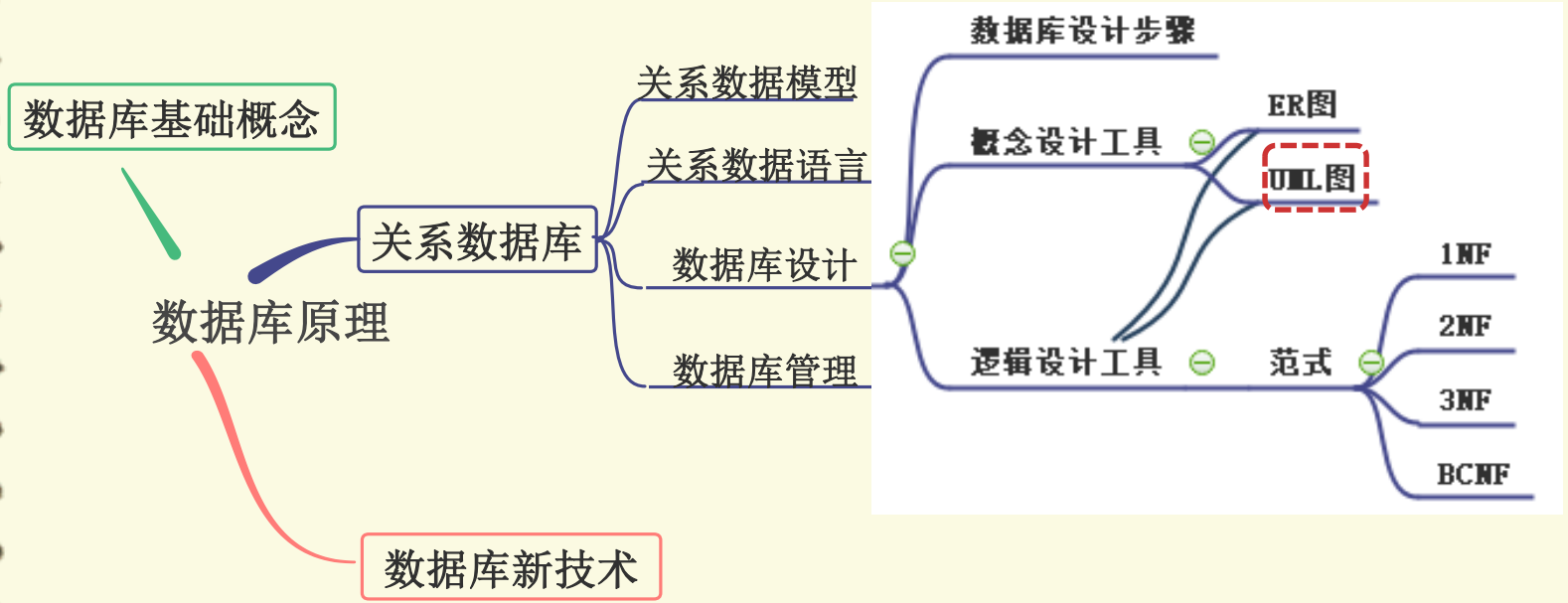
数据库原理

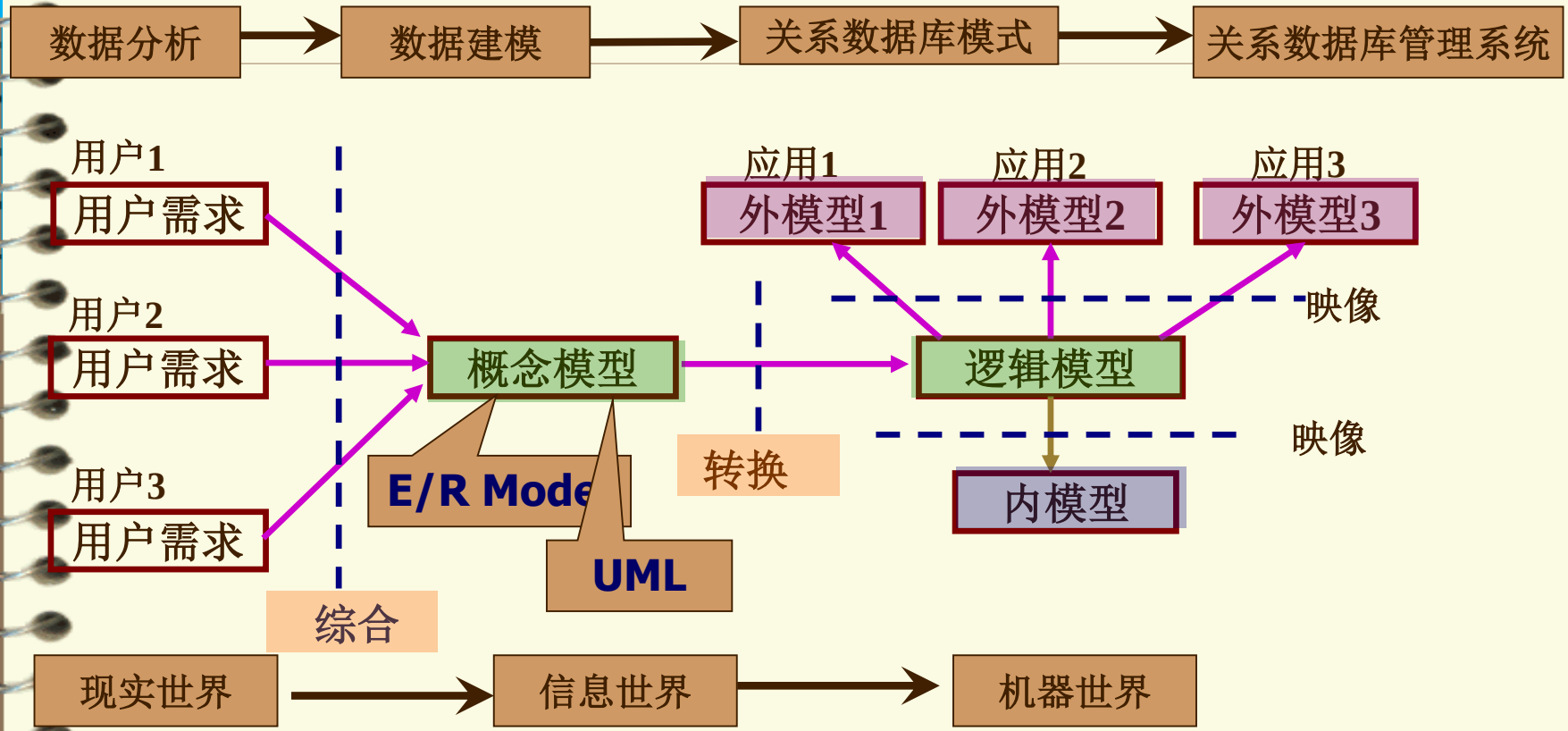
第5章 数据库设计

辽东学院 鲁 琴

本节要点


数据库基础概念
数据库原理
关系数据库
关系数据模型
关系数据语言
数据库设计
数据库管理
数据库新技术
数据库设计步骤
概念设计工具
ER图
UML图
逻辑设计工具
范式
1NF
2NF
3NF
BCNF

数据库设计过程


数据分析
数据建模
关系数据库模式
关系数据库管理系统
用户1
用户需求
用户2
用户需求
用户3
用户需求
概念模型
E/R Mode
UML
综合
转换
应用1
外模型1
应用2
外模型2
应用3
外模型3
映像
逻辑模型
映像
内模型
现实世界
信息世界
机器世界

数据建模

- 即对于一个特定的应用，如何在数据库中表示数据
- 设计关系模型方法：
  - ✓关系模型设计理论
  - ✓概念设计模型
    - E/R－－传统的
    - UML子集－－目前常用的

UML

1 UML模型

2 UML模型到关系模式的转换

UML

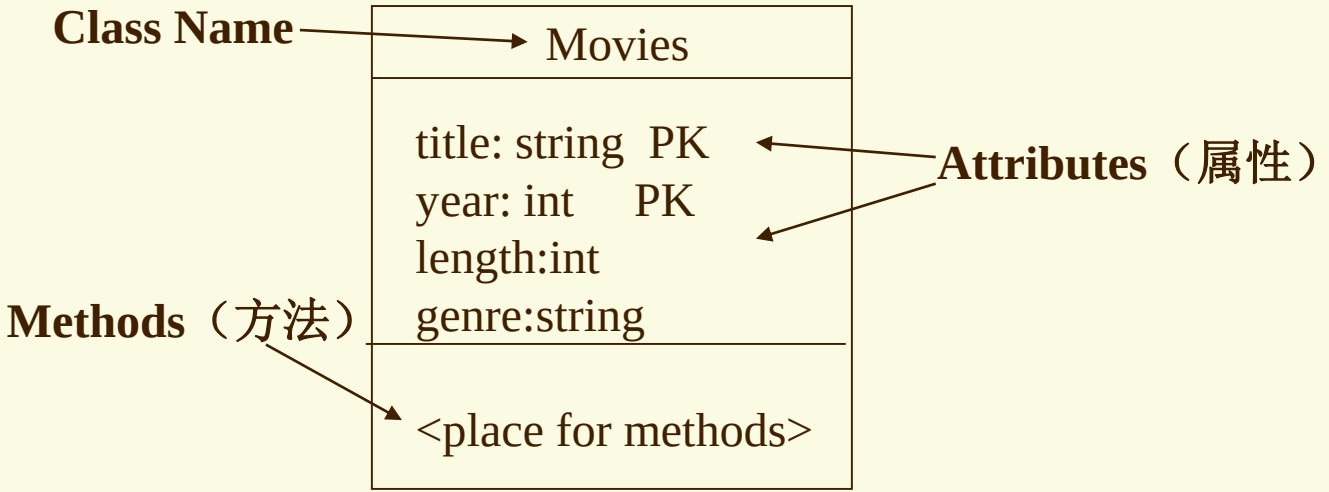
- Unified Modeling Language，统一建模语言
- UML 用于面向对象建模，但是现在也用于数据库建模
- UML 与 E/R模型相似，但是不提供多元联系

UML 和 E/R 术语对比

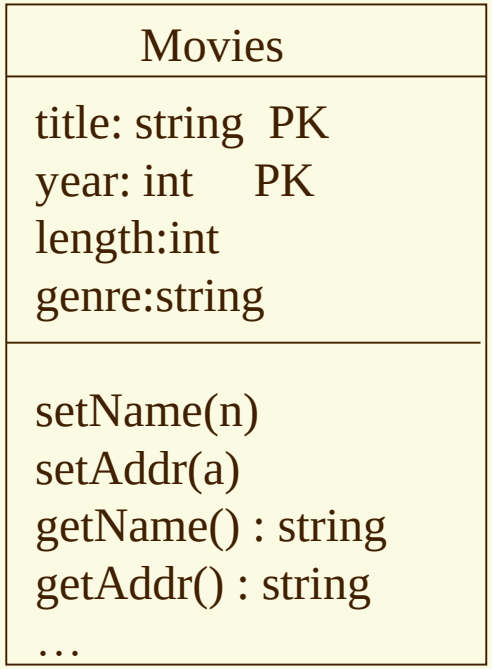
UML	E/R
Class（类）	Entity set（实体集）
Association（关联）	Binary relationship（二元联系）
Association Class（关联类）	Attributes on a relationship（联系的属性）
Subclass（子类）	Isa hierarchy（Isa层次关系）
Aggregation（聚集）	Many-one relationship（多对一联系）
Composition（组成）	Many-one relationship with referential integrity（带参照完整性的多对一联系）

UML 类

UML中的类与 E/R中的实体集相似

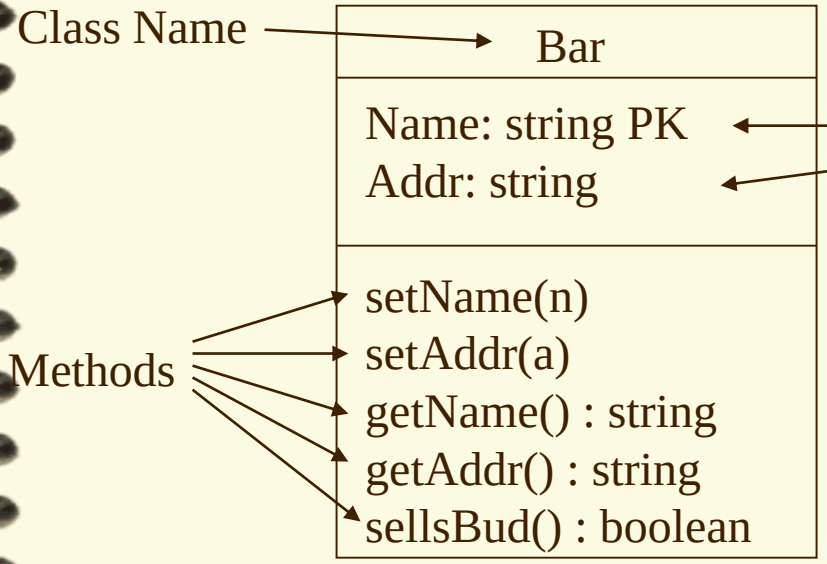


UML 类



- 类是具有相同属性和方法的对象的集合
- 属性是静态的，是状态，具有数据类型
- **PK** 表示主键
- 方法是动态的，是行为，包括参数的声明和返回值的声明

实例：Bar Class



当**UML**用于数据建模时

删除方法

增加主键

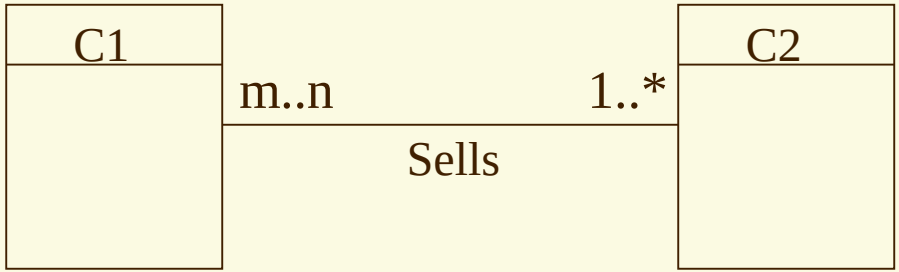
属性类型可选

Associations（关联）

- 两个类间对象的联系称为关联（association）
- 表示方法
  - ✓两个类间用直线（箭头可选）连接
  - ✓连接名字通常写在直线下方
  - ✓在直线连接两个类的端点处标上关联类型（Multiplicity）

关联类型（Multiplicity）

- 在直线连接两个类的端点处标上关联类型（**Multiplicity**）
  - ***m ..n*** 表示与C2类中一个对象有关的C1类对象的个数最少为 **m**，最多为 **n**。
  - * 表示 “无上限”；例如 **1..*** 表示 “最少一个”



关联类型（Multiplicity）

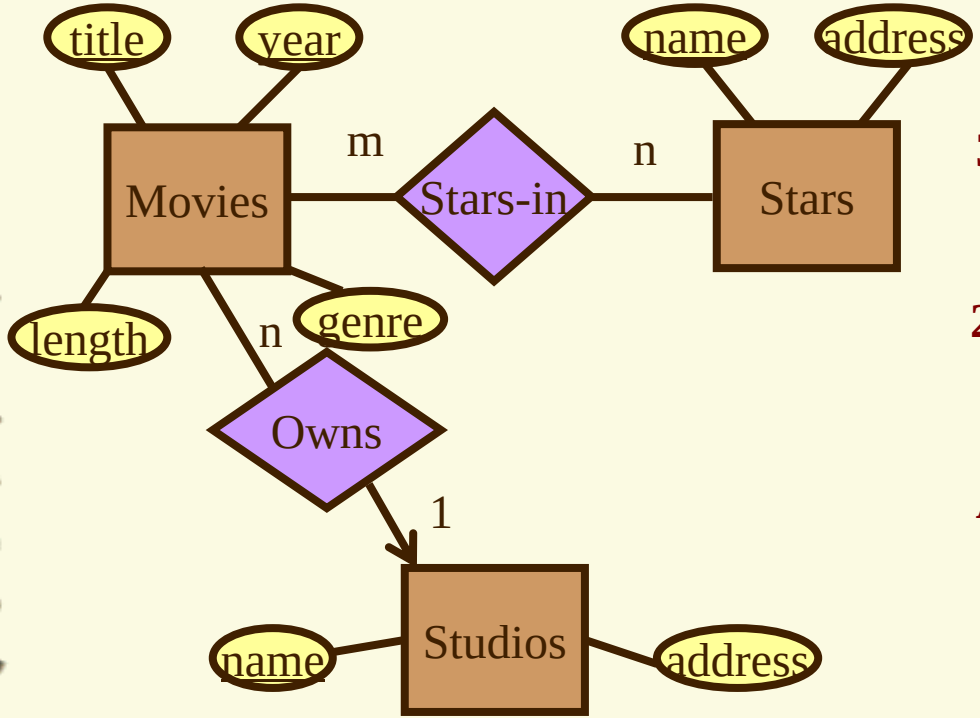
- 在直线连接两个类的端点处标上关联类型（**Multiplicity**）
  - ***m*** **..*** 表示“没有上限”
  - **0..n** 表示“没有下限”
  - **0..*** 表示“根本没有限制”

每个学生最多能在5个校园申请课程
每个校园容纳学生数为10000到20000之间

关联类型（Multiplicity）

- 关联类型（Multiplicity）简写和默认值
  - “*”是“0..*”的简写
  - “1”是“1..1”的简写
  - 默认值为“1..1”

实例：Movie数据库设计 E/R图

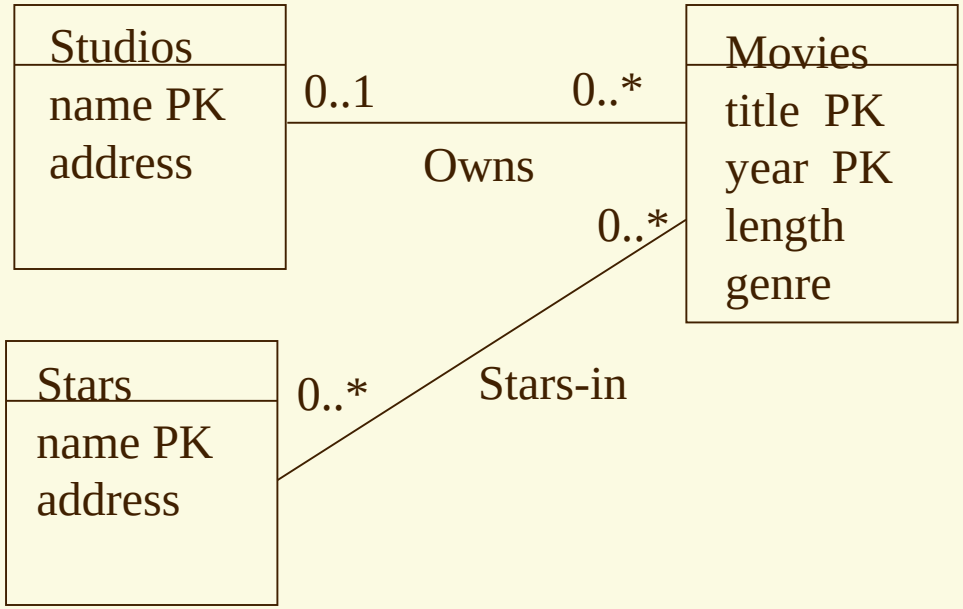


3 entity sets

2 relationships

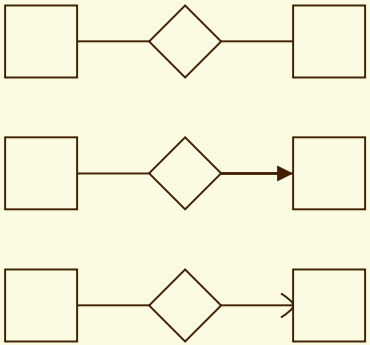
Attributes

实例：Movie数据库设计UML图

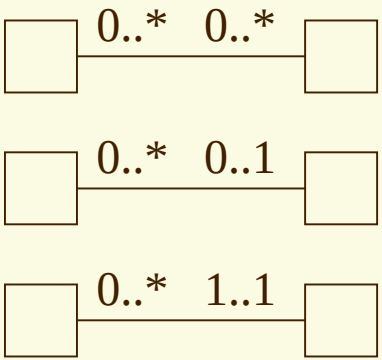


UML和E/R关联类型对比

E/R



UML

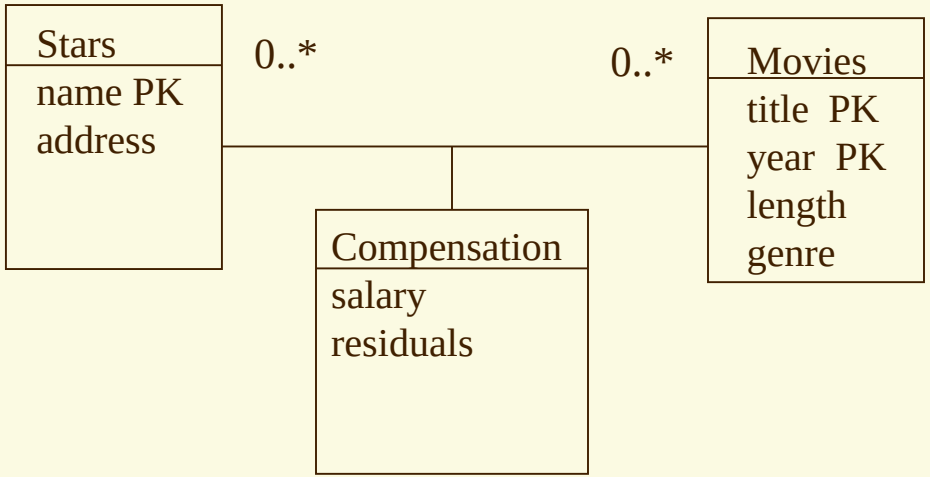


关联类（Association Classes）

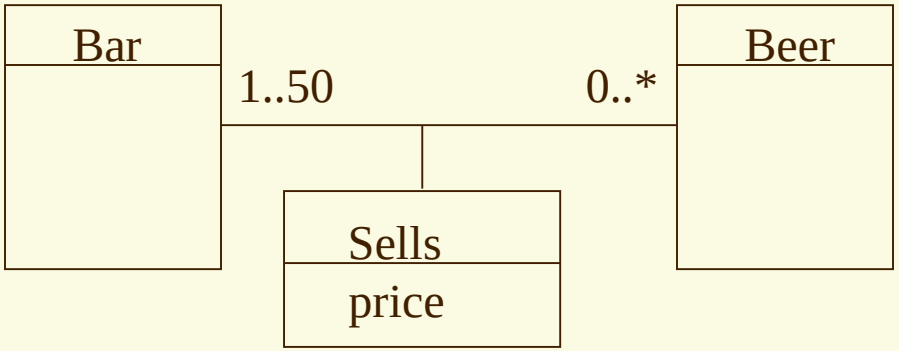
关联也可以有属性

- 称为关联类 （*association class*）.
- 与E/R图中联系的属性类似

实例：Association Class

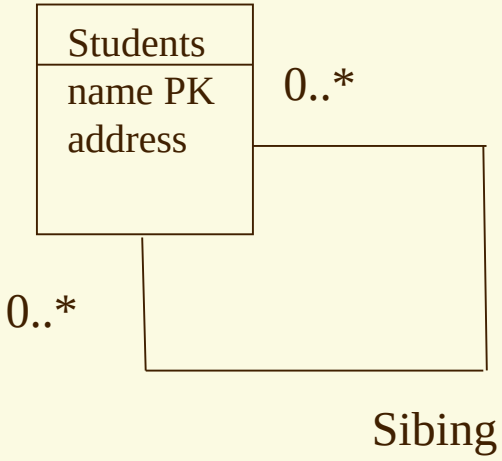


Example: Association Class

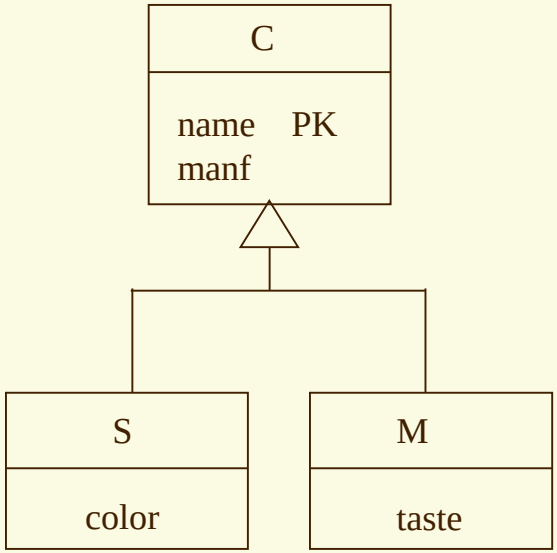


自身关联（Self-Associations）

类自己与自己关联

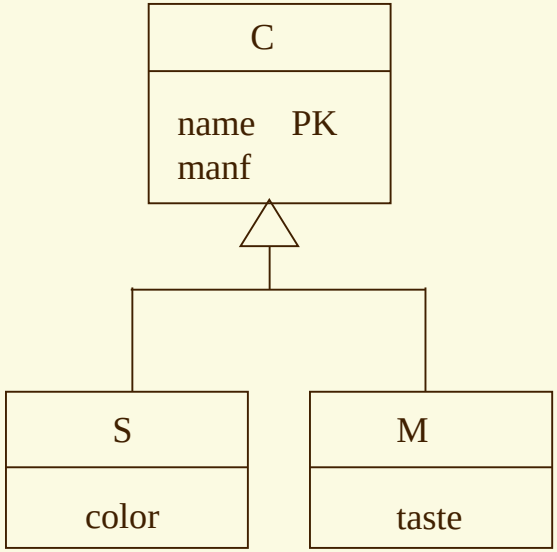


子类（Subclasses）



- UML类都可以包含下级子类
- 子类用连线连接父类，与父类连接处以空心三角指向父类
- 主键来自父类（Superclass）
- 子类继承父类的属性（包括attributes and associations)
- 子类可以有子类自己的属性以及与其它类的关联

子类的四种类型



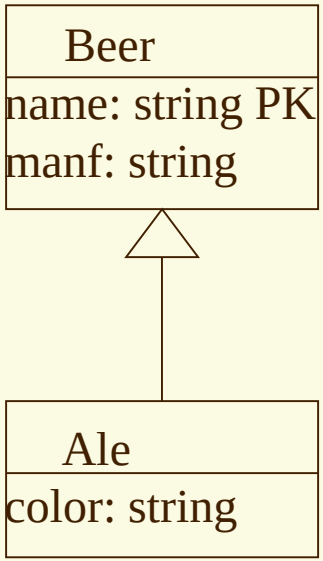
UML 允许4种类型的子类

- ***Complete***（完全） **(父类中的每个对象都 是某个子类的成员) 或 *partial***（部分）**.**
- ***Disjoint***（分散）**(一个对象不能包含在两个子类中）或 *overlapping***（重叠）**.**

子类

- 在Object-Oriented系统中，子类是 *disjoint*—即两个子类中不存在同一对象。
- E/R 模型自动允许*overlapping*子类
- E/R 模型和 OO系统都允许 *complete* 或 *partial* 子类

Example: Subclasses



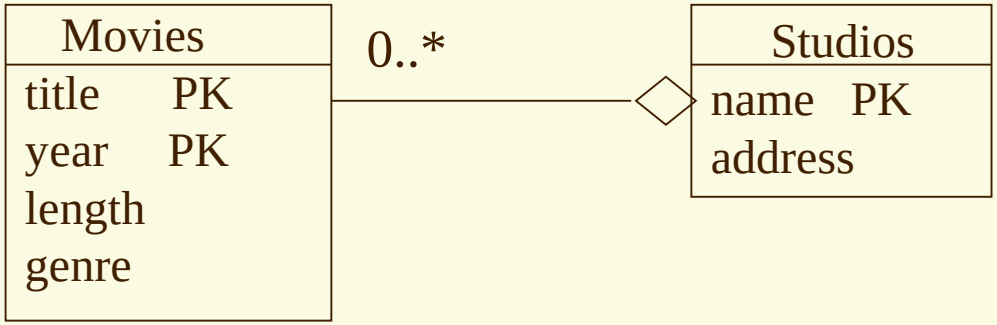
聚集(Aggregations)
组成(Compositions)

有两种类型的多对一(n:1)关联(many-one associations)

- 聚集(Aggregations)
- 组成(compositions)

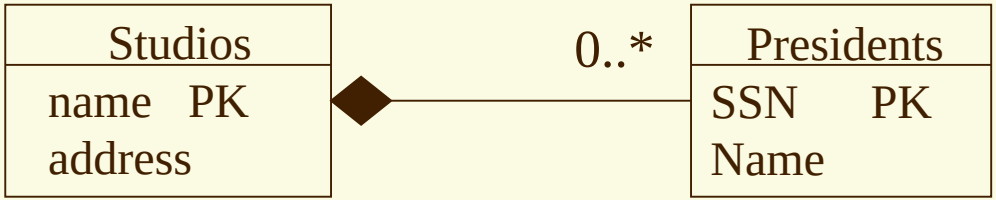
聚集(Aggregations)

- 聚集用连线连接两个类,一方以空心菱形箭头结束
- 空心菱形箭头指向一方参与对象的个数必须为 **0..1**,不需要另外标注



组成（Compositions）

- 组成与聚集类似.
- 但是菱形箭头一方参与对象必须为 **1..1.**
- 菱形箭头相反一方类的每个对象必须与菱形箭头方的一个对象关联
- 组成以实心菱形表示



画出 UML，并转换成关系

图书馆中有多本书，每本书都可以借给不同的读者。当一本书借出是它的状态（ status）为 “borrowed out”, 当这本书被归还时状态为“in library”. 每个读者一次可以借至少一本书，并且可以在不同时间借阅同一本书. 同时还需记录借书时间和还书时间.

读者（Readers）的属性包括ID, name, telephone和department;

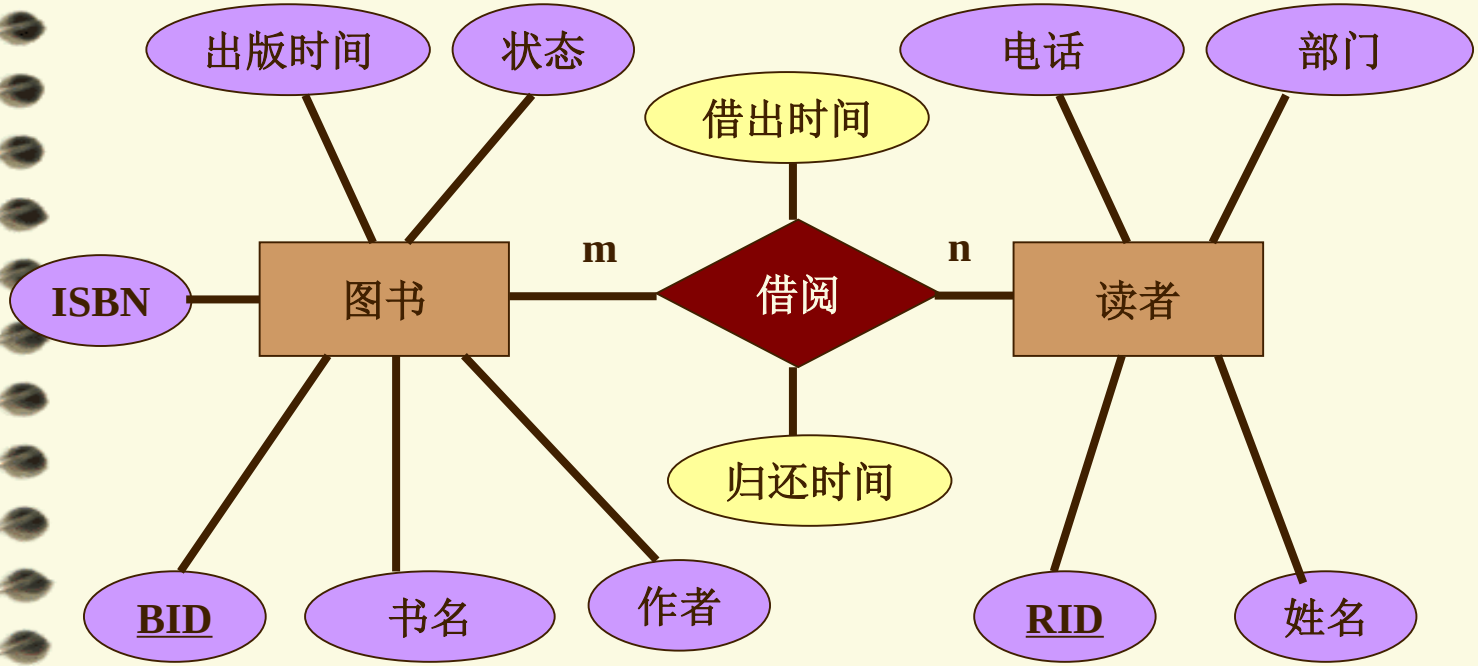
书（Books）的属性包括ID, bookname, author, date of print和 status.

实例：数据库设计

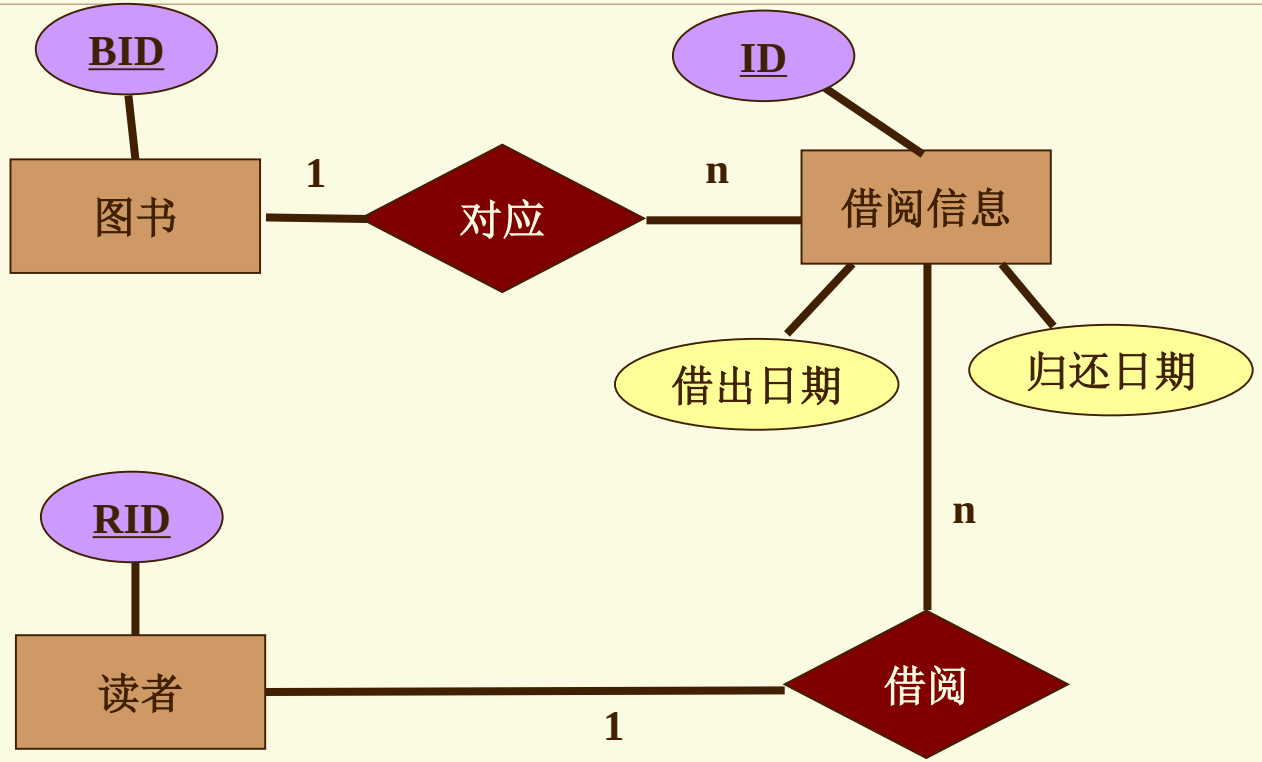
图书馆图书借阅管理系统，其主要功能有读者管理，书籍管理和借阅管理。

- （1）需要管理读者的ID，姓名，电话号码和所在部门
- （2）需要管理书籍的ID，书名，作者，出版社，ISBN，出版时间和状态（在库或借出）
- （3）需要记录图书借阅信息，包括借书时间和归还时间。每个读者可以借多本书，每本书在不同时间可以被不同用户借阅

图书 (BID, 书名, ISBN,作者, 出版时间,状态)
读者 (RID, 姓名, 电话 ,部门)
借阅 (RID,BID, 借出时间, 归还时间)

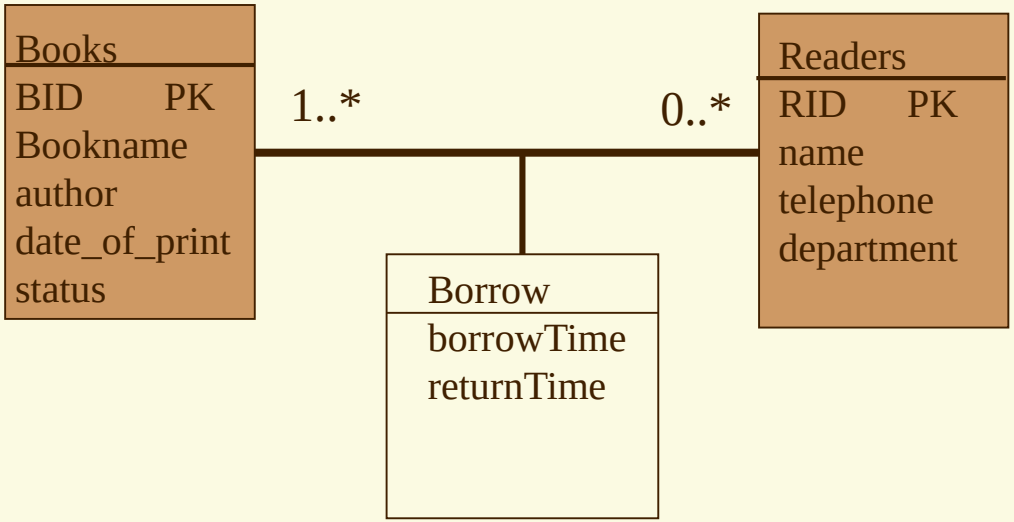


图书 (BID, 书名, ISBN,作者, 出版时间,状态)
读者 (RID, 姓名, 电话 ,部门)
借阅 (ID,RID,BID, 借出时间, 归还时间)



UML ⇨ Relations

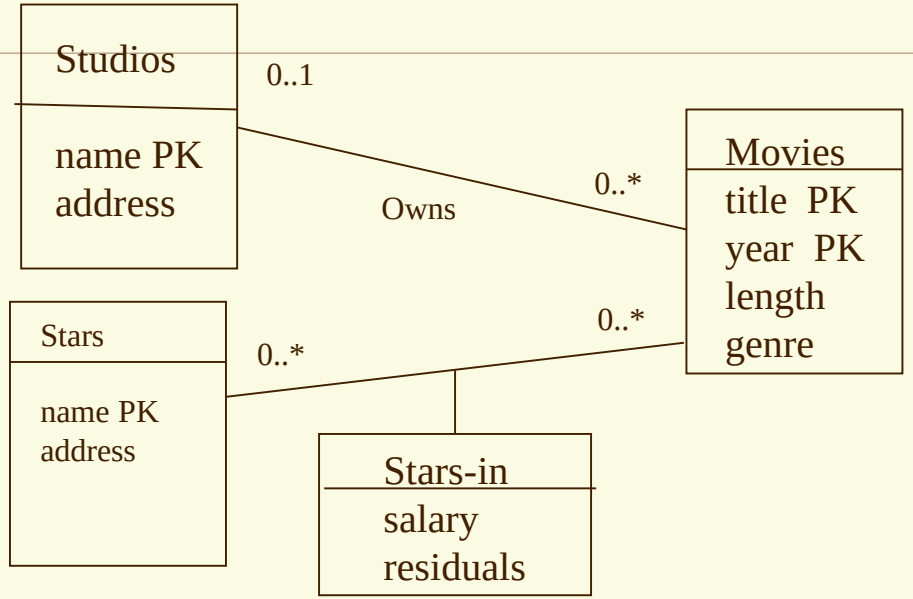
Reader (RID, name, telephone ,department)
Book (BID, bookname, author, date_of_print ,status)
RB (RID,BID, borrowTime, returnTime)



UML 转换为关系

1. 类的转换（Classes to relations）
2. 关联的转换(Associations to relations)

实例



Movies (title, year, length, genre,studioname)
Stars (name, address)
Studio (name, address)
StarIn (movietitle, movieyear, starname,salary,residuals)

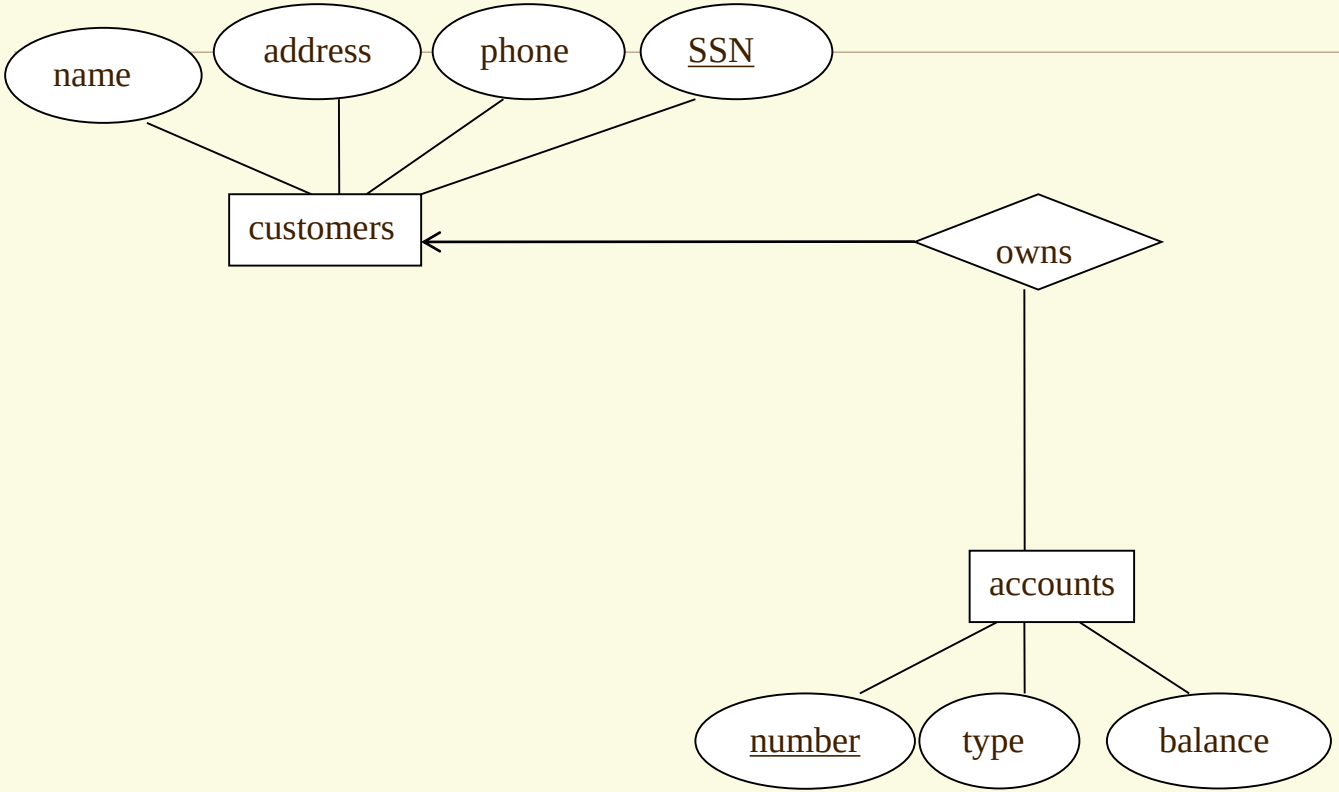
UML 子类转换为关系

1. **E/R风格: 每个子类关系只存储其自身属性和码**
2. **OO风格: 子类关系存储其自身和其父类所有的属性**

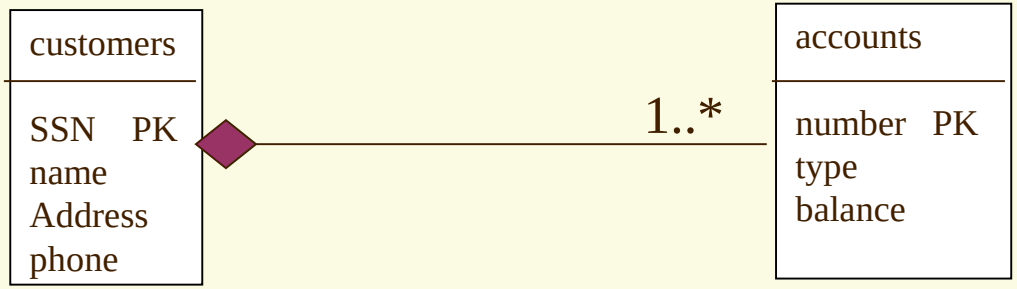
实例： Give UML and then convert to relations

设计一个银行数据库，包含客户（customers）信息和对应的账户信息（accounts）。客户信息包括客户name, address, phone, 和 SSN. 账户信息包括 types(取值为savings或 checking) 和 balances.

E/R


name
address
phone
SSN
customers
owns
accounts
number
type
balance

UML

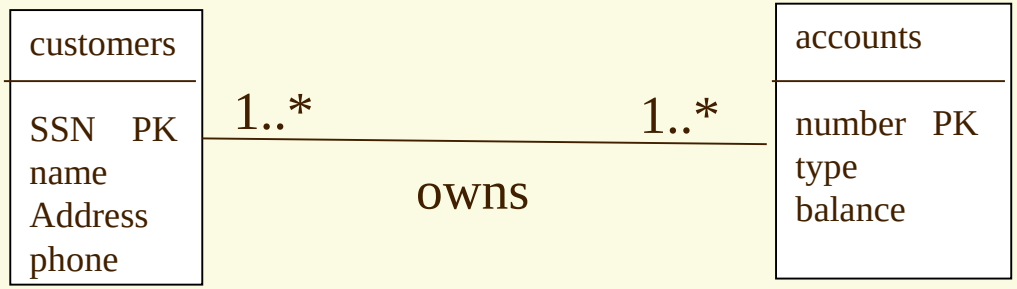


costomers(SSN,name,Address,phone)
Accounts(number,type,balance,costomerSSN)

或

costomers(SSN,name,Address,phone)
Accounts(number,type,balance)
owns(SSN,number)

UML



costomers(SSN,name,Address,phone)
accounts(number,type,balance,costomers)
owns(SSN,number)