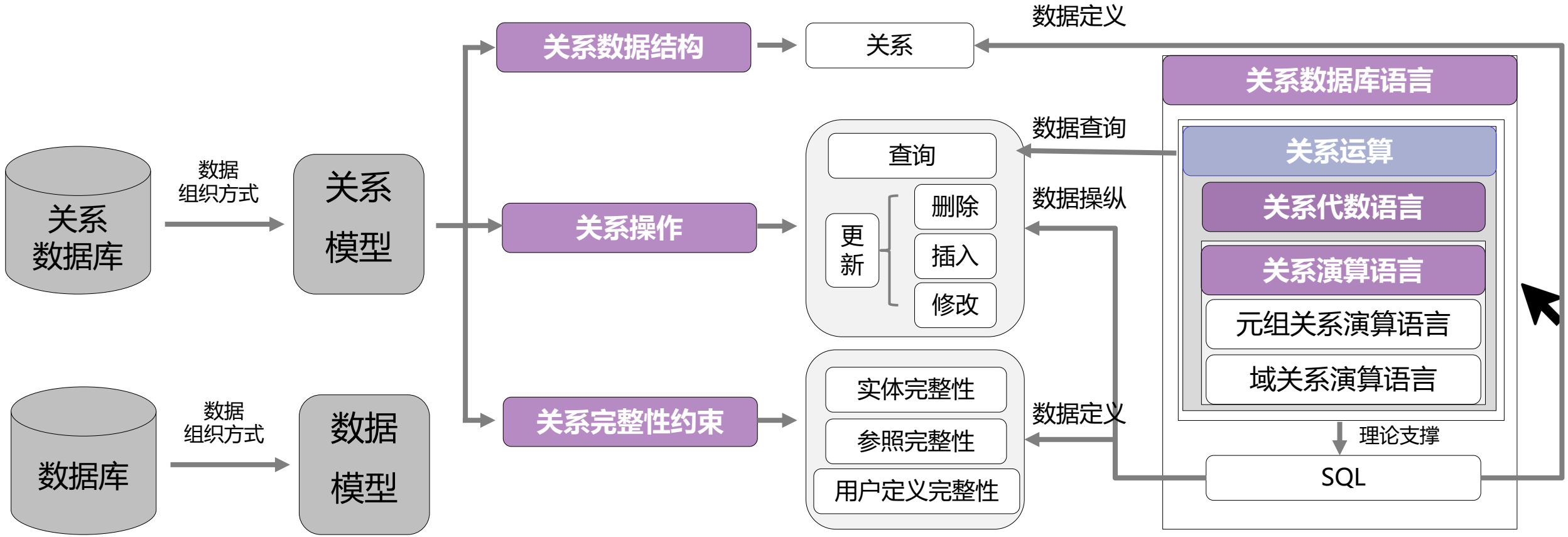
$$\Pi_{\text{Sno, Sname, Grade}}(\sigma_{\text{Cno="1"}}(\text{SC} \bowtie \text{Student}))$$

等价

$$\Pi_{\text{Sno, Sname, Grade}}((\sigma_{\text{Cno="1"}}(\text{SC})) \bowtie \text{Student})$$

- 上述两个表达式是等价的，但是运算的执行顺序不同
- 在连接前先执行选择运算可以减少参与连接的元组数，因此执行效率会提高。
- 如何将一个关系代数表达式重写为等价的更高效的关系代数表达式是一个重要的问题，常见的操作包括选择下推、投影下推等（详见第11章）

目录


关系数据结构
关系
数据定义
关系数据库语言
关系运算
关系代数语言
关系演算语言
元组关系演算语言
域关系演算语言
理论支撑
SQL
数据查询
查询
数据操纵
更新
删除
插入
修改
关系完整性约束
实体完整性
参照完整性
用户定义完整性
数据定义
关系数据库
数据
组织方式
关系
模型
关系操作
数据库
数据
组织方式
数据
模型

元组关系演算

- **关系代数**是一种**过程化查询语言**，通过在关系代数表达式中指定一个运算的序列，并按照此序列依次执行，可以得到查询的结果。
- **元组关系演算**是**非过程化查询语言**，只描述想要获取数据的信息，不描述获取信息的具体过程。
- 元组关系演算表达式的定义为：$\{t \mid P(t)\}$
  - 上述表达式返回所有使得公式P为真的元组t的集合
  - P由原子公式组成，原子公式可以是以下形式之一：
    - （1）$t \in R$，其中t是元组变量，R是关系；
    - （2）$t[x]\ \Theta\ s[y]$，其中t和s是元组变量，x是t所属的关系的属性，y是s所属的关系的属性，Θ是比较运算符；
    - （3）$t[x]\ \Theta\ c$，其中t，x，Θ同上，c是属性x的域中的常量。

元组关系演算

- 【例2.28】查询年龄大于等于18的学生的学号、姓名、性别、年龄、所在系

$$\{t \mid t \in \text{Student} \wedge t[\text{Sage}] \geq 18\}$$

- 查询结果为：

Sno	Sname	Sgender	Sage	Sdept
2021310722	赵宇	男	19	CS
2021310723	张敏	女	18	CS
2021310724	王勇	男	18	MA

元组关系演算

- 【例2.29】查询年龄大于等于18的学生的学号。

$$\{t \mid \exists s \in \text{Student}(s[\text{Sage}] \geq 18 \wedge t[\text{Sno}] = s[\text{Sno}])\}$$

- 上述语句的含义为：
  - 查找满足以下条件的元组t的集合，使得关系Student中存在元组s，s在属性Sage上的取值大于等于18且s和t在属性Sno上取值相同
  - s和t均为元组变量，但是元组变量t仅定义在属性Sno上

- 查询结果为：

Sno
2021310722
2021310723
2021310724

域关系演算

- **域关系演算**也是**非过程化查询语言**。
- **域关系演算**使用属性域中取值的**域变量**来代替元组关系演算中的**元组变量**

$$\{< x_1, x_2, \ldots, x_k > \mid P(x_1, x_2, \ldots, x_k)\}$$

- 域关系演算表达式的定义为：
  - 其中$x_1, x_2, \ldots, x_k$均为域变量，P是由原子公式组成的公式
  - 上述表达式返回所有使得公式P为真的域变量$x_1, x_2, \ldots, x_k$组成的元组的集合
  - 原子公式可以是以下形式之一：
    - （1）$< x_1, x_2, \ldots, x_k > \in R$，其中$R$是包含$k$个属性的关系，$x_1, x_2, \ldots, x_k$为域变量或域常量；
    - （2）$x\ \Theta\ y$，其中x和y是域变量，Θ是比较运算符（包括>，<，=，≥，≤，≠）；
    - （3）$x\ \Theta\ c$，其中x，Θ同上，c是x所属属性的属性域中的常量。

域关系演算

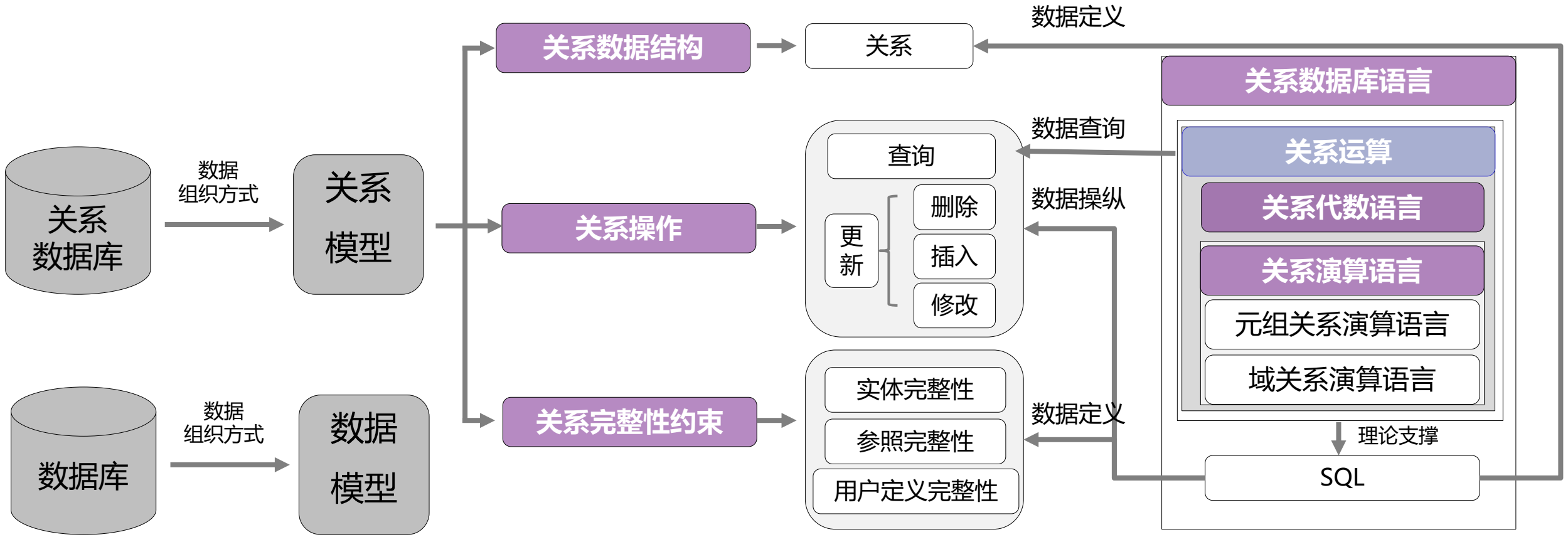
- 【例2.34】使用域关系演算重写【例2.28】，即查询年龄大于等于18的学生的学号、姓名、性别、年龄、所在系。

$$\{< no, name, gen, age, dept > \mid < no, name, gen, age, dept > \in Student \land age \geq 18 \}$$

- 【例2.35】使用域关系演算重写【例2.29】，即查询年龄大于等于18的学生的学号：

$$\{< no > \mid \exists name, gen, age, dept(< no, name, gen, age, dept > \in Student \land age \geq 18) \}$$

本章知识框架


关系数据库
数据组织方式
关系模型
数据库
数据组织方式
数据模型
关系数据结构
关系操作
关系完整性约束
关系
查询
更新
删除
插入
修改
实体完整性
参照完整性
用户定义完整性
数据定义
数据查询
数据操纵
数据定义
关系数据库语言
关系运算
关系代数语言
关系演算语言
元组关系演算语言
域关系演算语言
理论支撑
SQL

本章小结

➢ 关系模型

- 数据库中一种重要的数据模型
- 关系数据库的数据组织方式和重要理论基础
- 任何关系数据库系统都离不开关系模型

➢ 关系模型有三个组成部分：

- 关系数据结构
- 关系操作
- 关系完整性约束

本章小结（续）

- **关系模型的数据结构就是关系，是一种由行和列组成的二维表，表中的数据是描述客观事物的内容**
- **关系操作包括了查询、删除、插入和修改**
- **关系完整性约束是一类重要的约束**
  - 包括了实体完整性约束、参照完整性约束和用户定义完整性约束
  - 实体完整性约束是指通过关系的主键约束，以保证关系中的每个元组都可以唯一识别
  - 参照完整性约束是指关系之间的联系的约束，以保证关系之间数据的一致性
  - 用户定义完整性约束值根据数据库应用场景，用户设置的具体的约束条件，可以反映数据的特殊语义要求

本章小结（续）

- **为了更好地查询关系数据库的内容，科学家提出了关系运算的概念和理论。**
- **关系运算本质上是一种形式化的关系查询语言**
  - 可以用于从数据库中查询数据，具体包括了关系代数和关系演算
- **关系代数定义了一个关系代数运算的集合**
- **关系代数运算的输入为一个或两个关系，运算结果为一个新的关系**
  - 每一个关系代数运算都有与之相对应的关系代数符号
  - 选择（σ）、投影（Π）、并（$\cup$）、差（$-$）、笛卡尔积（$\times$）、重命名（ρ）是六种基本的关系代数运算
  - 另外，交（$\cap$）、连接（$\bowtie$）、赋值（$\leftarrow$）、除（$\div$）是由基本关系代数运算组合得到的附加关系代数运算，因此这四种运算并不增加关系代数的表达能力
  - 扩展关系代数运算可以实现一些使用基本关系代数运算和附加关系代数运算无法实现的功能，包括去重（δ）、广义投影（Π）、聚集（$\mathcal{G}$）、分组（$\mathcal{G}$）、排序（τ）等

本章小结（续）

- **多个关系运算的组合可以形成一个关系代数表达式**
  - 关系代数表达式可以用来表达想要从数据库中获取的数据信息
  - 关系代数表达式中各运算符号的计算顺序为从左到右，括号的优先级最高
- **关系代数是过程化查询语言**
- **元组关系演算和域关系演算是非过程化查询语言**
- **关系代数通过描述在数据库中执行一系列操作的过程来获取数据**
- **元组关系演算和域关系演算则通过描述想要获取的数据的信息**