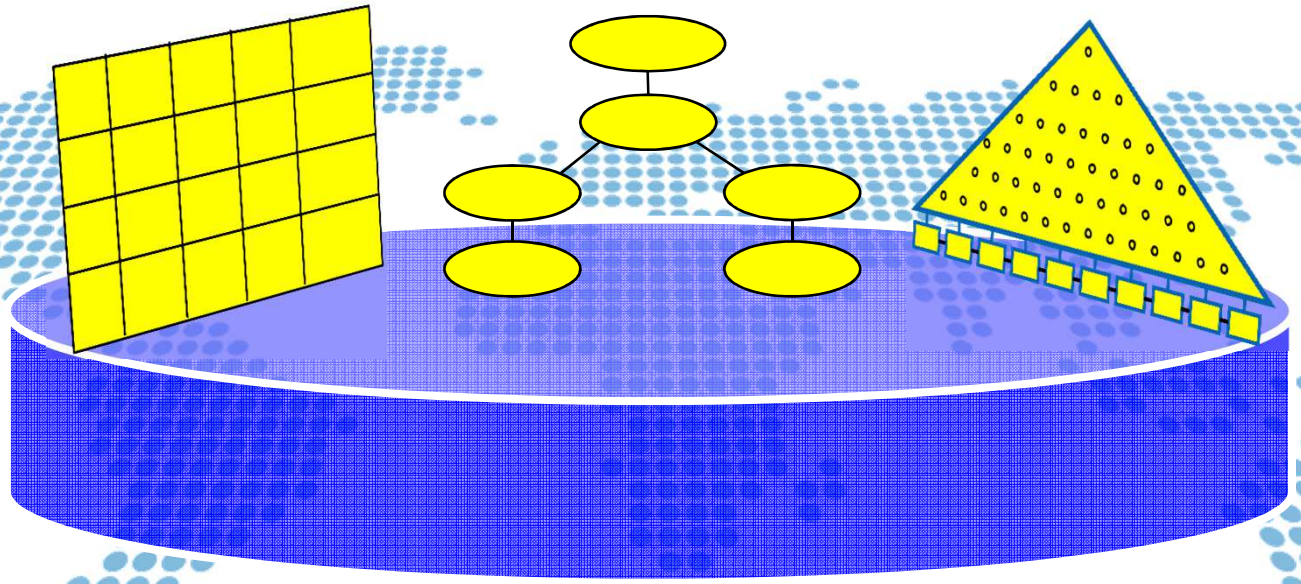
数据库系统

ER模型

陈世敏

（中科院计算所）



Outline

- 数据库设计过程
- ER模型
- 应用举例
- UML模型介绍

数据库设计过程

1) 需求分析
2) 概念设计（ER模型等）
3) 逻辑设计（ER模型➔关系模型）
4) 模式细化（归一化等）
5) 物理设计（物理模式，性能等）
6) 应用与安全设计（定义外部模式等）

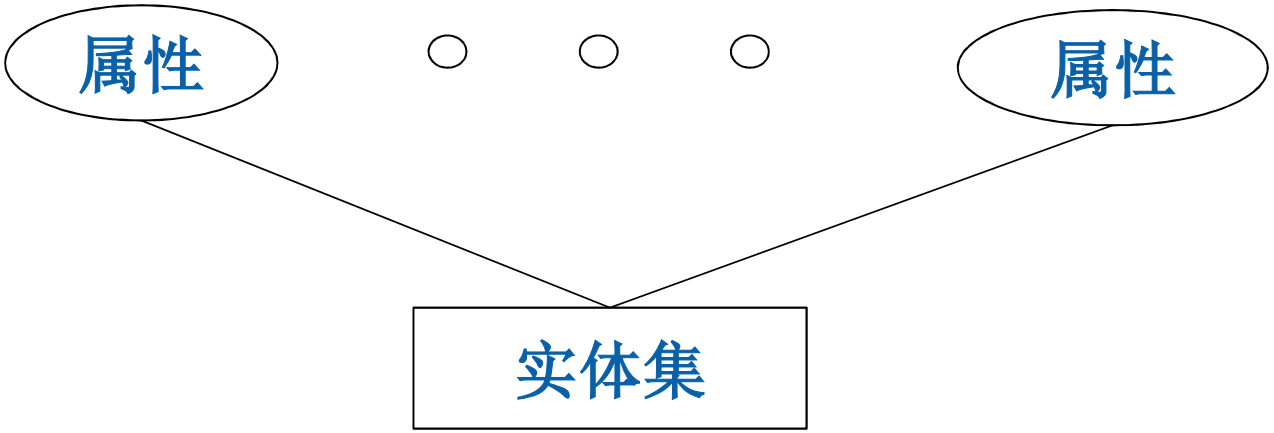
实体、实体集、属性

- 实体（Entity）
  - 现实世界的对象，可以与其它对象区分
  - 例如：一个学生
- 实体集（Entity Set）
  - 相似实体形成的集合
    - 例如：学生的集合，运动员的集合，课程的集合，教师的集合
  - 注意这些实体集之间可以有交集
- 实体的属性（Attribute）
  - 一个实体集的每个实体都有相同的属性
    - 例如：学生可以有姓名、学号、出生日期、专业等
  - 属性的取值范围是一个值域（Domain）
  - 属性的详细程度取决于应用的需要

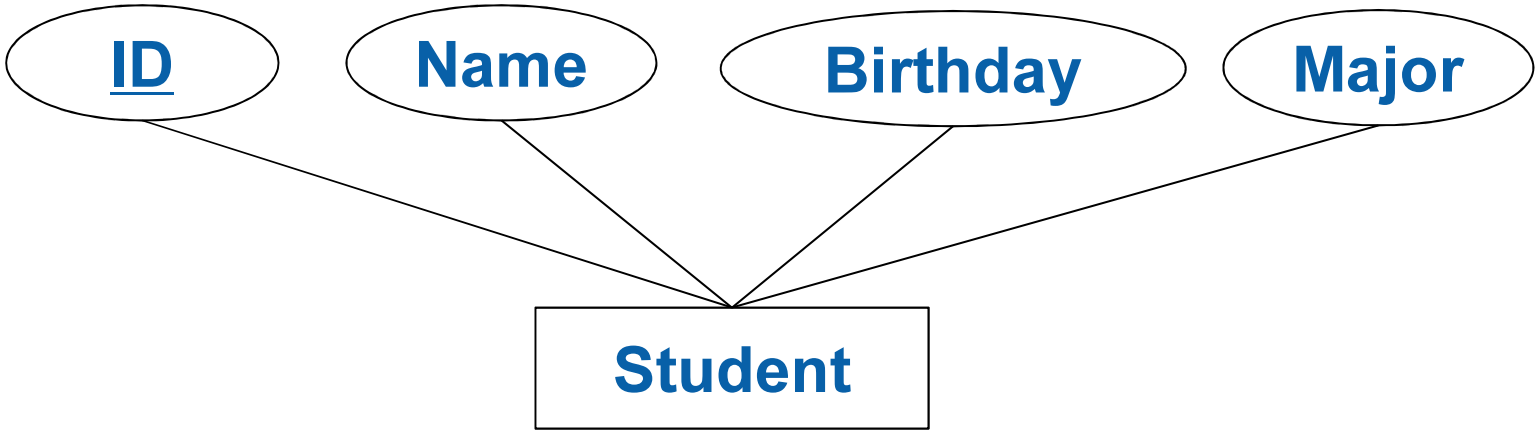
键（Key）

- 键
  - 可以唯一确定一个实体的属性集
    - 两个不同的实体，键不同
  - 是最小的属性集，即其任何子集都不是键
- 候选键（Candidate key）
  - 可能存在多个候选键
- 主键（primary key）
  - 选定一个候选键为主键

实体集和属性的表示



实体集和属性的表示

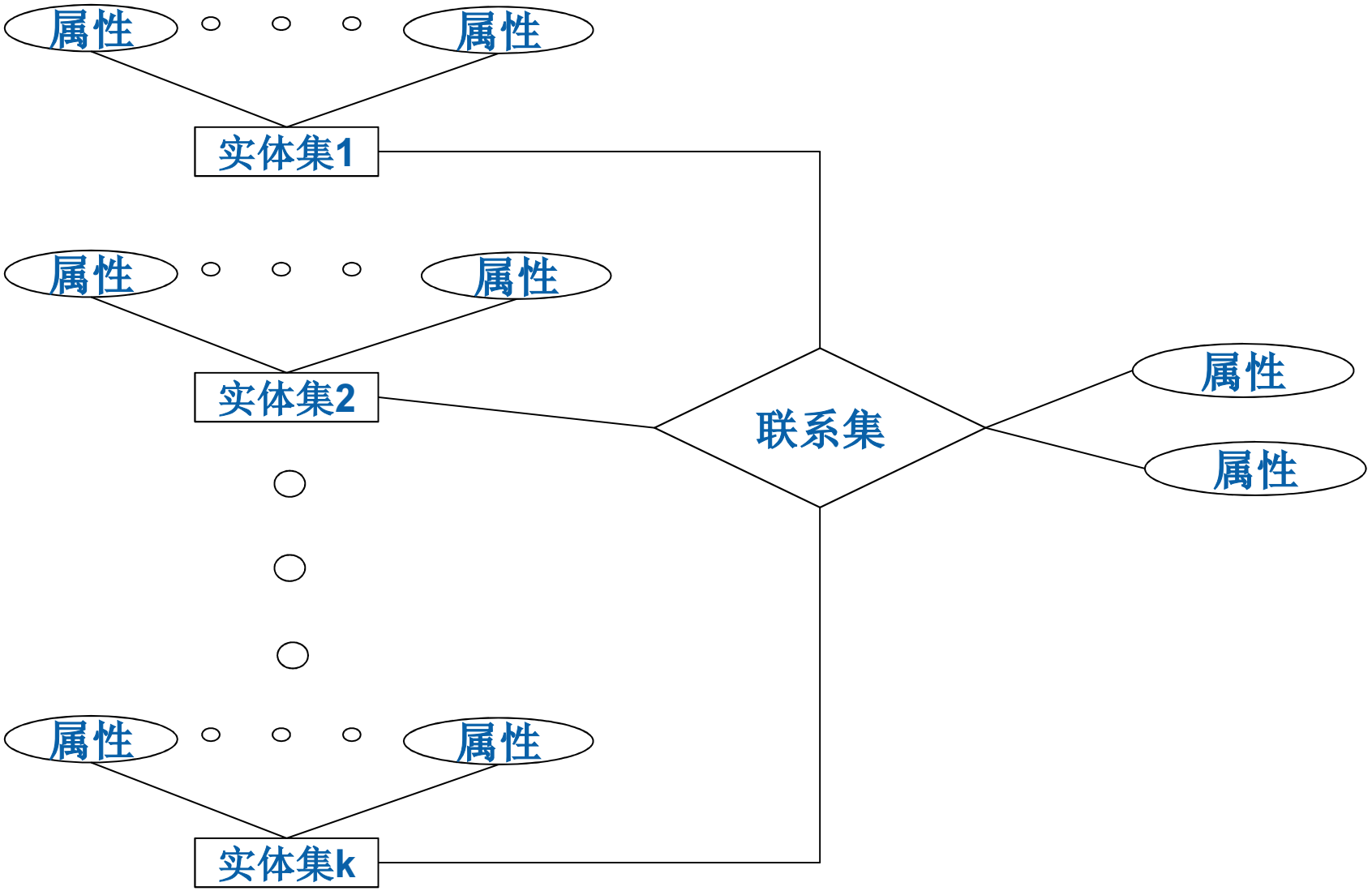


ID是主键，对主键所包含的属性加下划线

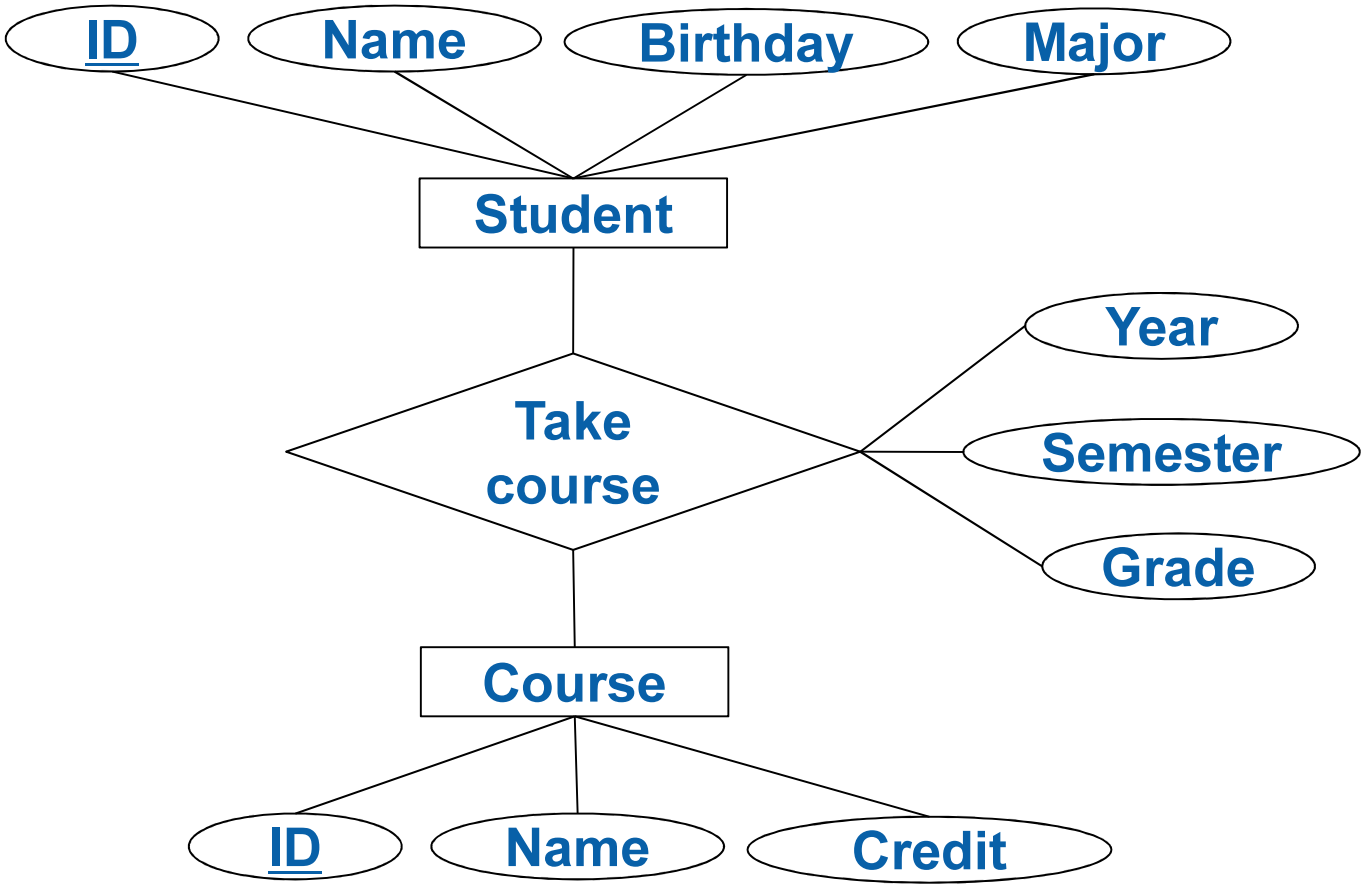
联系、联系集、属性

- 联系（Relationship）
  - 两个或多个实体之间的关联
- 联系集（Relationship Set）
  - 类似联系的集合
  - $\{(e_1,\ldots,e_n) | e_1 \in E_1, \ldots, e_n \in E_n\}$ 其中每个n元组表示这n个实体之间的一个联系
  - 上述是一个n-元联系
  - 常见的联系有二元联系、三元联系等
- 联系的属性
  - 用于描述联系，而不是描述任何一个单一实体

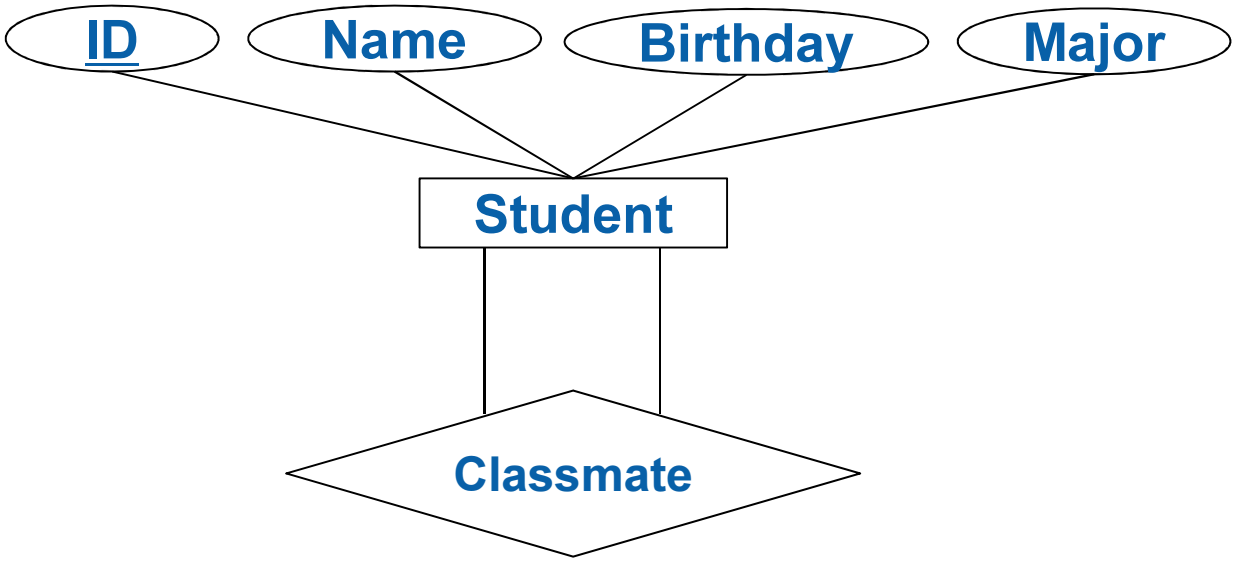
实体集与联系集



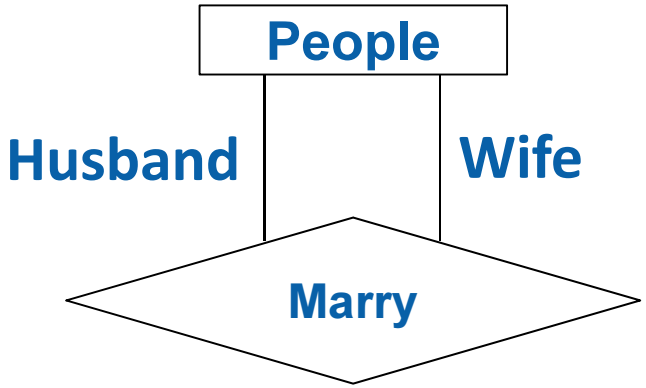
实体集与联系集


ID
Name
Birthday
Major
Student
Year
Take course
Semester
Grade
Course
ID
Name
Credit

单一实体集内部的联系



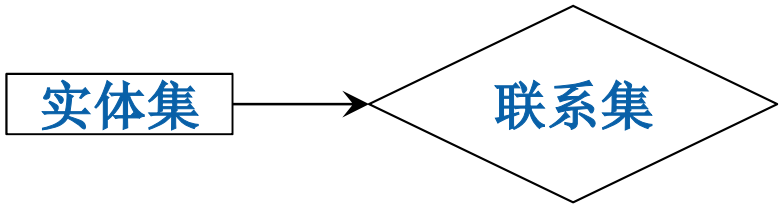
单一实体集内部的联系



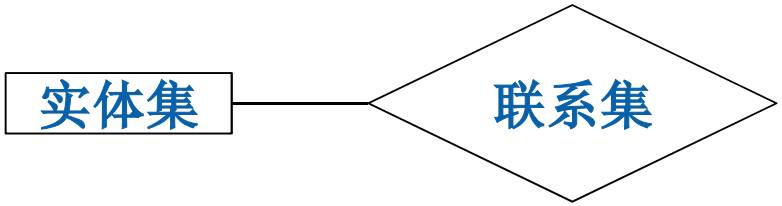
如果联系是非对称的，可以在边上标注联系中的角色

多对多和1对多联系

- 考虑一个实体集与一个联系集
- 每个实体可能存在于多少个联系中呢？
  - 每个实体只可以有唯一的联系，**加箭头表示**，也被称作键约束（key constraint）

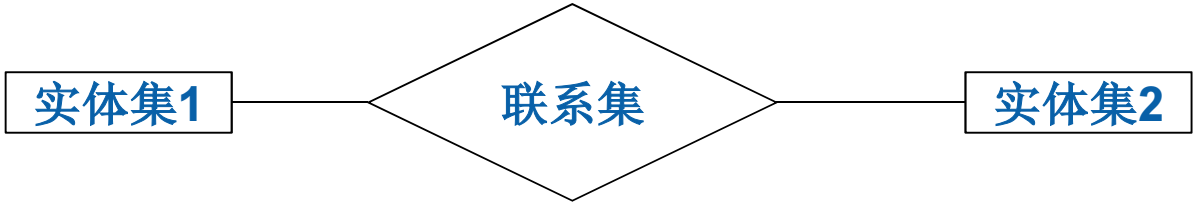


  - 每个实体可能参与多个联系，那么**没有箭头**

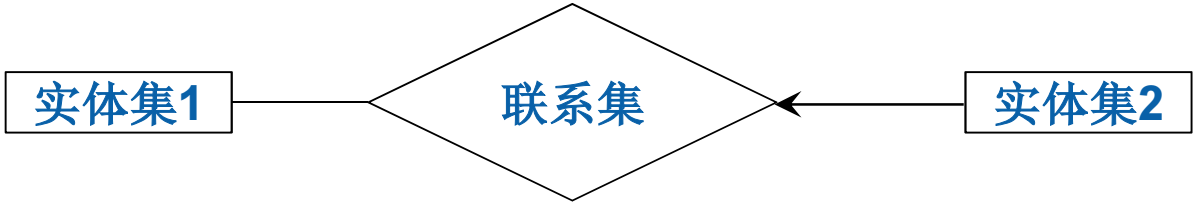


多对多和1对多联系

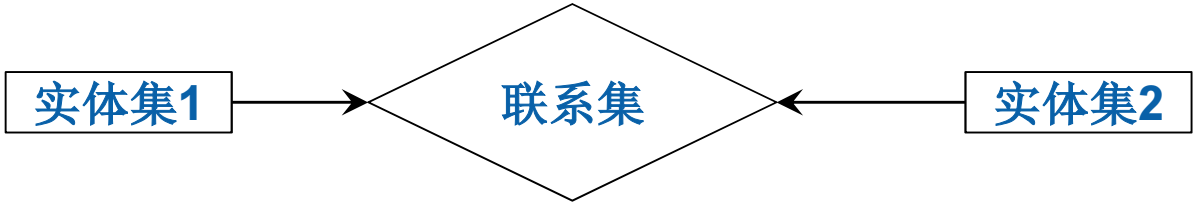
- 多对多（Many-to-many）



- 1对多（One-to-many）

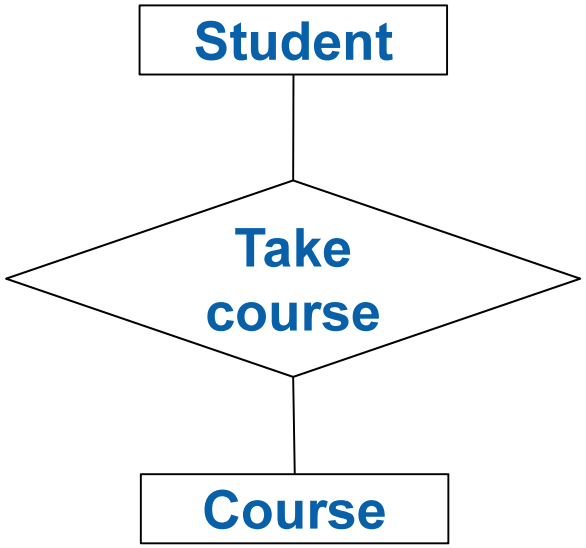


- 1对1（One-to-one）



多对多联系(Many-to-Many)

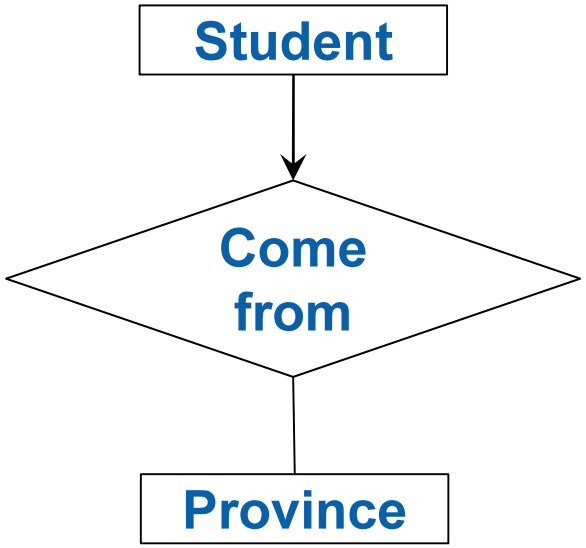
单边表示
多对多



- 每个Student可以选多个Course
- 每个Course可以有多个Student选

1对多联系（One-to-Many）

单边加箭头
代表 1对多

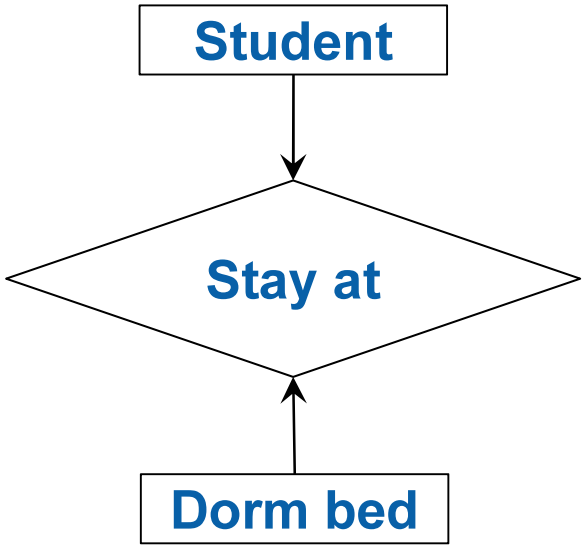


每个学生来自
的省是唯一的

- 每个Student只可能来自一个Province
- 每个Province可以有多个Student

1对1联系（One-to-One）

单边加箭头
代表 1对多

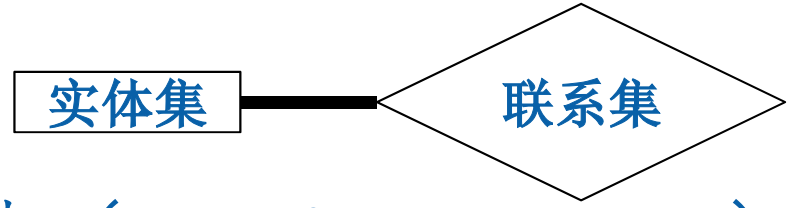


每个学生与宿舍床位是1对1的联系

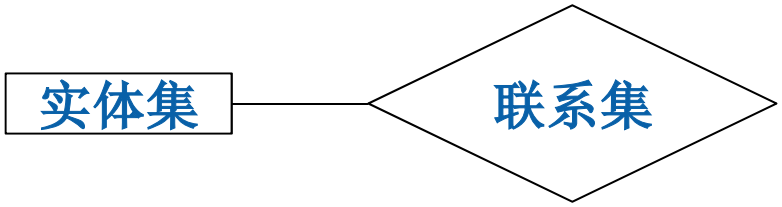
- 每个Student只可以被分配一个Dorm bed
- 每个Dorm Bed只可以分配给一个Student

参与约束（Participation Constraints）

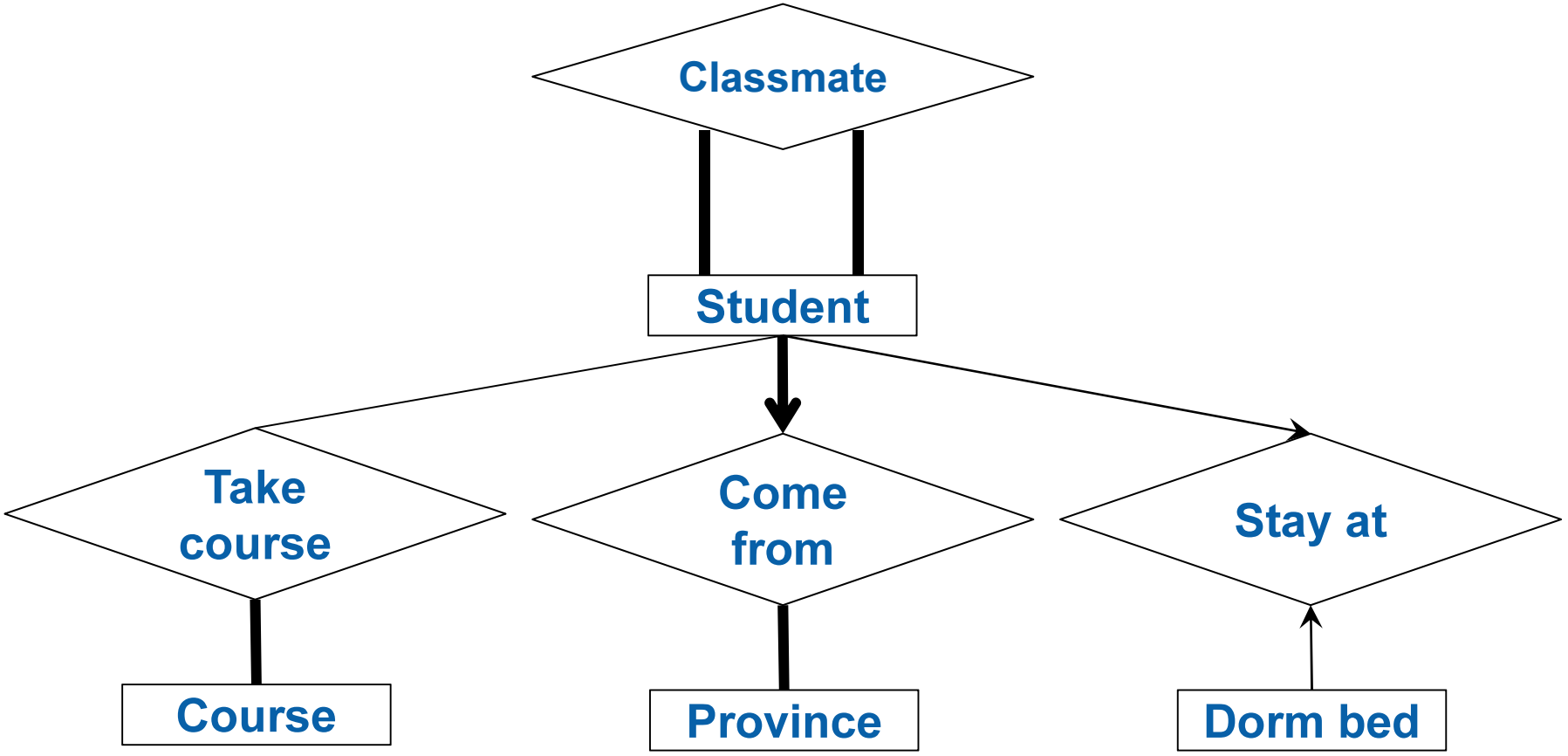
- 考虑一个实体集与一个联系集
- 每个实体是否都参与到某个联系中呢？
- 完全参与（Total participation）
  - 每个实体都参与到某个联系中，用粗线表示



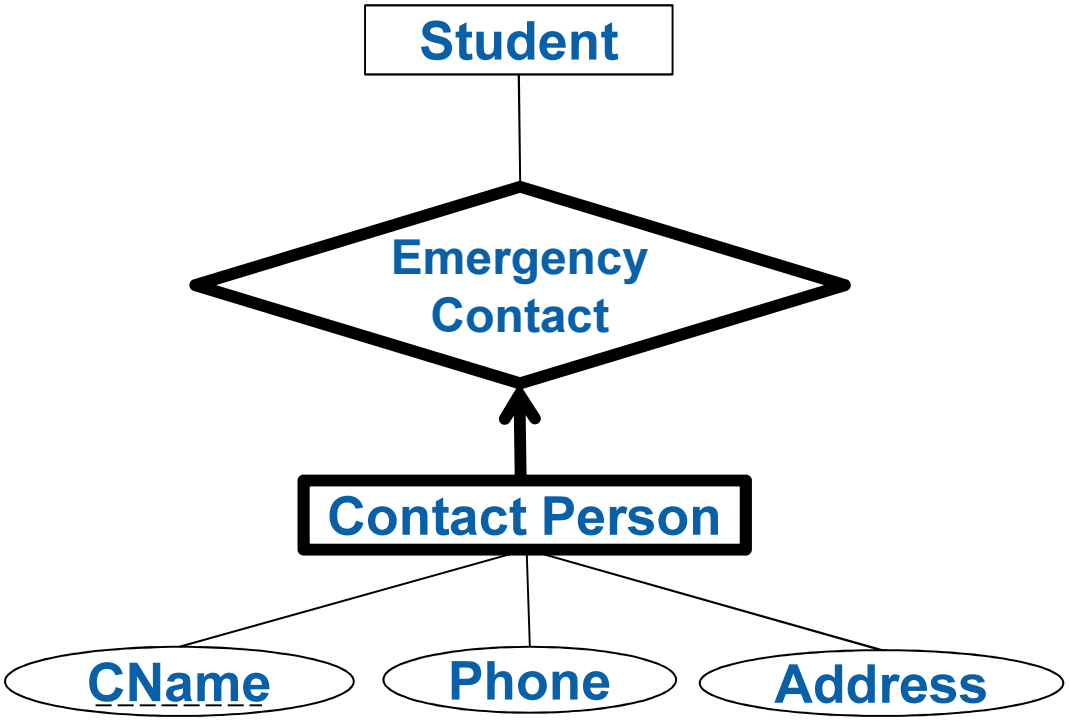
- 部分参与（Partial participation）
  - 有的实体没有参与到联系中，用细线表示



参与约束（Participation Constraints）

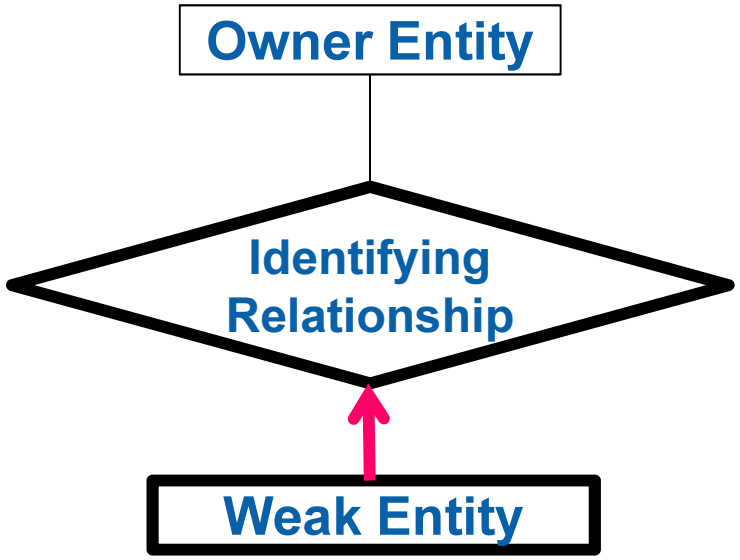


弱实体（Weak Entities）



- 这里联系人是从属于学生的，学校记录联系人的信息只是因为学生的原因
- 联系人是Weak Entity

弱实体（Weak Entities）的特征

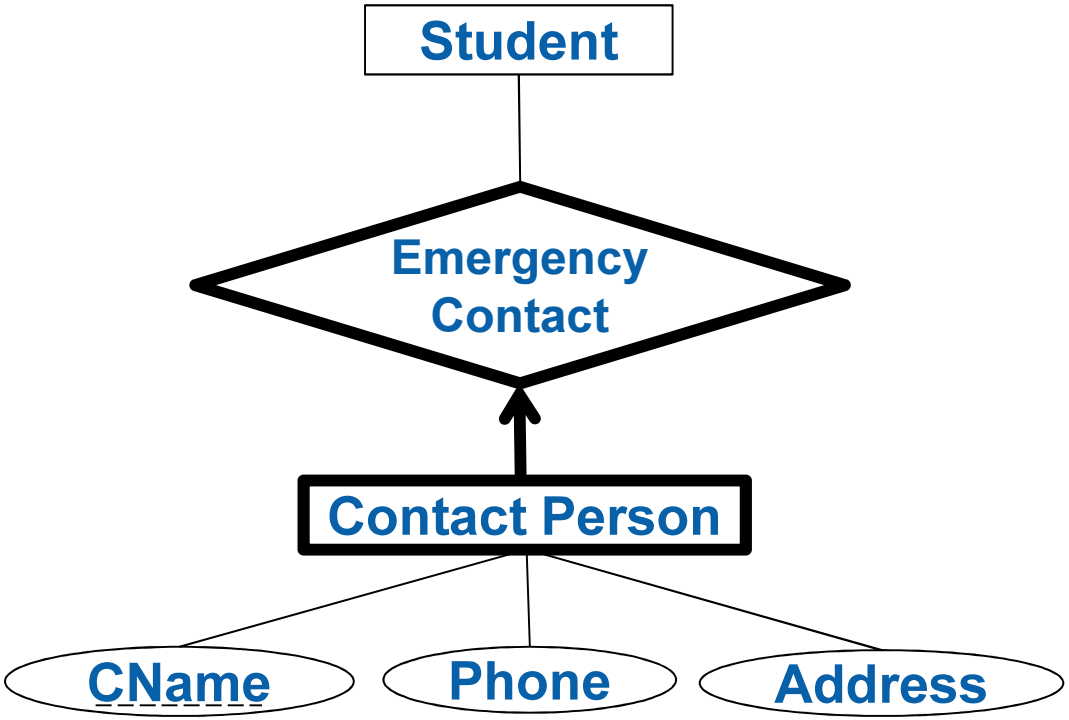


粗线表示

粗线表示

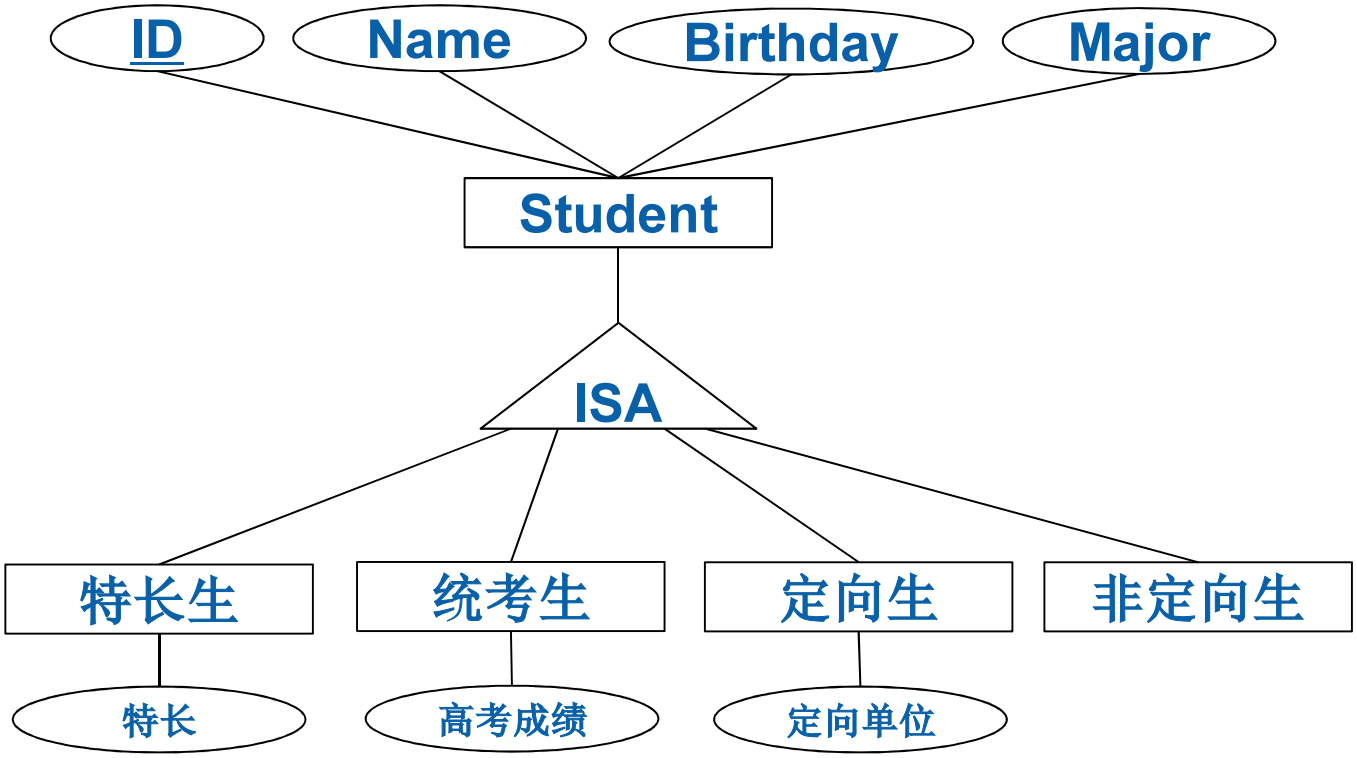
- 每个Weak entity必须有且仅有唯一的owner entity，粗线+箭头
- 反过来，对owner entity没有限制

部分键（Partial Key）



- 对同一个学生的所有联系人，Cname可以唯一确定其联系人
- 但是，不同学生的联系人的姓名可能碰巧相同
- 这种键叫部分键（partial key），用虚线下划线表示

类层次



- 目的
  - 某些属性只对部分实体有意义；或者某种联系只与部分实体相关
- 用ISA表示实体集之间的继承关系
- 子实体集的属性包含父实体集的属性+增加的属性

类层次的约束

- 交迭约束（Overlap）
  - 两个子类是否可以有交集？存在共同的实体？
  - 默认：不相交
  - 如有，需要特殊说明
    - 例如：统考生和定向生可以相交

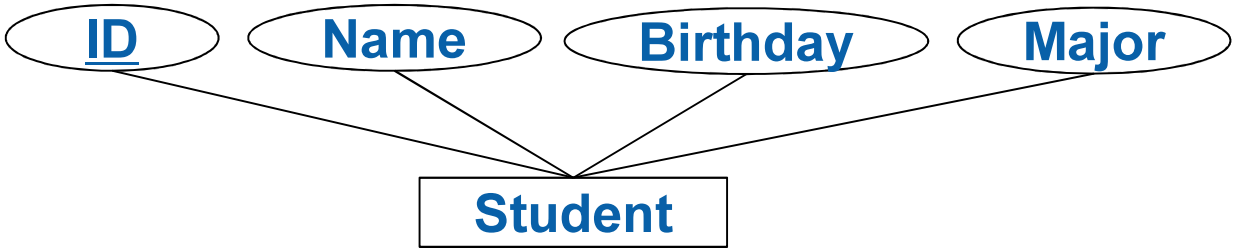
- 覆盖约束（Covering）
  - 多个子类的并集是否覆盖了父实体集？
  - 默认：没有覆盖约束
  - 如有，需要特殊说明
    - 例如：定向生和非定向生覆盖了所有学生

建模的设计原则

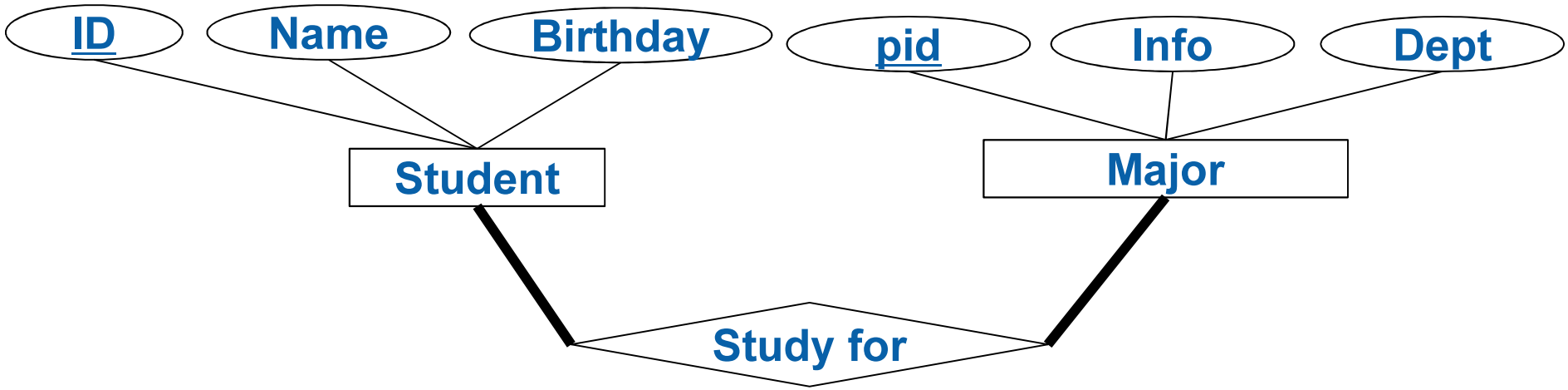
- 与现实相符
- 在需求的基础上，尽可能简单
- 避免一个信息在多个地方冗余表示
  - 例如，Take-course包括学生姓名
- 选择恰当的方法来表示
  - 属性？
  - 实体？
  - 联系？

实体 vs. 属性

- Major是一个属性



- Major是一个实体



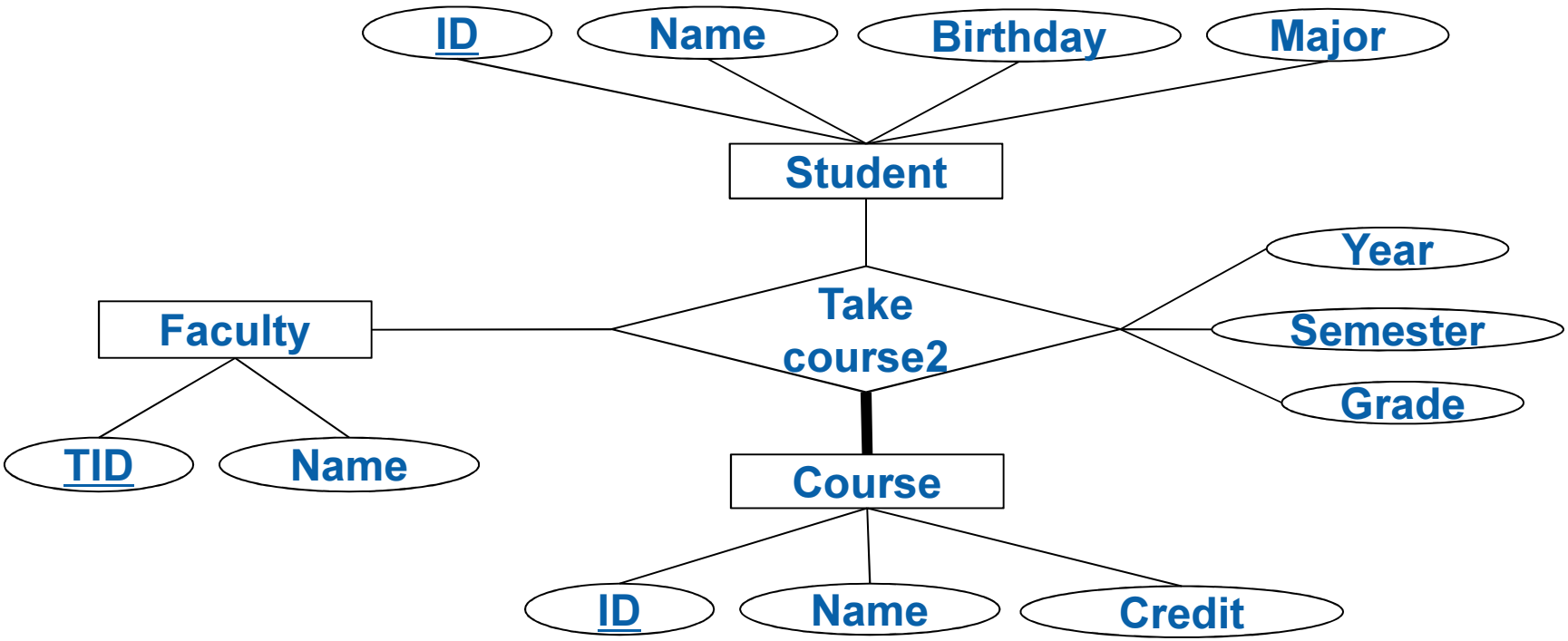
- 每个学生现在可以有一个或多个专业
- 专业可以有进一步详细的属性

实体 vs. 属性

- 考虑两方面
  - 只有1个，还是可以有多个
  - 是简单的值，还是有丰富的内部结构

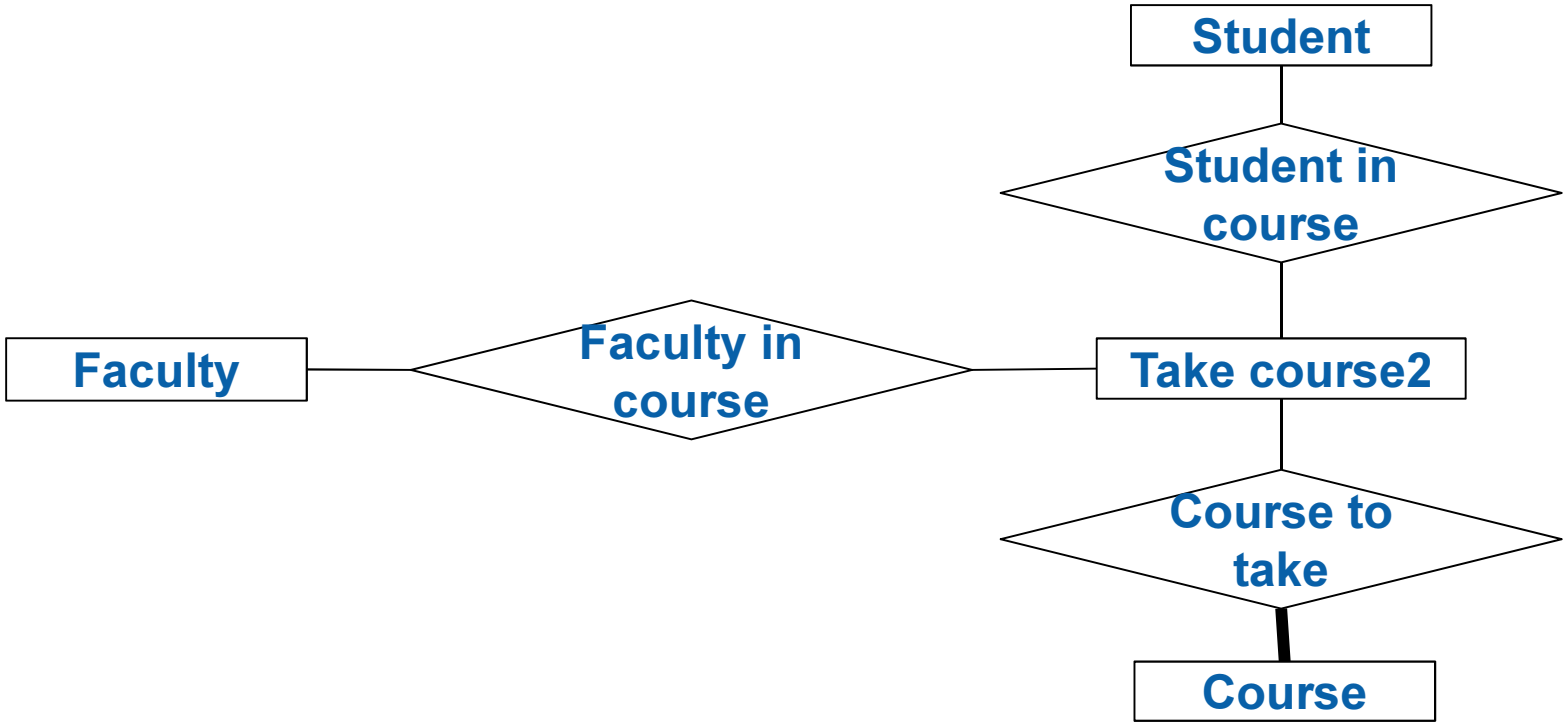
- 属性
  - 只有1个，而且是简单的值
- 否则，用实体表示

多元联系可以用实体表示



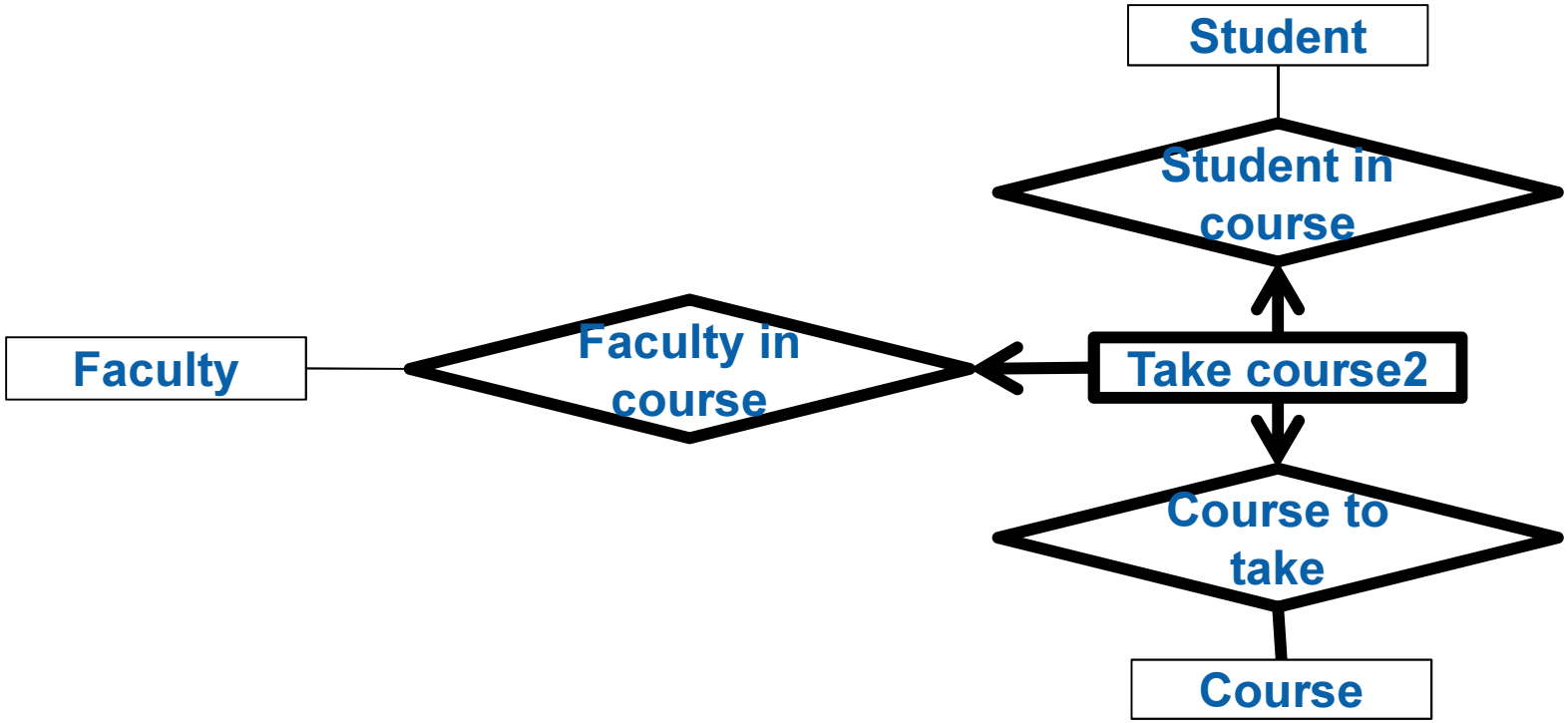
- 这是一个多元关系

多元联系可以用实体表示



- 转化为一个实体和多个二元关系

多元联系可以用实体表示



- 转化为一个实体和多个二元关系
- 用于连接的实体集是一个弱实体集

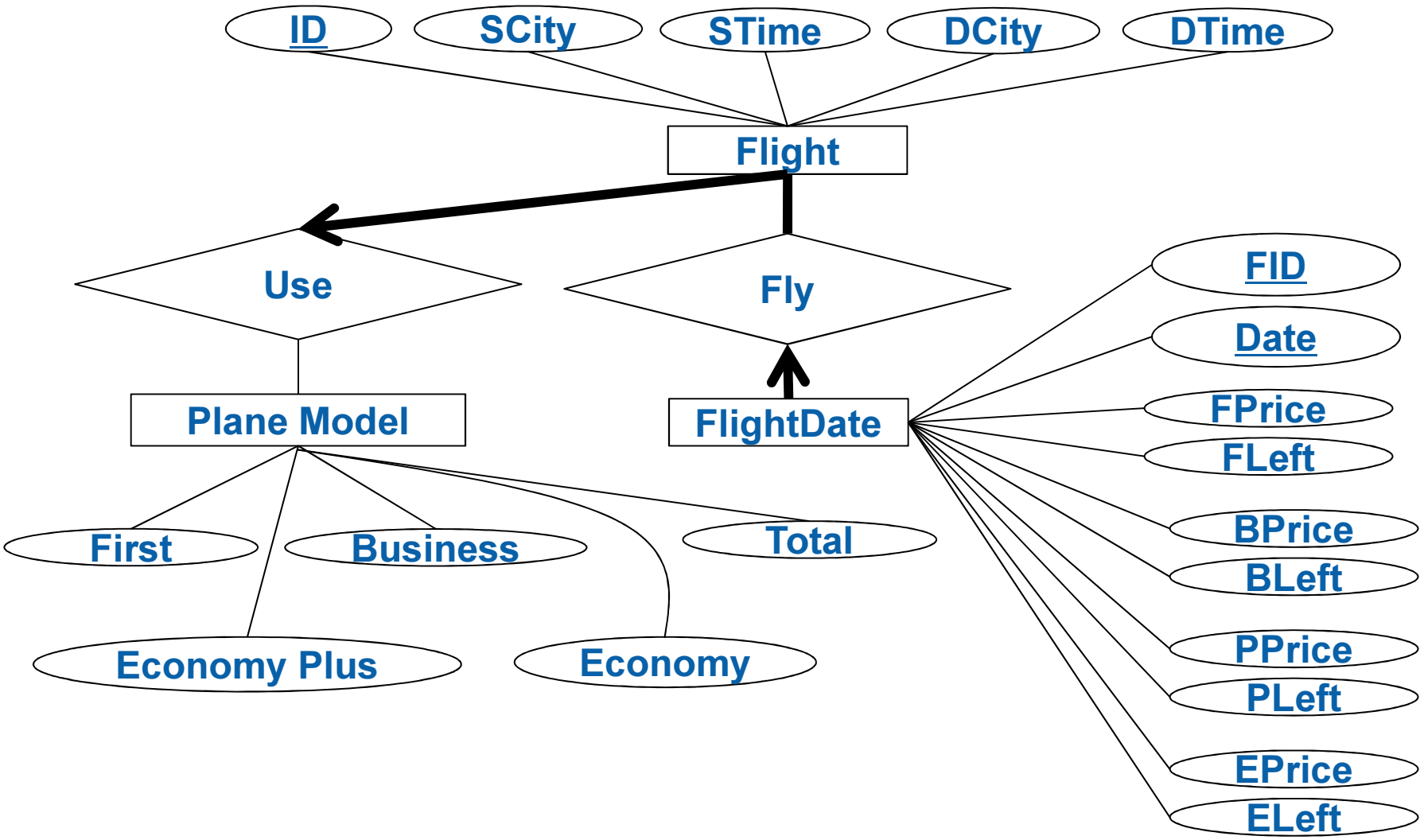
Outline

- 数据库设计过程
- ER模型
- 应用举例
- UML模型介绍

飞机订票网站

- 假设设计一个类似airchina.com.cn的飞机订票网站
- 需求分析(1)
  - 每个航班都有自己的一系列的运营日期
  - 每个航班的信息包括起飞城市/机场，到达城市/机场，计划起飞时间，计划到达时间，机型
  - 每种机型都有头等舱，公务舱，优先经济舱，经济舱
  - 不同航班，不同舱位，不同日期，座位有不同的售价

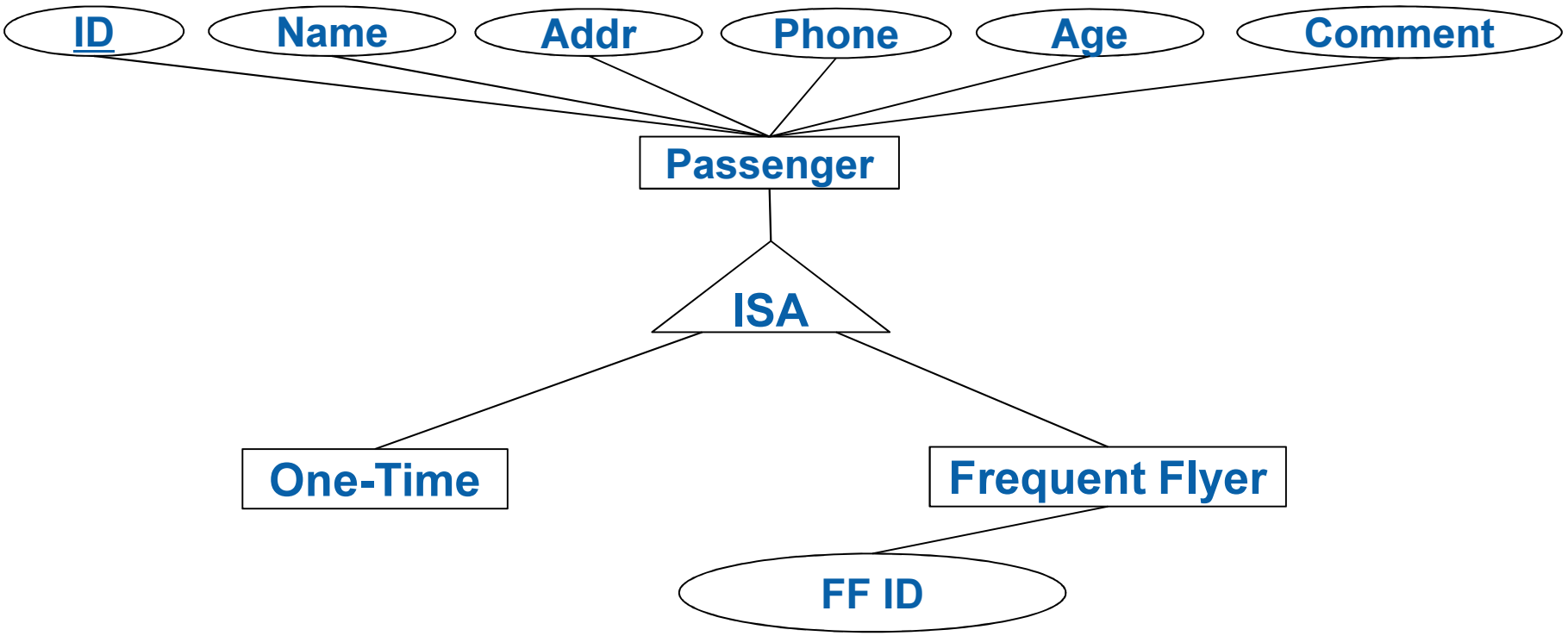
航班



需求分析（2）

- 乘客信息包括身份证号，姓名，年龄，住址，电话，以及备注信息（例如，对于座位的要求、对于饮食的特殊要求等）
- 乘客可以是临时乘客，或常旅客
- 常旅客有常旅卡号

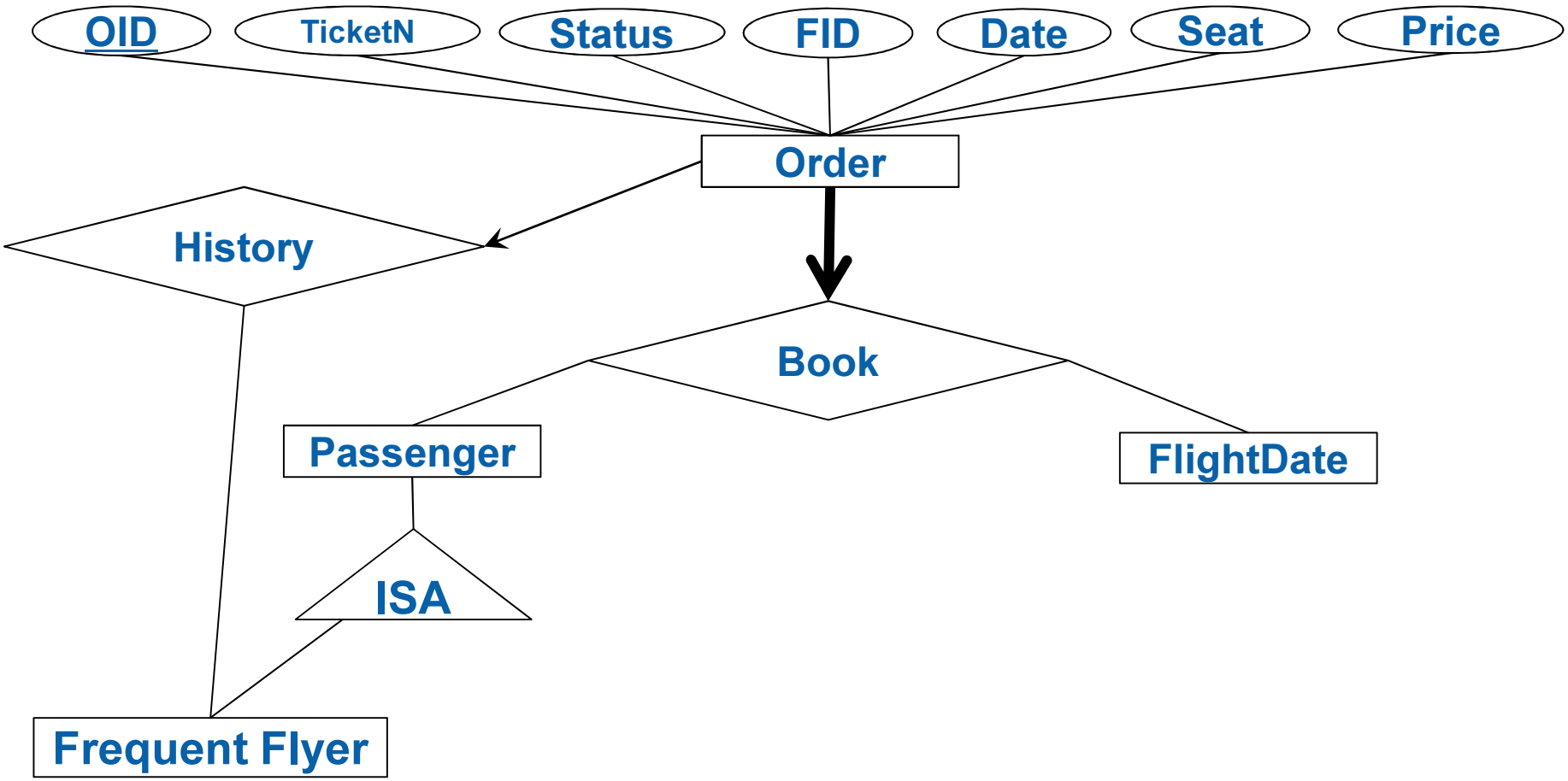
乘客


ID
Name
Addr
Phone
Age
Comment
Passenger
ISA
One-Time
Frequent Flyer
FF ID

需求分析（3）

- 乘客可以下订单购买机票
- 每个订单包含订单号，机票号，付款状态，航班和日期，但是可确定或未确定具体座位，票价
- 乘客可以根据订单号或机票号，查询已有订单，修改已有订单，取消已有订单
- 对于常旅客，可以查询历史订单信息

订单


OID
TicketN
Status
FID
Date
Seat
Price
Order
History
Book
Passenger
FlightDate
ISA
Frequent Flyer

更多应用？

- 火车订票网站（类似12306）？
  - 不太一样，火车是有多个站的
  - 怎么处理呢？
- 购物网站？
  - 商品不是一种了
  - 也有顾客

Outline

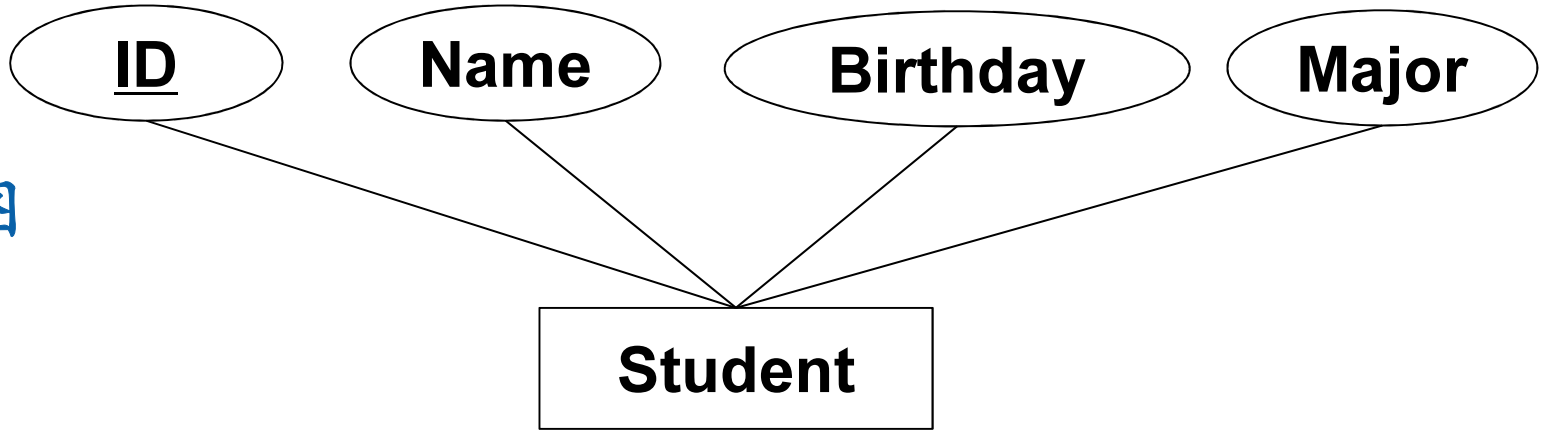
- 数据库设计过程
- ER模型
- 应用举例
- UML模型介绍

UML（Unified Modeling Language）

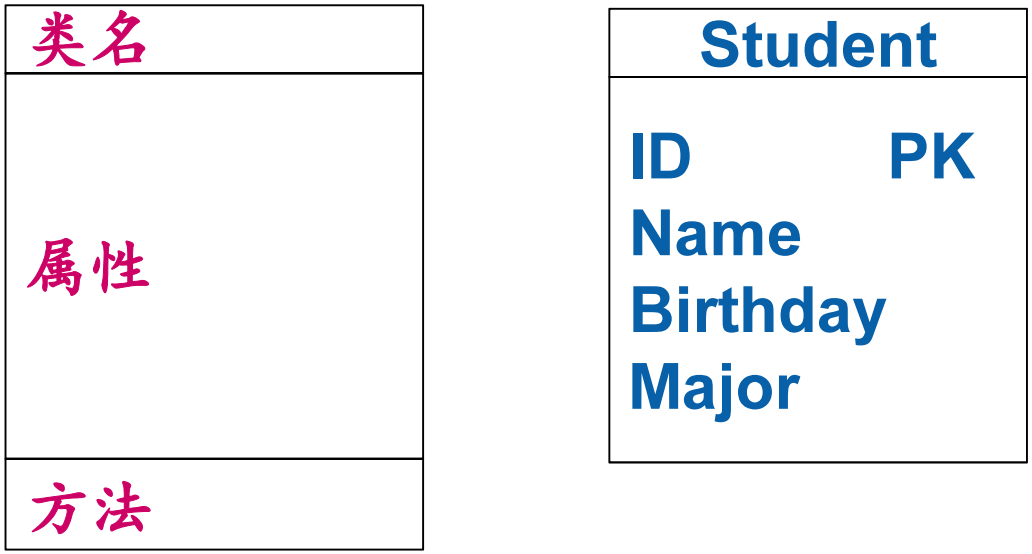
- UML包含了更广泛的软件设计过程
- 其中，UML类图(Class Diagram)可用于对数据库进行概念设计
- ER图与UML的关系
  - ER图的实体——UML的类
  - ER图的联系——UML的类之间的联系和联系类
  - UML可以方便地支持子类继承关系

实体和属性的表示

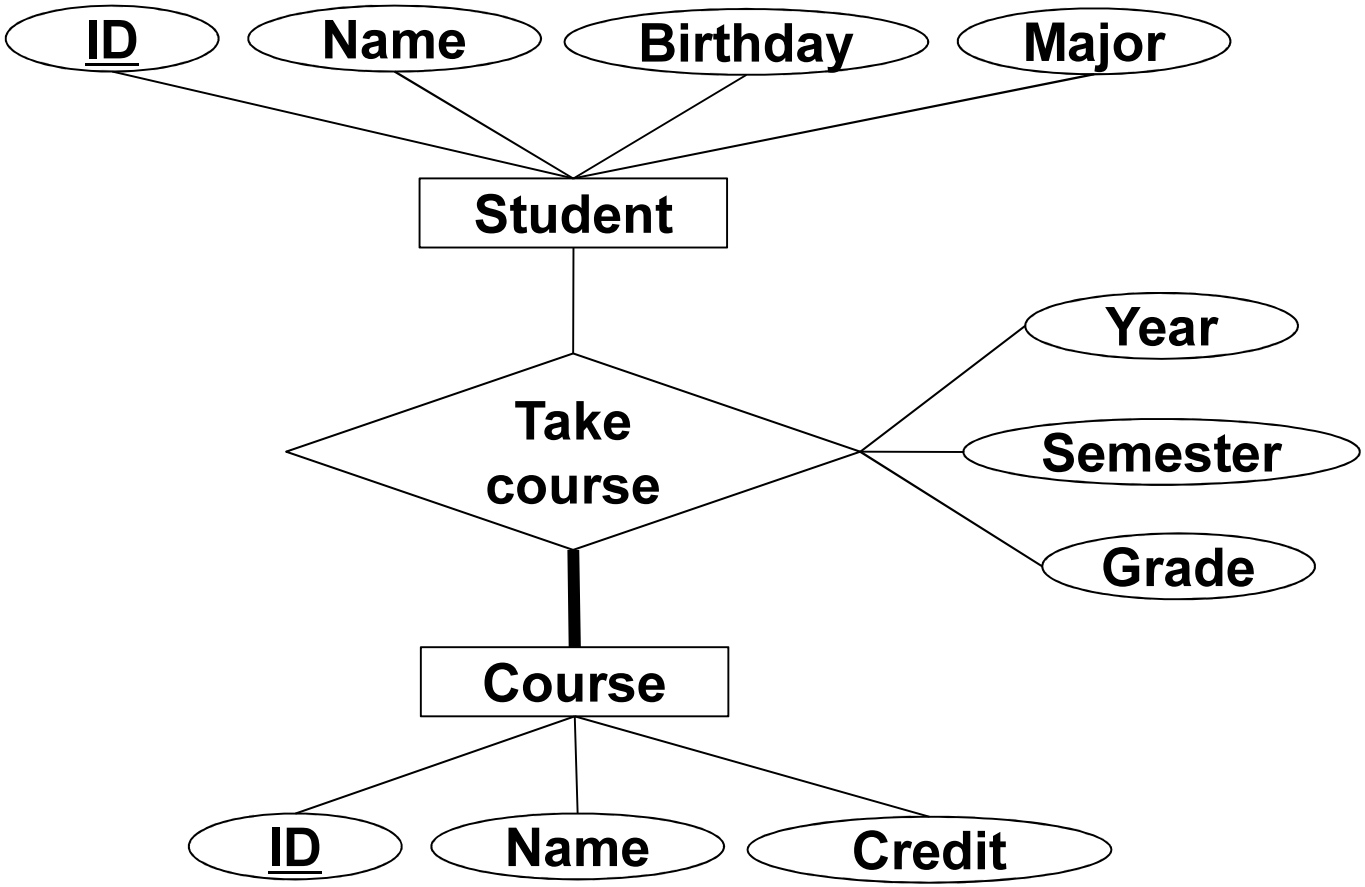
ER图



UML类图

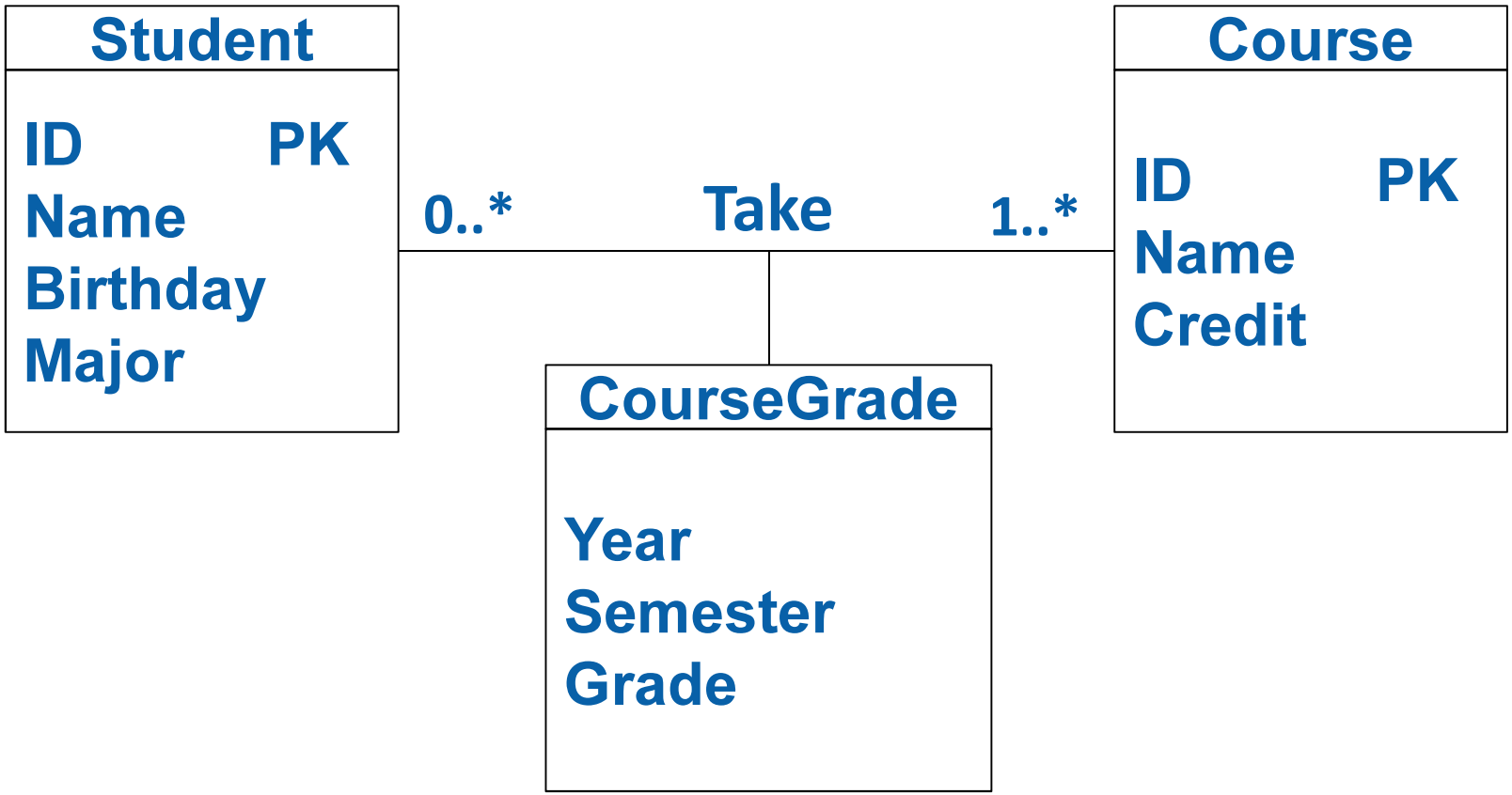


联系（ER图）

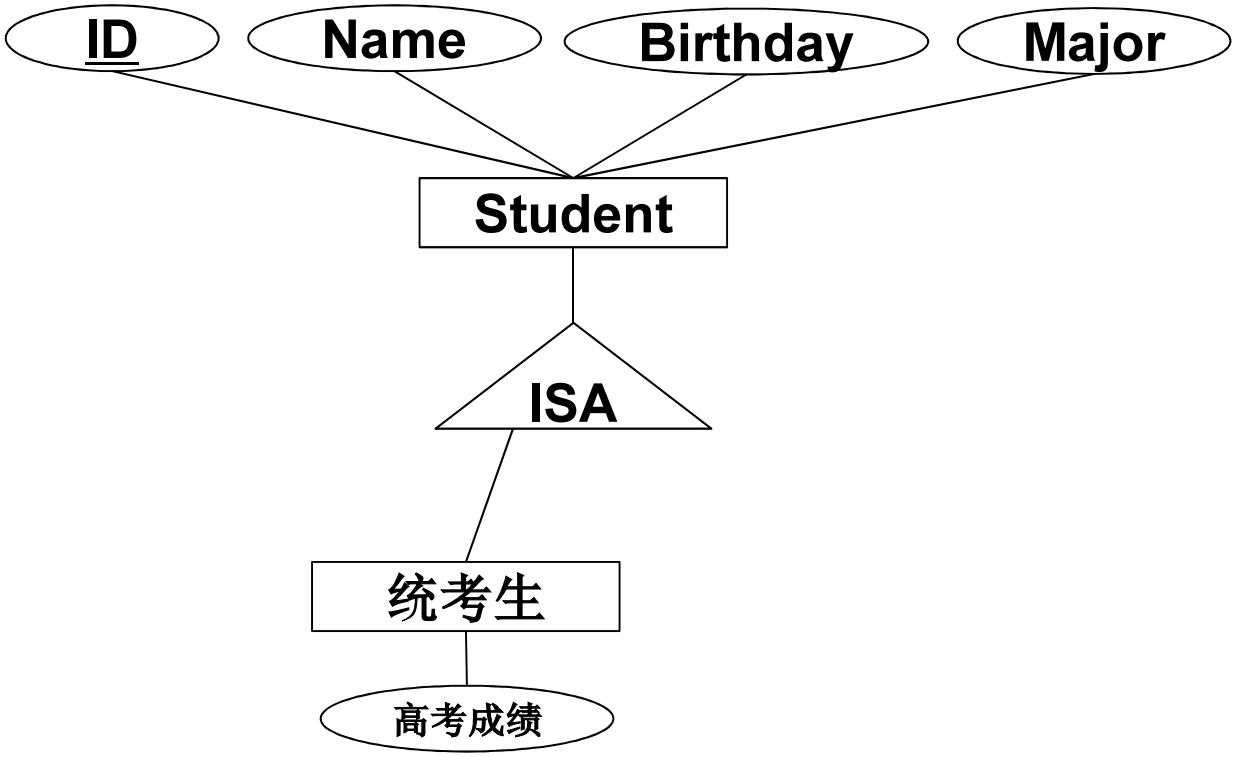

ID
Name
Birthday
Major
Student
Year
Take course
Semester
Grade
Course
ID
Name
Credit

联系（UML）

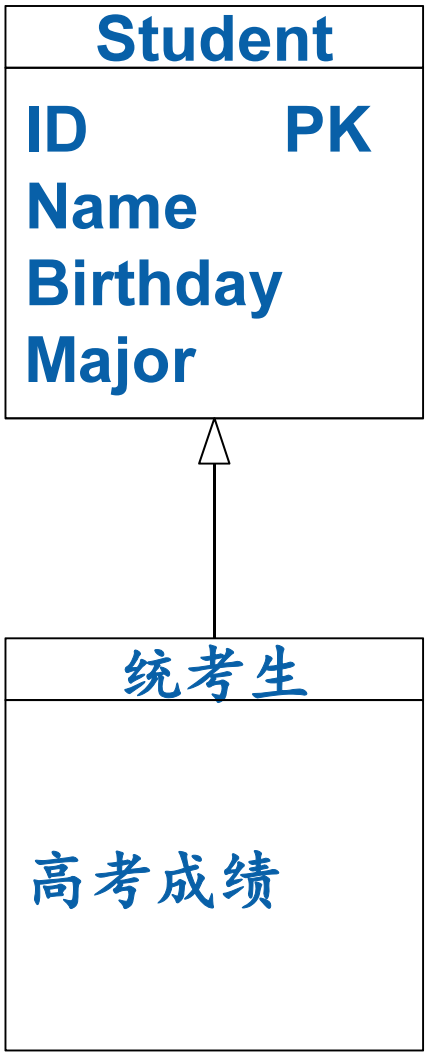
每个学生可以选0到多门课
每门课需要有1到多个学生选



子类继承（ER图）



子类继承（UML）



其它需要注意的

- Aggregation（聚合）
  - 这里我们特意不介绍Aggregation
  - ER图的聚合≠UML的聚合≠关系运算中的Aggregation操作
  - 更多的复杂性，容易混淆
  - 实际上，很多时候概念设计可以不用这些复杂的结构

Outline

- 数据库设计过程
- ER模型
- 应用举例
- UML模型介绍