数据库系统原理

陈岭

浙江大学计算机学院

8 数据库设计和E-R模型

- 实体集
- 联系集
- 设计问题
- 映射约束
- 键
- E-R图
- 扩展的E-R特性
- E-R模式设计
- E-R模式到表的转换

实体集

- 数据库可被建模为：
  - 实体集合
  - 实体间联系
- 实体是客观存在的对象并且与其他对象可区分
  - 例如： 特定的人，公司，事件，植物
- 实体具有属性
  - 例如：人具有姓名和地址
- 实体集是相同类型的实体的集合，他们具有相同的性质
  - 例如：所有人的集合，所有公司的集合

实体集

- 例，实体集 *instructor*

属性名称 ⇨

ID	*name*	*dept_name*	*salary*
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000

属性值

一个实体

instructor

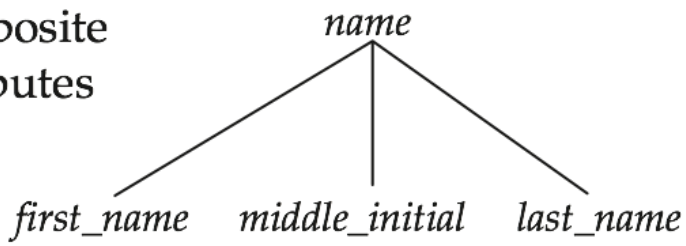
实体集

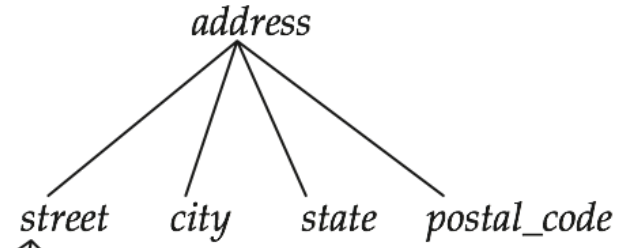
- 实体用一个属性集合来表示，即实体集中所有成员都具有的描述性特性
- 域：属性允许取值的集合
- 属性种类：
  - 简单属性与复合属性
  - 单值属性与多值属性
    - 例，多值属性：*phone-numbers*
  - 派生属性
    - 可由其他属性计算得到
    - 例，给定出生日期可计算出年龄
    - 基属性或存储属性

实体集

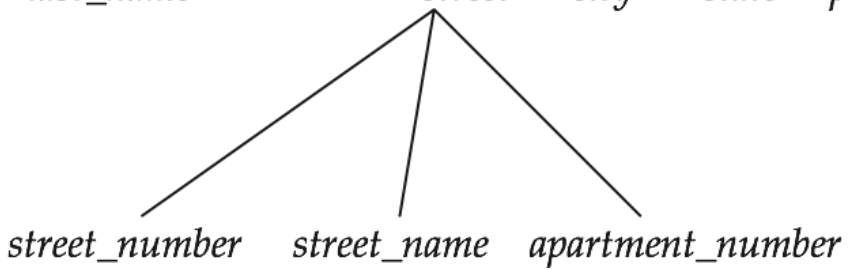
- 复合属性

复合 composite
属性 attributes



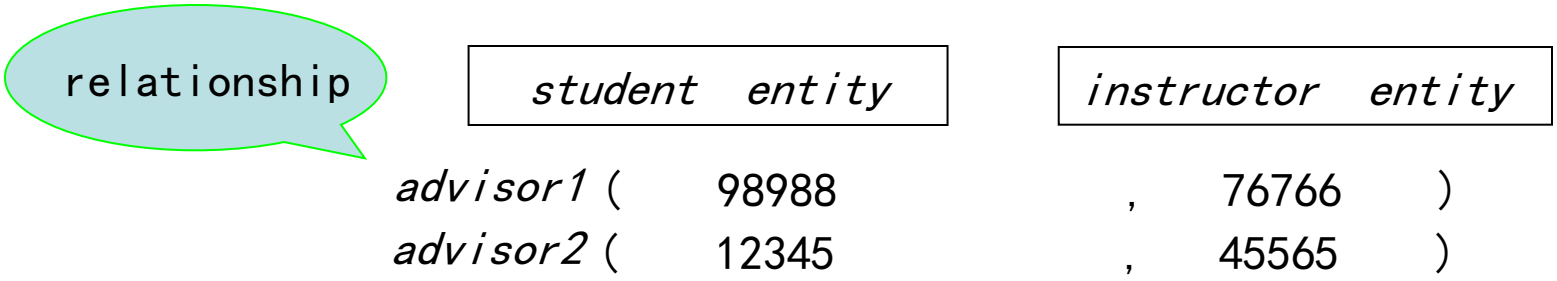


分量 component
属性 attributes



联系集

- 联系是指多个实体之间的相互关联



- 联系集是相同类型联系的集合
  - 一个联系集包含多个同类联系（或联系实例，relationship instance）
  - 一个联系集表示二个或多个实体集之间的关联

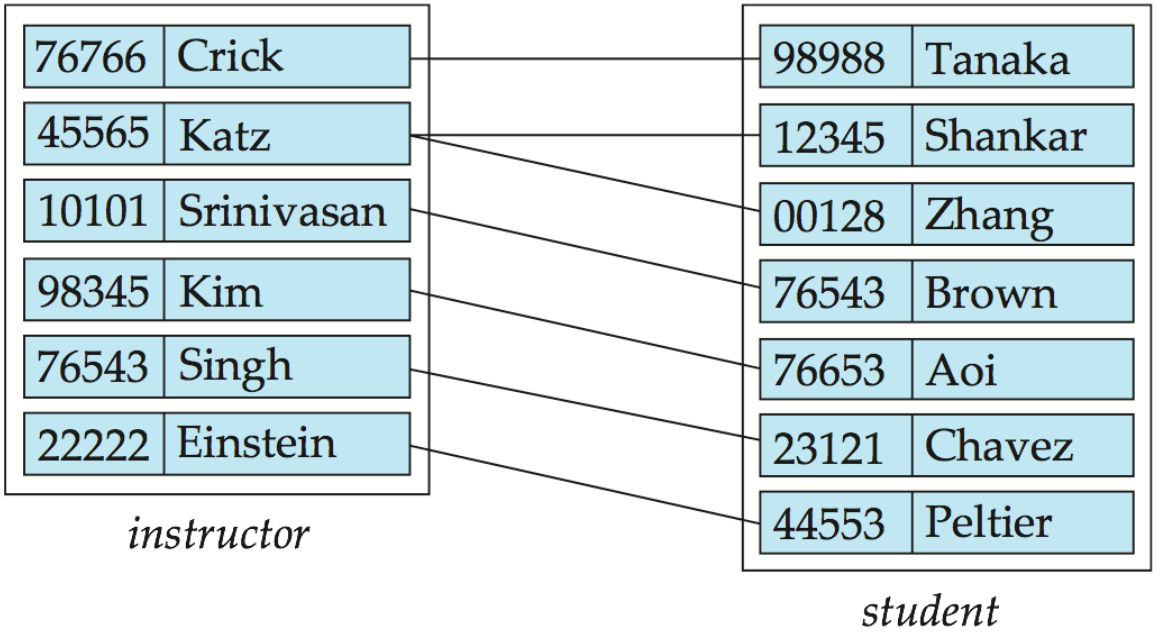
联系集

- 正规地说，联系集是 $n \geq 2$ 个实体集上的数学关系，每个实体取自一实体集
  - $\{(e_1, e_2, \cdots e_n) \mid e_1 \in E_1, e_2 \in E_2, \cdots, e_n \in E_n\}$
  - 其中 $(e_1, e_2, \cdots, e_n)$ 是一个联系，E_i 为实体集
  - 例，(98988, 76766) ∈ advisor，

    其中，98988 ∈ student，76766 ∈ instructor

联系集

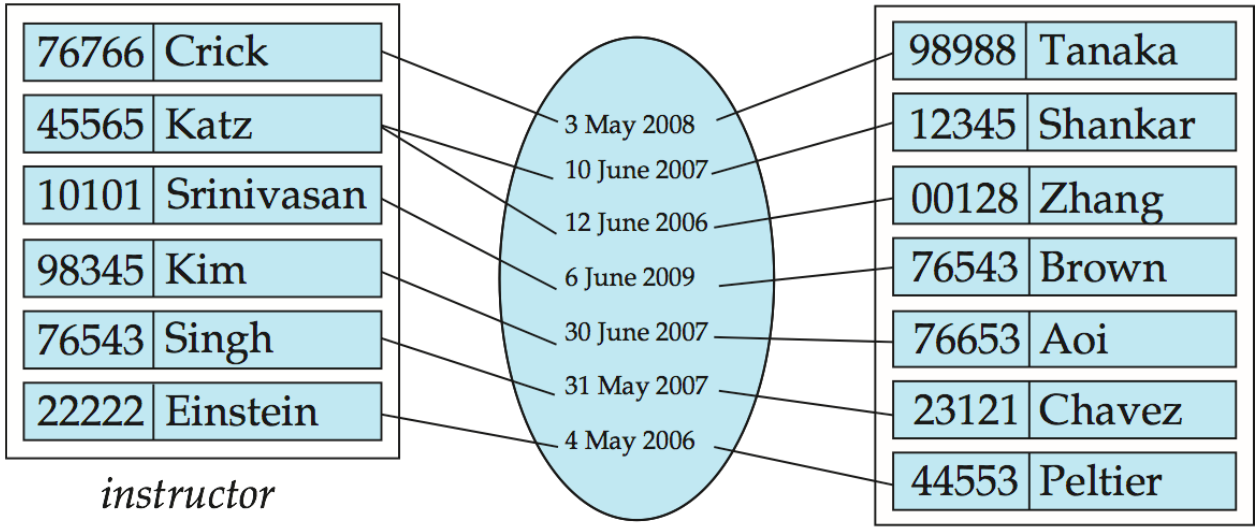
- 联系集*advisor*



advisor (*s_ID*, *i_ID*)

联系集

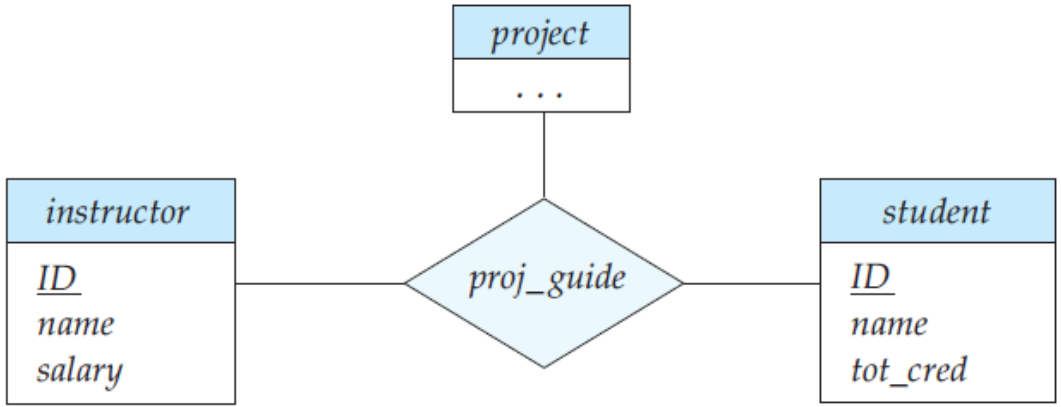
- 联系集也可具有属性
- 例如，实体集*instructor*和*student*之间的*advisor*联系集可具有属性*date*



advisor (*s_ID*, *i_ID*, *date*)

联系集的度

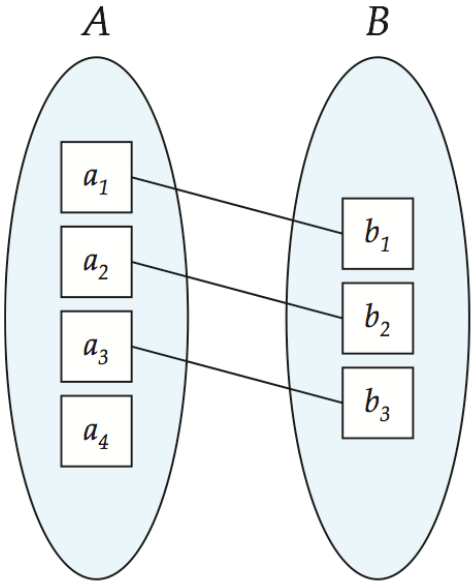
- 参加联系的实体集的个数
- 涉及两个实体集的联系集称为二元的（二元联系）
- 联系集可以涉及多于两个的实体集
  - 例，假设一个student在每个项目上最多只能有一位导师，如下图，包含三个实体集*instructor*、*student*和*project*（三元联系）
  - 多于两个实体集之间的联系较少见，数据库系统中的联系集一般多为二元的



映射基数

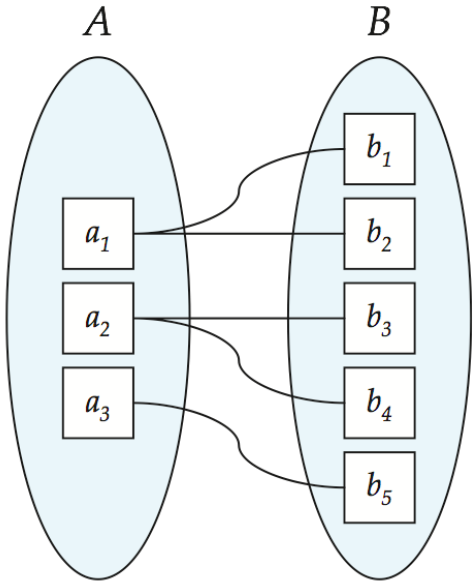
- 表达可与一个实体通过联系集进行关联的其他实体的个数
- 描述二元联系集最有用
- 二元联系集的映射基数有以下几种情况：
  - 一对一，如：就任总统（总统，国家）
  - 一对多，如：分班情况（班级，学生）
  - 多对一，如：就医（病人，医生）
  - 多对多，如：选课（学生，课程）

映射基数



(a)

一对一

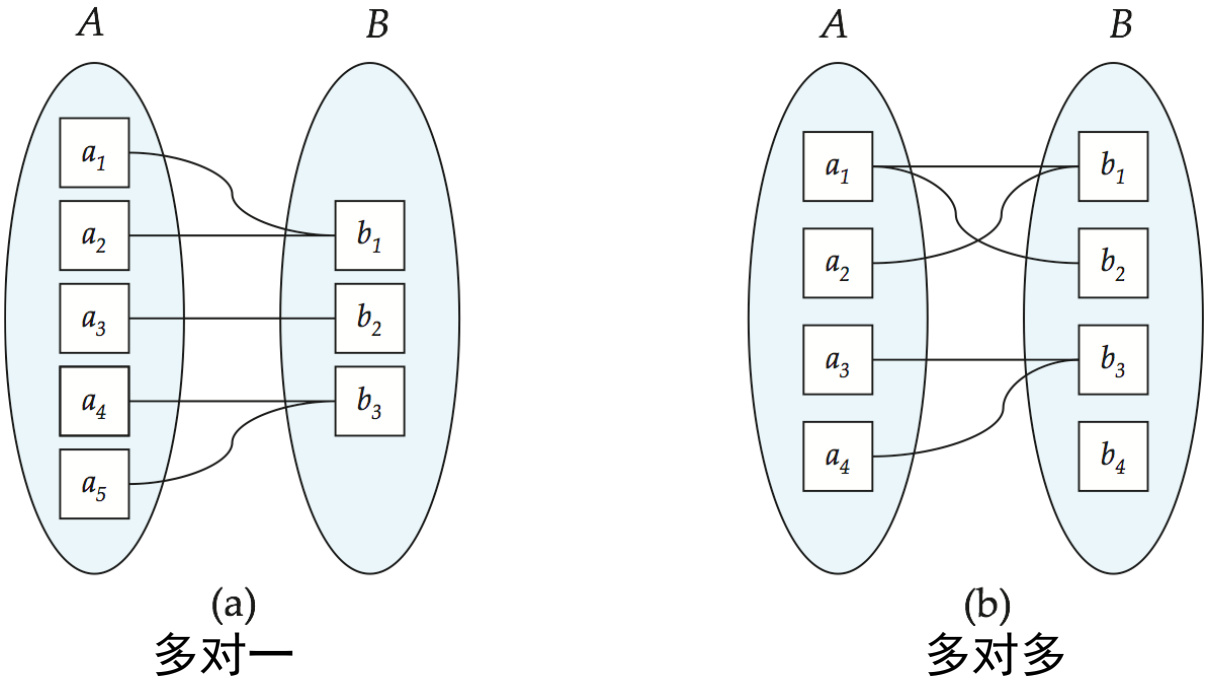


(b)

一对多

注意：A和B中的某些元素可能未被映射到另一集合中的任何元素

映射基数



注意：A和B中的某些元素可能未被映射到另一集合中的任何元素

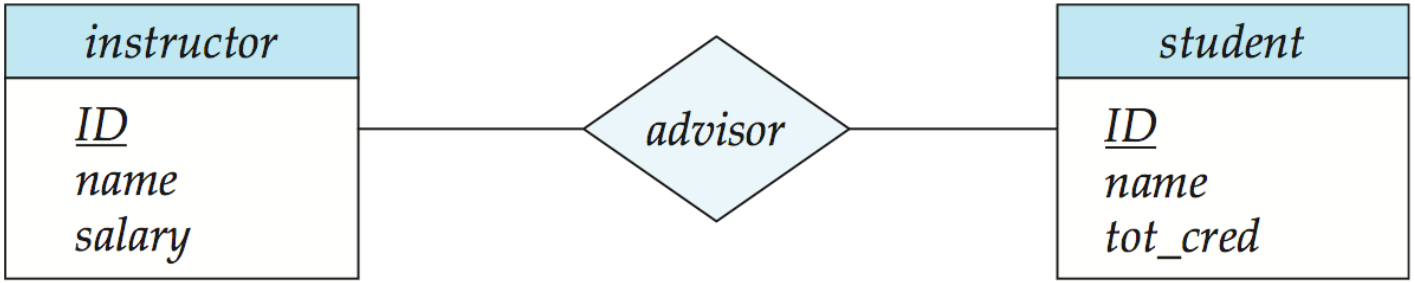
键、码

- 实体集的超码是能够唯一标识每个实体的一个或多个属性
- 候选码是实体集的最小超码
- 候选码可能存在多个，我们只会选择一个候选码作为主码或主键
- 例，*instructor*（*ID*, *name*, *dept_name*, *salary*）
  - 候选码：*ID*
  - 超码：{*ID*}，{*ID*, *name*}，{*ID*, …}

键、码

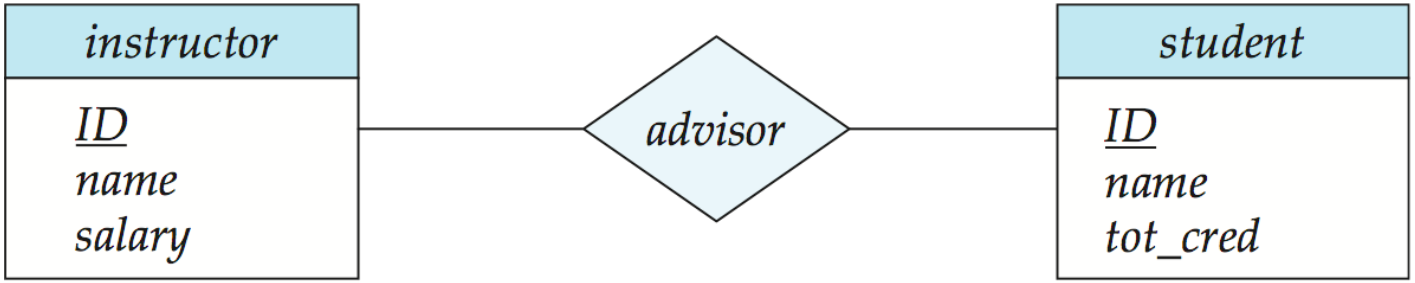
- 参与一个联系集的各实体集的码的组合，构成该联系集的超码
  - (*s_ID, i_ID*) 是*advisor*的超码
  - 注意：这意味着一对实体在一个联系集上最多有一个联系
- 联系集的候选码依赖于联系集的映射基数（1:1，1:n，m:n）
- 在选择主键时，如果有多个候选码，需要考虑关系集合的语义
  - 例，作为码的属性不能为空，值不应常变

E-R图



- 分成两部分的矩形代表实体集。有阴影的第一部分包含实体集的名字，第二部分包含实体集中所有属性的名字
- 菱形代表联系集
- 未分割的矩形代表联系集的属性。构成主码的属性以下划线标明
- 线段将实体集连接到联系集

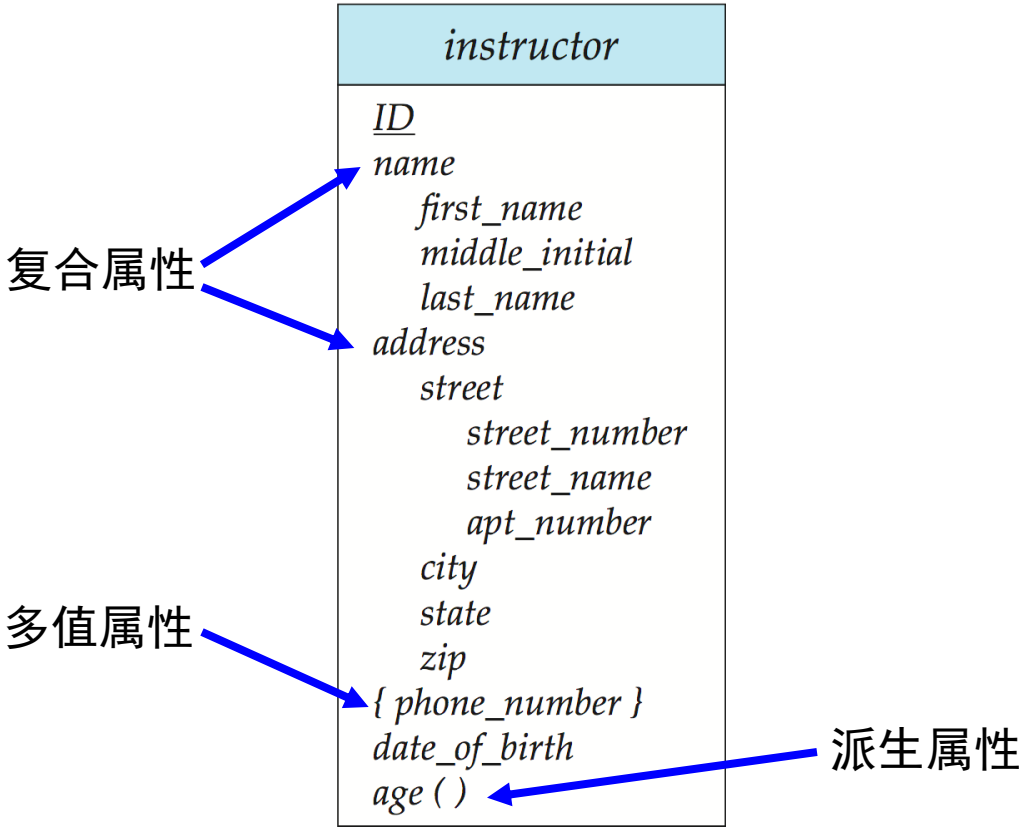
E-R图



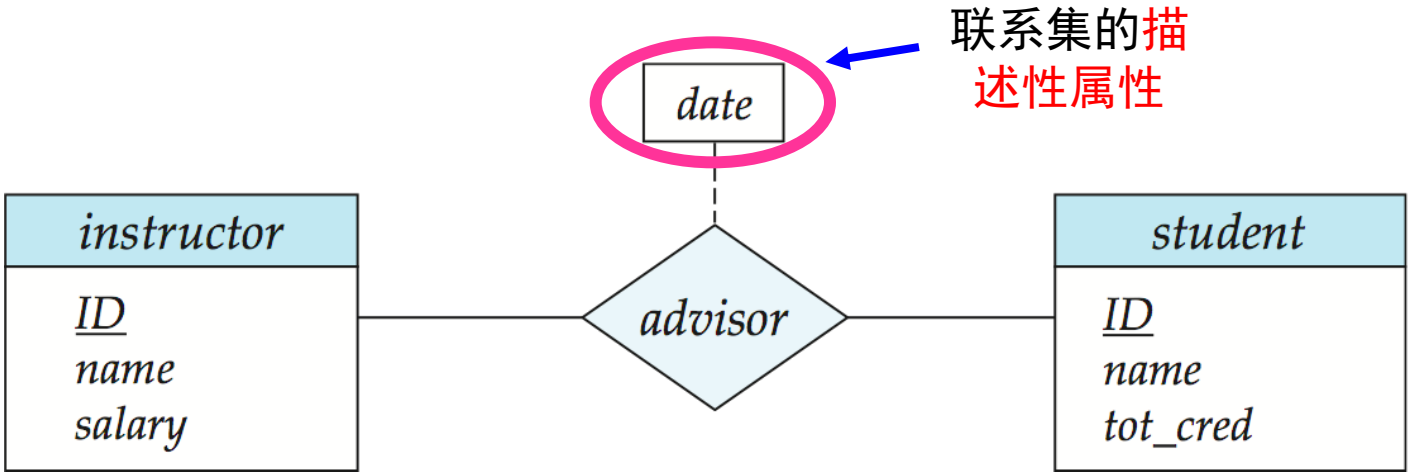
- 虚线将联系集属性连接到联系集
- 双线显示实体在联系集中的参与度
- 双菱形代表连接到弱实体集的标志性联系集

E-R图

- 包含复合、多值和派生属性的E-R图

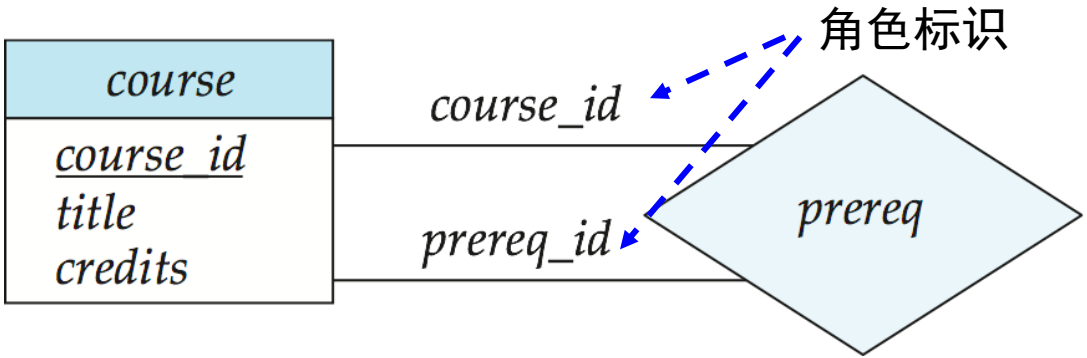


- 带有属性的联系集


联系集的描述性属性
date
instructor
ID
name
salary
advisor
student
ID
name
tot_cred

角色

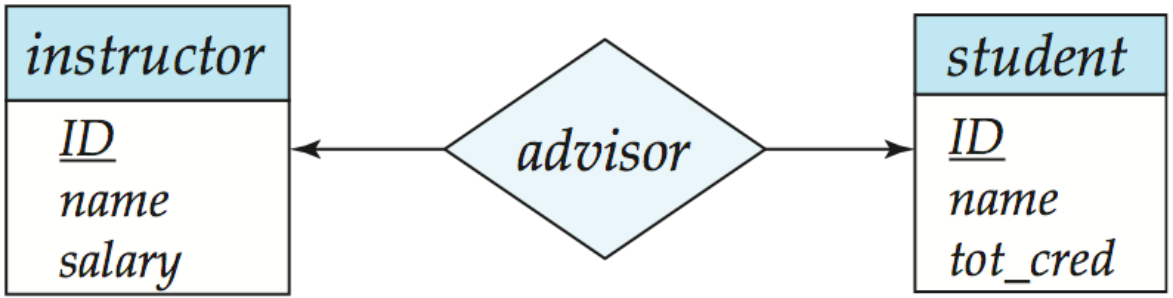
- 参加联系的实体集不必是互不相同的
  - 例，自环联系集（recursive relationship set）
- 角色：实体在联系集中的作用
  - 例，下图给出了*course*实体集和*prereq*联系集之间的角色标识*course_id*和*prereq_id*



- 角色标记是可选的，用于明确联系的语义

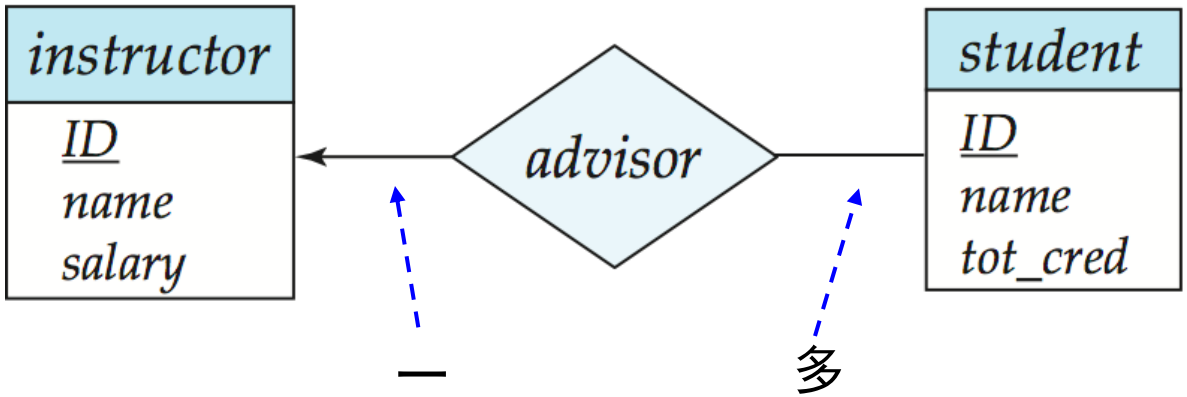
基数约束

- 在联系集与实体集之间用有向直线(→)表示“一”，无向直线(—)表示“多”
- 一对一联系：
  - 实体集instructor和student之间的联系集可以是一对一，表示一名教师可以指导至多一名学生，并且一名学生可以有至多一位导师
  - 注：一个联系集的类型取决于被表达对象的语义约束及设计者的意图



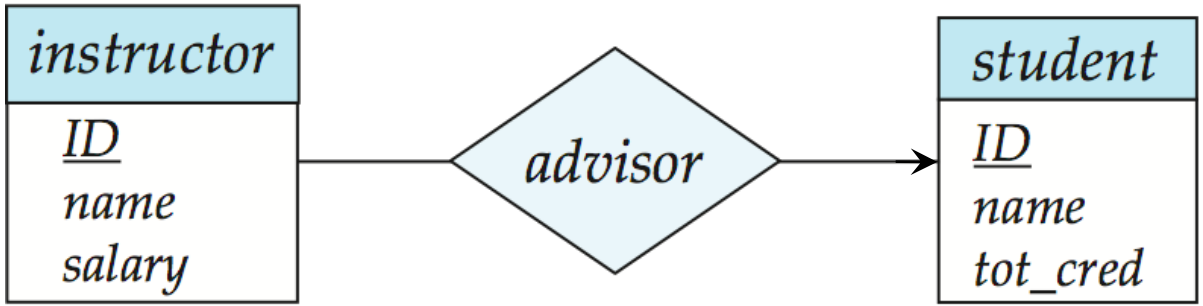
基数约束

- 一对多联系：
  - 实体集instructor和student之间的联系集可以是一对多，表示一名教师可以指导多名学生，但一名学生可以有至多一位导师



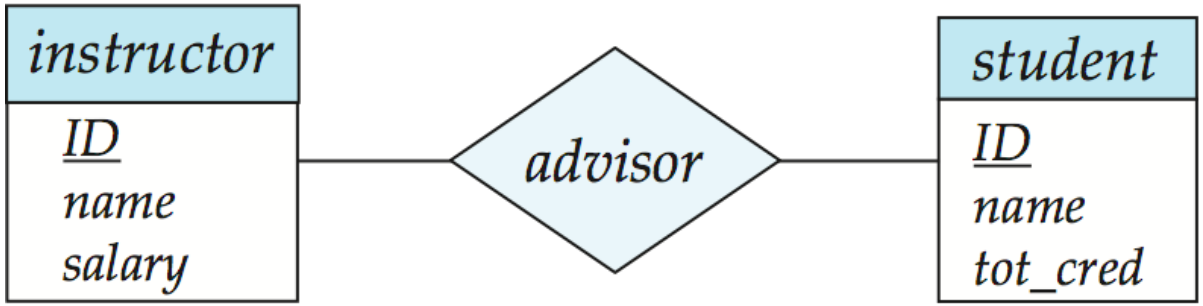
基数约束

- 多对一联系：
  - 实体集instructor和student之间的联系集可以是多对一，表示一名教师可以指导至多一名学生，但一名学生可以有多位导师



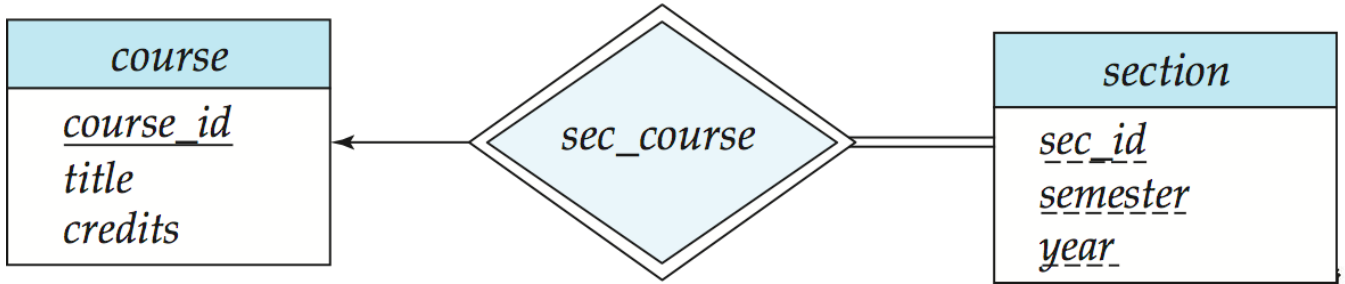
基数约束

- 多对多联系：
  - 实体集instructor和student之间的联系集可以是多对多，表示一名教师可以指导多名学生，并且一名学生可以有多位导师



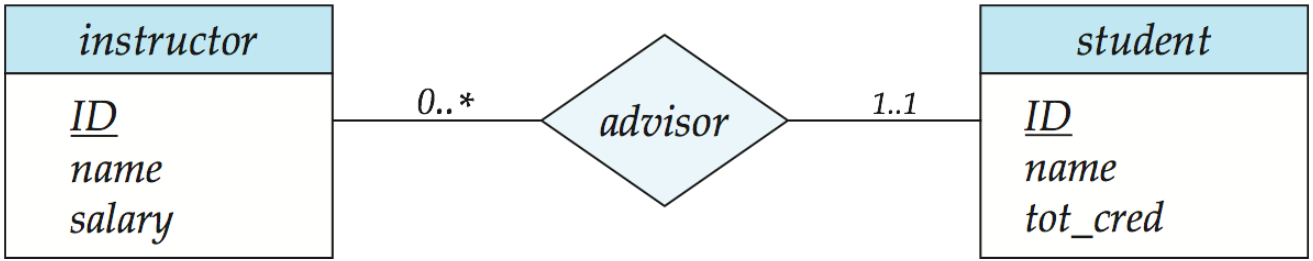
参与约束

- 实体集参加联系集的方式
  - 全参与(用双线表示)：实体集中的每个实体都至少参加联系集中的一个联系
  - 部分参与：某些实体可能未参加联系集中的任何联系
- 映射基数约束(Mapping cardinality constraints)，限定了一个实体与发生关联的另一端实体可能关联的数目上限
- 全参与和部分参与约束，则反映了一个实体参与关联的数目下限：0次，还是至少1次



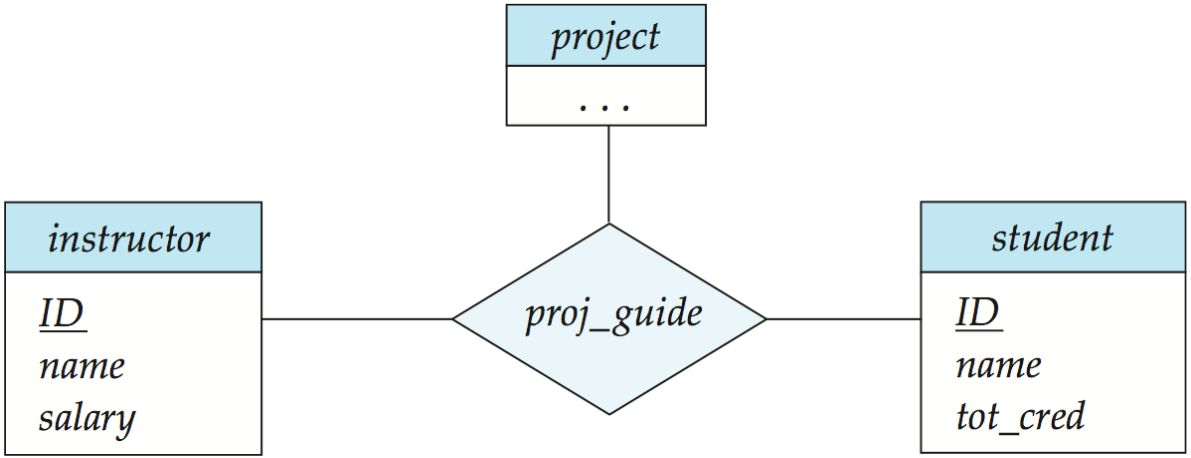
关系约束的另一种表示法

- 用关系约束的另一种表示法来标识基数约束和参与约束
  - 例，每个学生有且仅有一个导师，教师可以有零个或多个学生



具有三元联系的E-R图

- 例，一个教师可以指导多个学生，并可以参与到多个项目



二元与非二元联系

- 某些看起来似乎是非二元的联系可用二元联系更好地表示
  - 例，三元联系 parents将孩子与其父亲和母亲相关联，可以更好地用两个二元联系father和mother代替

parents(he, she, child) => father (he, child)
mother(she, child)

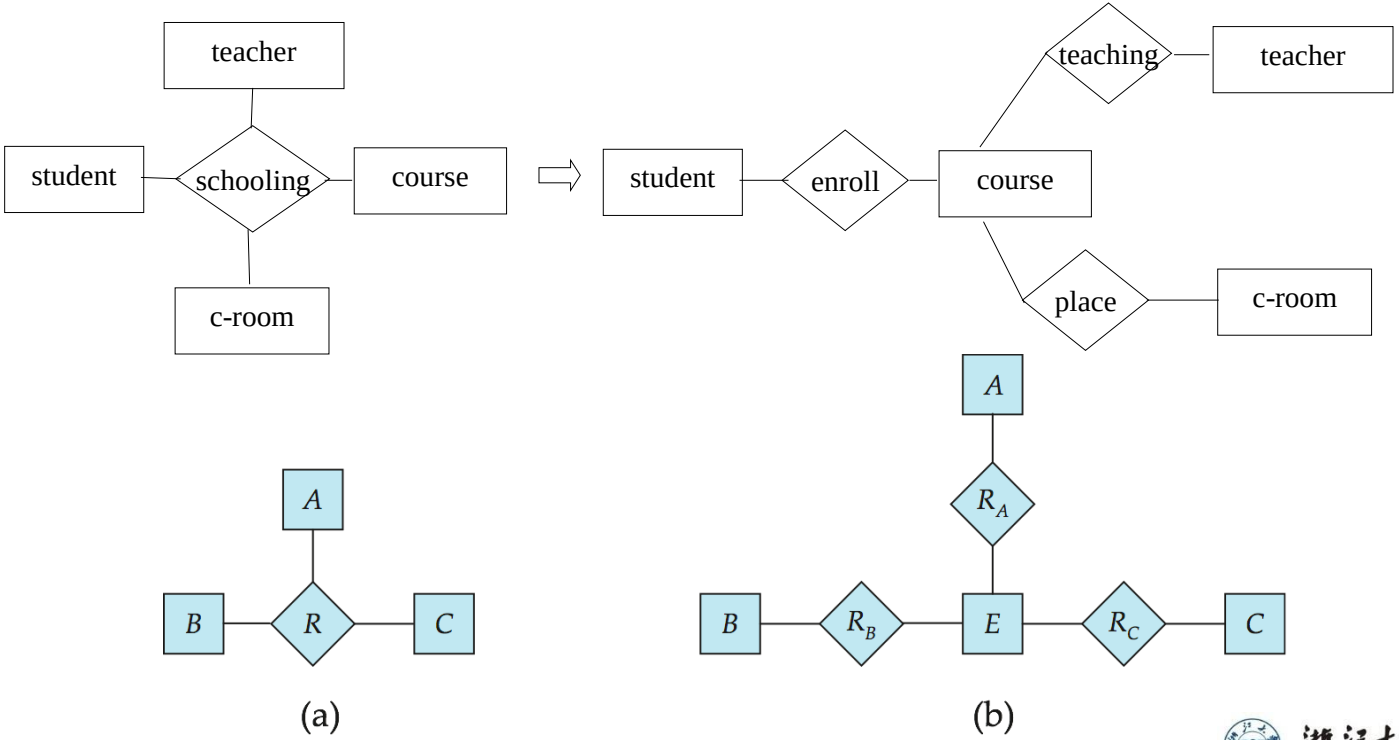
  - 使用二元联系可以表达部分信息（如，只知道母亲）
- 但有些联系用非二元更自然
  - 例，*proj_guide*(*instructor*, *project*, *student*)

非二元联系转换成二元联系

- 任何非二元联系都可以用二元联系表示，方法是人为创建一个实体集
  - 将实体集 A，B，C 之间的联系R 用实体集E 和以下三个联系集代替：
    - —R_A，将 E 与 A 关联
    - —R_B，将 E 与 B 关联
    - —R_C，将 E 与 C 关联
  - 为E 创建一个特殊的标识属性
  - 将R 的所有属性加给E
  - 对R 中每一个联系（a_i，b_i，c_i）
    - —创建实体集E 中的一个新实体e_i
    - —将（e_i，a_i）加入 R_A
    - —将（e_i，b_i）加入 R_B
    - —将（e_i，c_i）加入 R_C

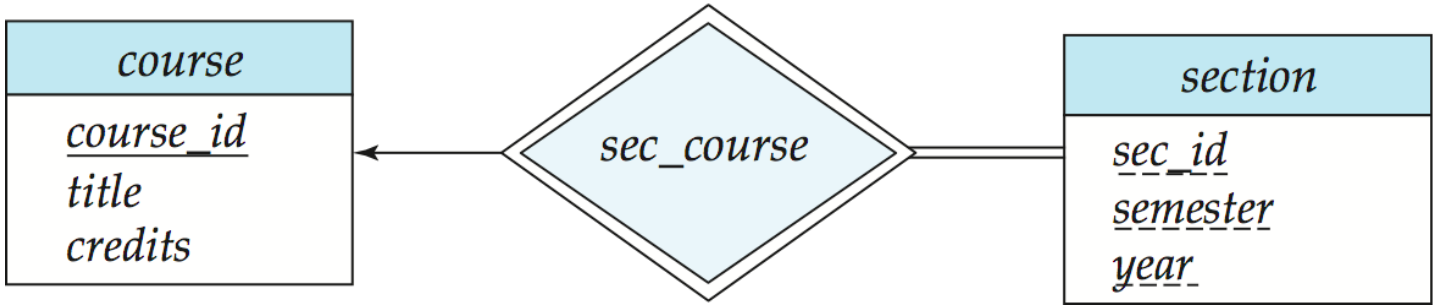
非二元联系转换成二元联系

- 例，将如下非二元联系*schooling* 转换成二元联系



弱实体集

- 不具有主键的实体集称为弱实体集
- 例，考虑一个*section*实体，它由课程编号、学期、学年以及开课编号唯一标识。显然，开课实体和课程实体相关联。假定我们在实体集*section*和*course*之间创建了一个联系集*sec_course*。若，实体集section为：*section*(*sec_id*, *semester*, *year*)，则其为弱实体集



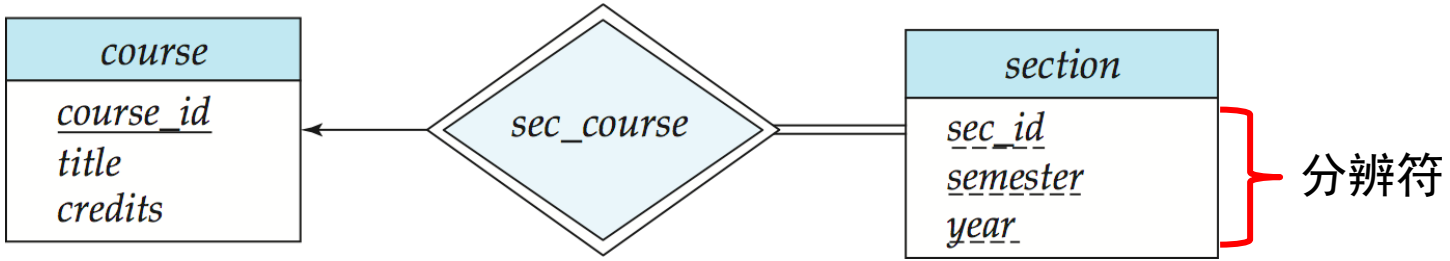
弱实体集

- 弱实体集的存在依赖于它的标识实体集（或属主实体集）的存在
  - 例，*course*(*course_id*, *title*, *credits*)
- 标识性联系：将弱实体集与其标识实体集相联的联系
  - 标识性联系是从弱实体集到标识实体集多对一的，并且弱实体集在联系中的参与是全部的
  - 例，*sec_course*

弱实体集

- 弱实体集的分辨符（或称部分码）是指在一个弱实体集内区分所有实体的属性集合
  - 例，弱实体集*section*的分辨符由属性*sec_id*，*year*以及*semester*组成
- 弱实体集的主码由它所依赖的强实体集的主码加上它的分辨符组成
  - section的主码：

{*course_id*, *sec_id*, *semester*, *year*}



弱实体集

- 注意：强实体集的主码并不显式地存于弱实体集中，而是隐含地通过标识性联系起作用
- 如果*course_id*显式存在，*section*就成了强实体，则*section*与*course*之间的联系变得冗余。因为*section*与*course*共有的属性*course_id* 已定义了一个隐含的联系