

数据库建模2

- 初级实体联系图

实体集 属性 联系

- 高级实体联系图


约束 子类 弱实体集


- 实体联系图到关系模型的转换

E/R图

◆ **实体联系图**(E/R图, Entity-Relationship Diagrams): 数据库建模的一种图形方法, 它用图形的方法, 描述实体及实体间的联系。

◆ 构成三要素

■ **实体集**(Entity sets): 具有相同特性的同一类事物的聚集。客观存在并且可以相互区别的事物称为**实体**(Entities)。 

■ **属性**(Attributes): 描述实体某个特性的值。 

! 属性的数据类型不能是聚集类型, 即不能是集合、数组、列表、结构等包含多项内容的数据类型。

■ **联系**(Relationships): 两个或多个实体集之间的连接关系。 

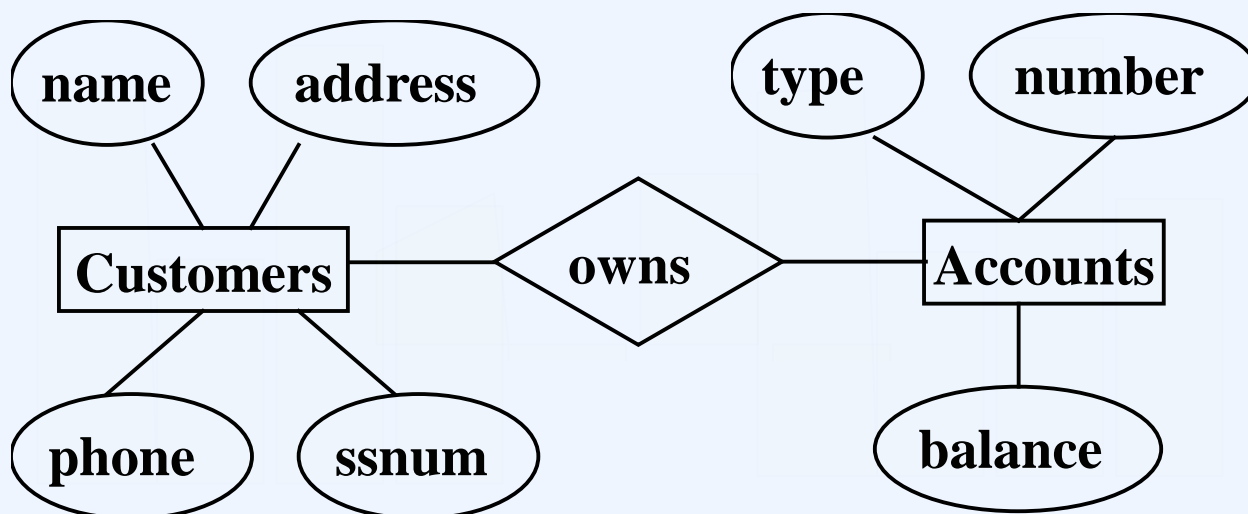


E/R图举例

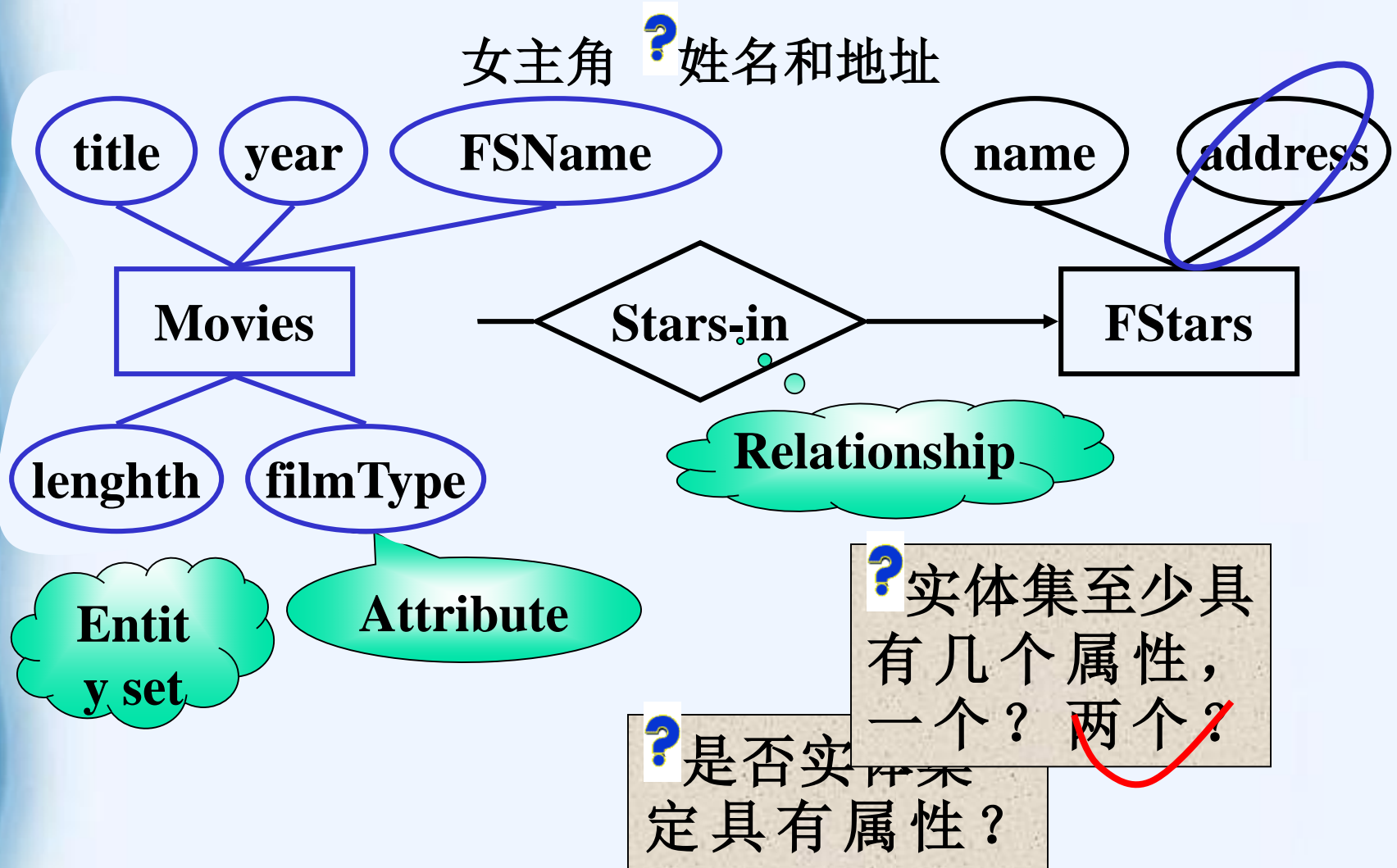
为银行建立数据模型，要求保存如下信息：

- 顾客：身份证号、姓名、地址和电话。
- 帐户：编号、类型(例如存款，支票)和结余。
- 记录顾客所拥有的银行帐户。

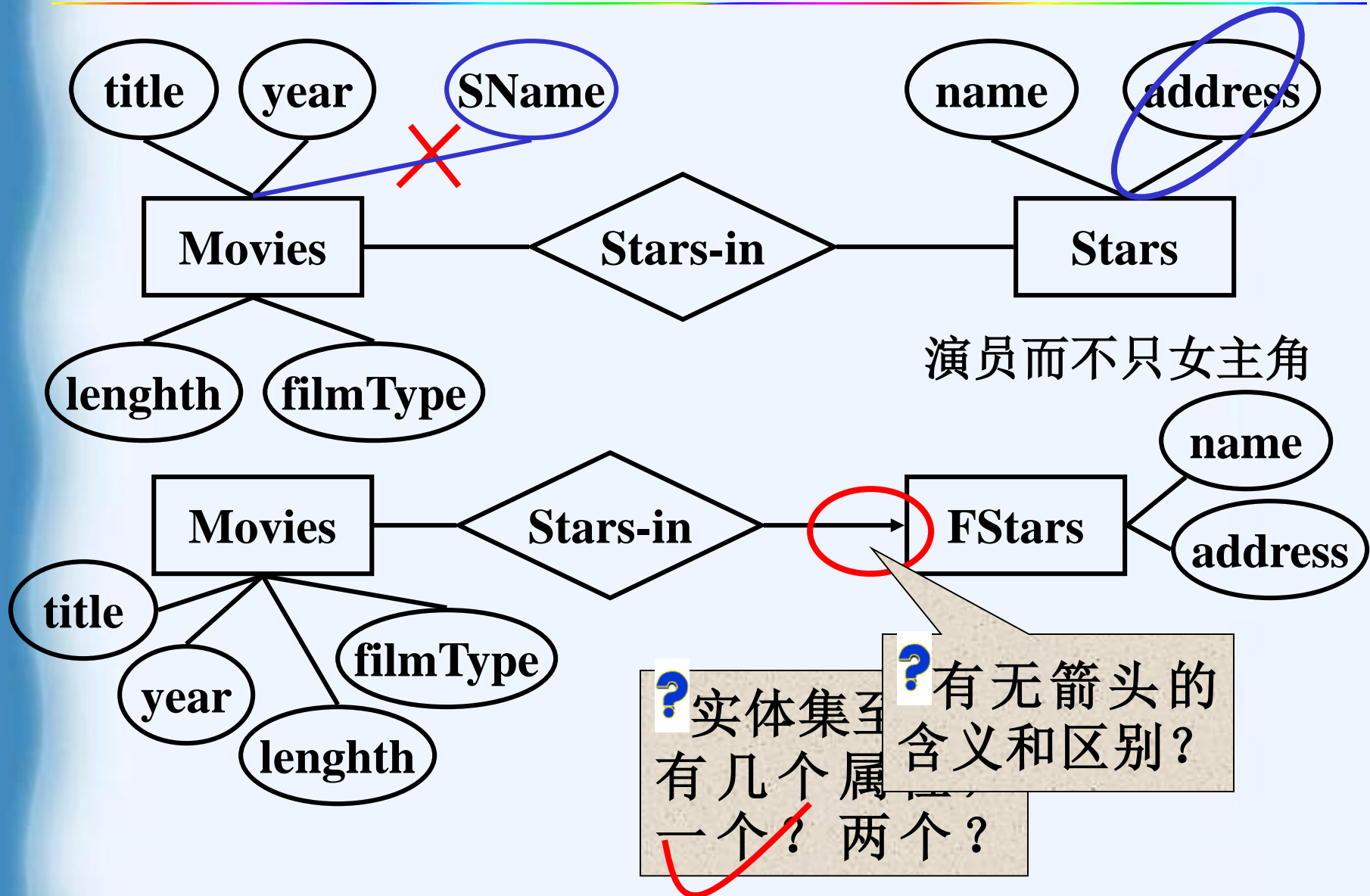
对这个数据库用E/R图进行描述。



E/R三要素-实体集和属性



E/R三要素-实体集和属性



E/R三要素-联系

◆E/R图中联系的特性:

- 具有多重性(多对多、多对一、一对一)。
- 多向性(联系可以涉及多个实体集)。
- 可以具有多种角色(一个实体集在一个联系中可以出现多次)。
- 可以具有自己的属性。

◆1、多重性: 包括一对一联系、一对多联系和多对多联系。

■**一对一联系** 即两个实体集之间每个实体都是一对一对应。例如两个实体集班级和班长。

■**一对多联系** 即两个实体集A和B之间, 实体集A中至少有一个实体对应着B中的多个实体, 而B中每个实体都唯一对应着A中的一个实体。例如实体集银行帐户和客户。

■**多对多联系** 即两个实体集A和B之间, A至少有一个实体对应B中的多个实体, B至少有一个实体对应A中的多个实体。例如实体集学生和教师之间。



联系的多重性

◆联系的多重性的表示方法：

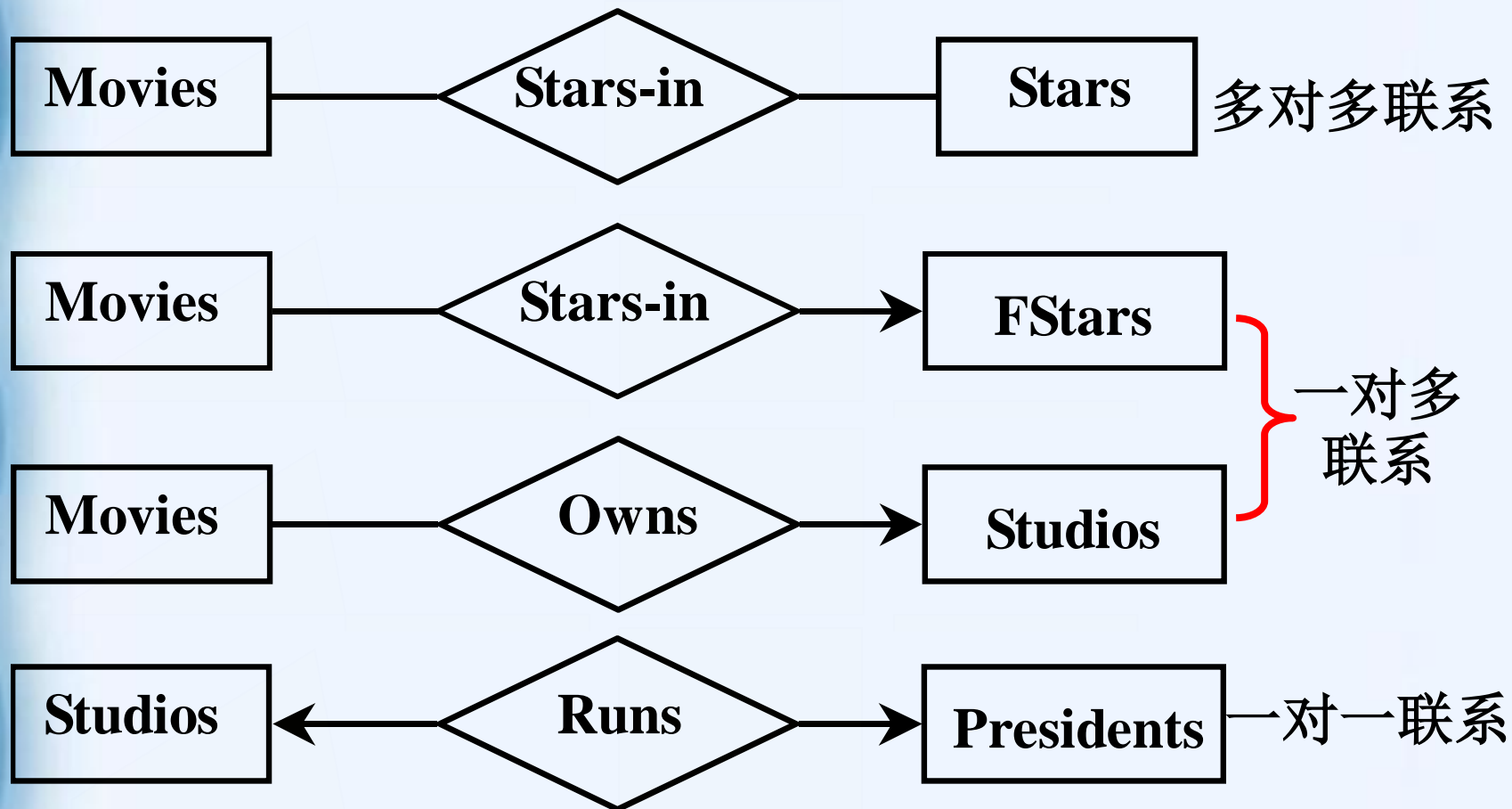
- 在E/R图中用连线的箭头表示。在仅有一个的那一方打箭头
- 对从E到F的一对一联系，分别画一个箭头指向E和F。
- 如果一个联系是从E到F的多对一联系，那么就画一个指向F的箭头。
- 对于多对多联系，不需要使用箭头。

❖确定是否该划箭头的方法：

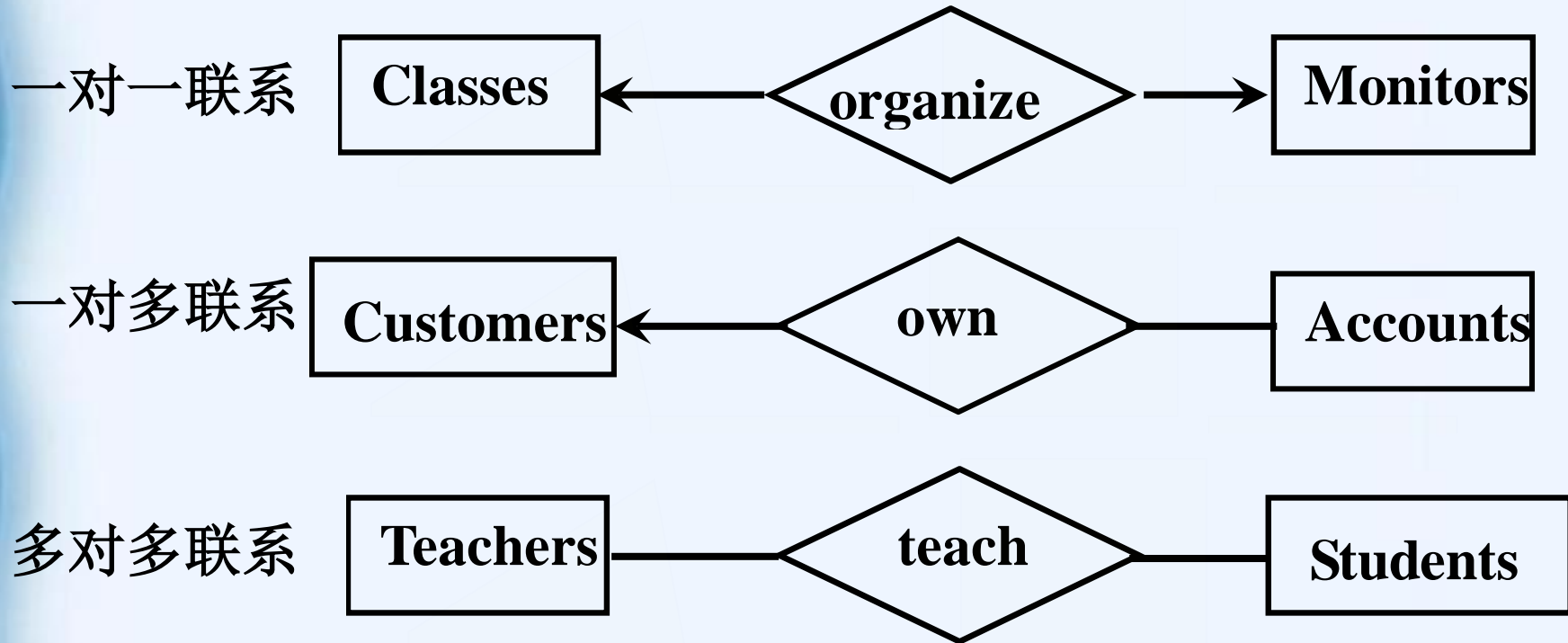
如果这个联系可以唯一地确定该实体集中的一个实体，则连向该实体集的连线应该加箭头，否则不加。



联系的多重性-例



联系的多重性-例

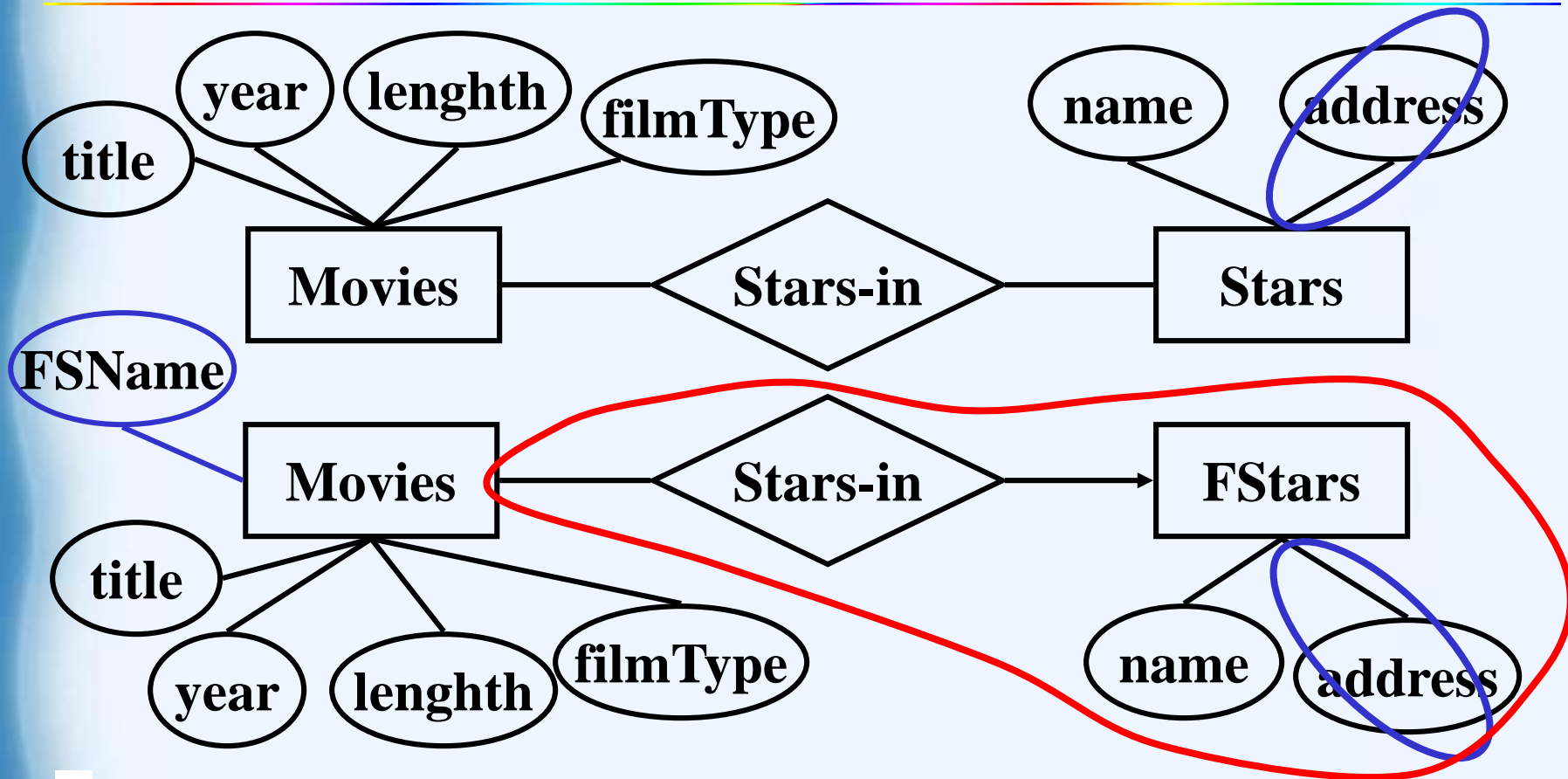


? 若两个实体集A和B，A是包含多的一方，是否说明B中所有实体都与A中的多个实体有关呢？

! 多对一联系是多对多联系的特例，而一对一联系是多对一联系的特例。



E/R图举例



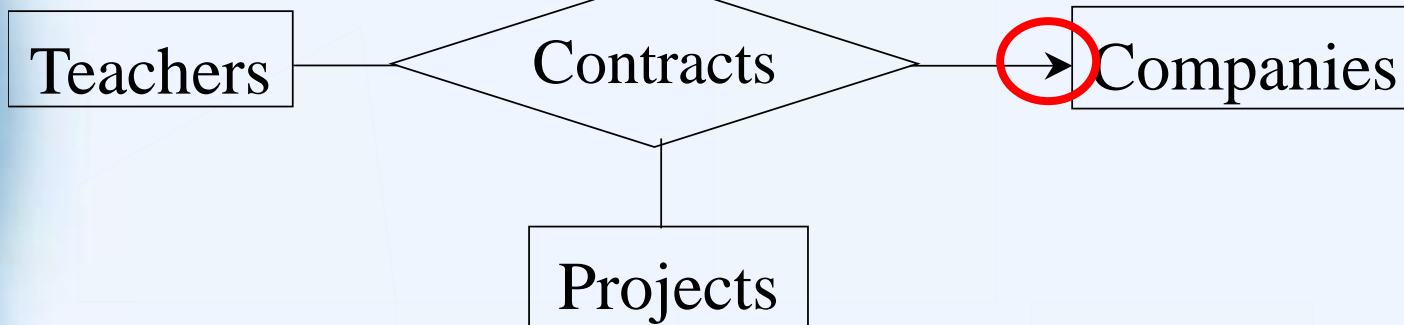
? 实体集至少具有几个属性，一个？两个？

! 在联系中包含多的一方只需有一个属性，包含唯一的一方至少需要两个属性。



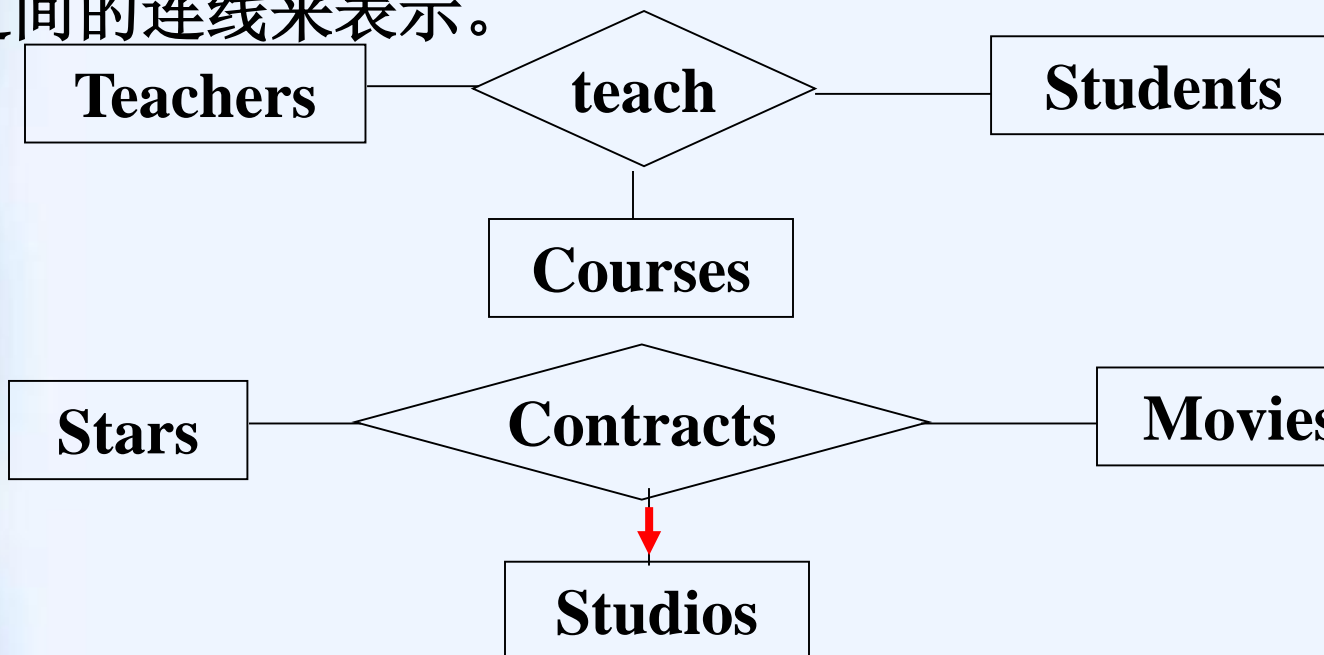
联系的多向性

例：希望记录教师、公司和项目三者之间签约的联系。



◆ 2、**多向性**：E/R模型中的联系可以涉及两个以上实体集。

E/R图中的多向联系通过从菱形的联系到所涉及的每个实体集之间的连线来表示。



? 为何无箭头指向任一实体集?

? 判断箭头指向?



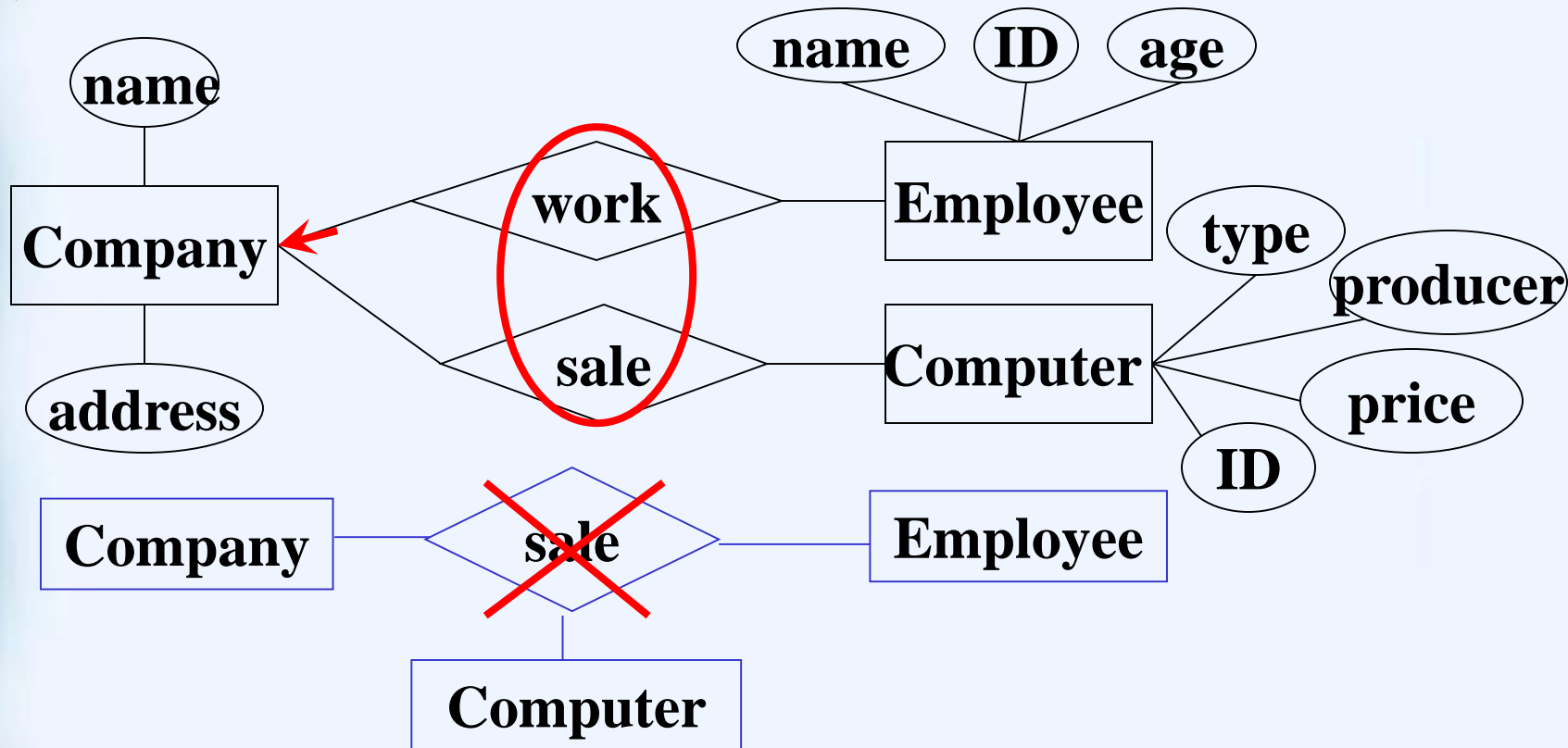
E/R图-例

描绘电脑销售数据库，要求在数据库中保存以下信息：

1) 电脑销售公司：公司名称、地址、雇员、销售的电脑。

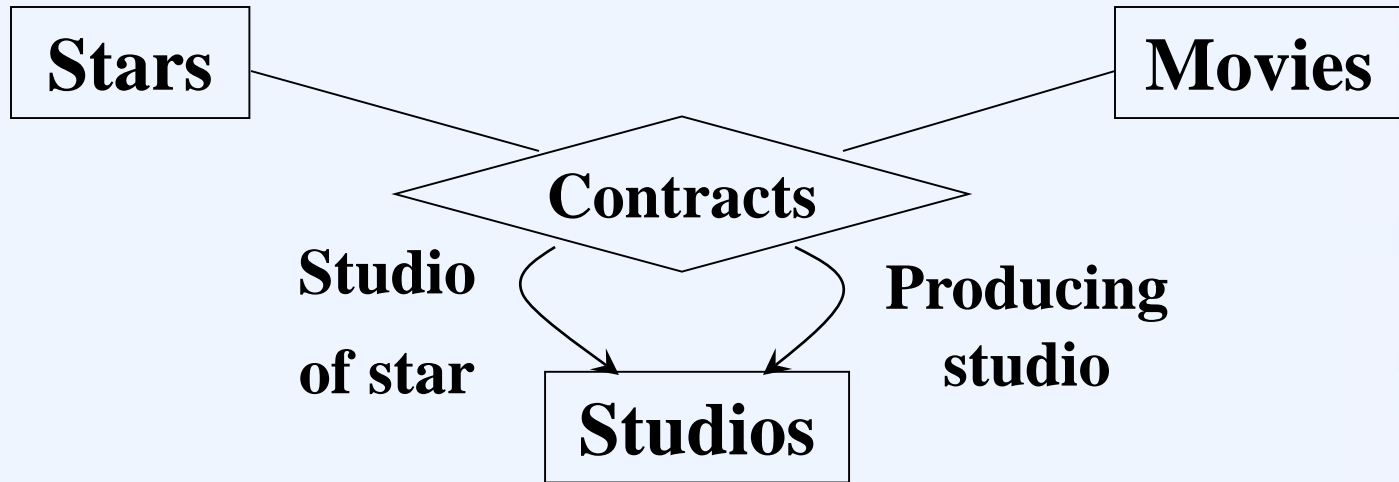
2) 销售公司雇员：雇员号、姓名、年龄。

3) 电脑信息：型号、制造商、价格、销售编号。



E/R图中的联系

◆3、联系的角色

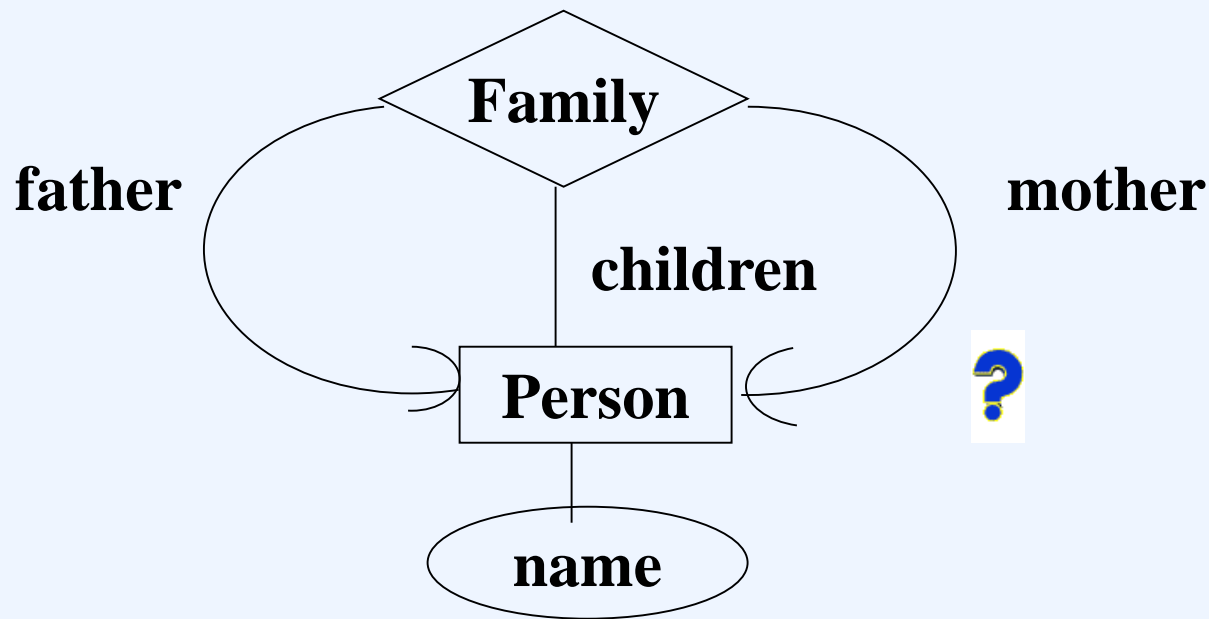


E/R图中的联系 例

设计一个家谱数据库中的类**Person**，要求：

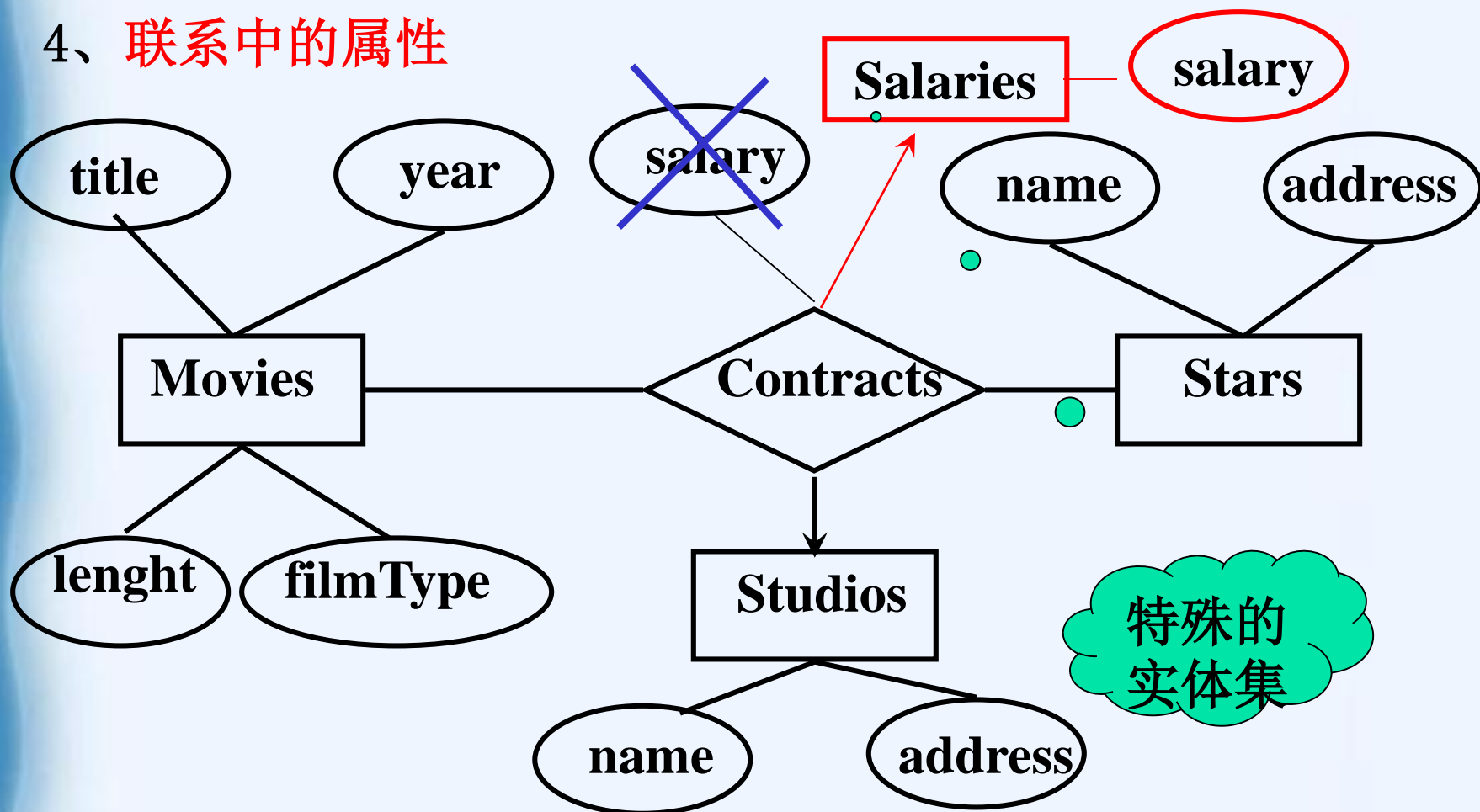
- 记录每个人的姓名。
- 记录联系：母亲、父亲和孩子。

对这个数据库用**E/R**进行描述。



E/R图中的联系

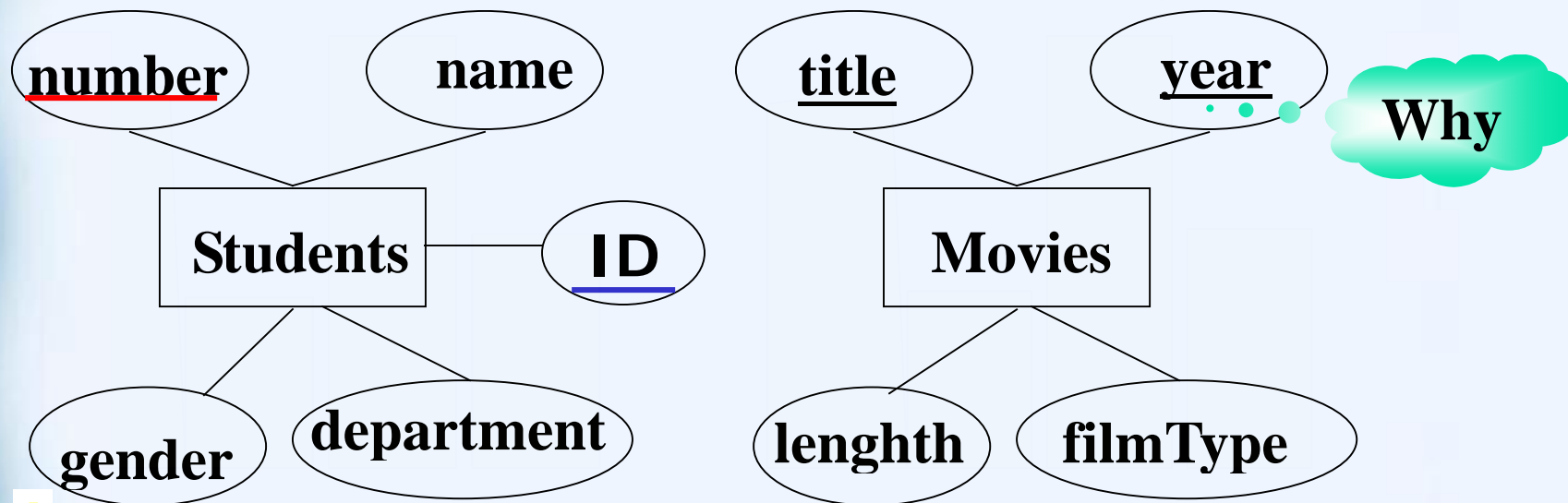
4、联系中的属性



E/R图中的约束

❖ 1、**键码(keys)** 是在实体集的范围内唯一标识一个实体的属性或属性集。一个实体集中的两个实体在构成键码的属性集上的取值不能相同。

■ 在E/R图中，用下划线表示某些属性或属性集构成了键码。



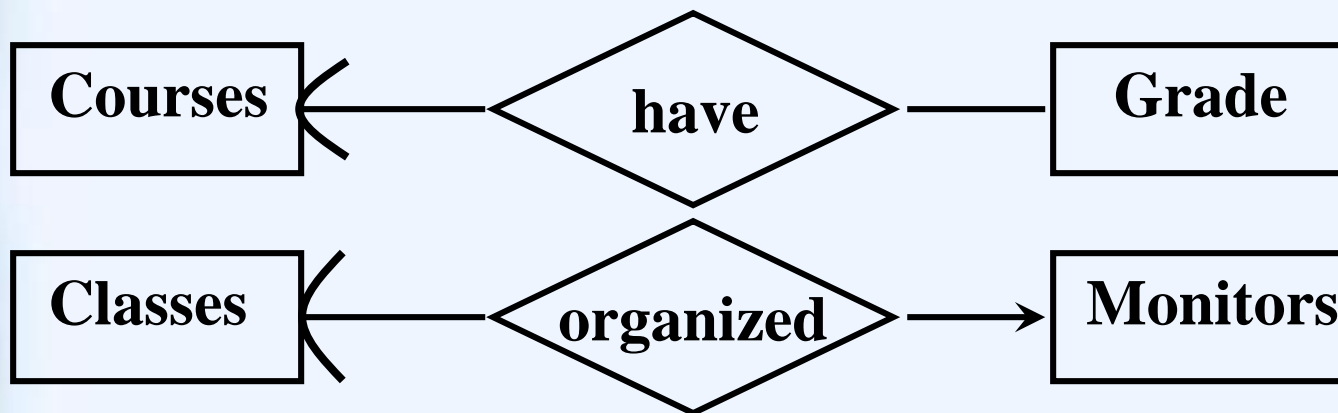
? 若一个实体集有多个键码，如何表示？

主键码在图中用下划线表示，其他键码可在附注中表示。



E/R图中的约束

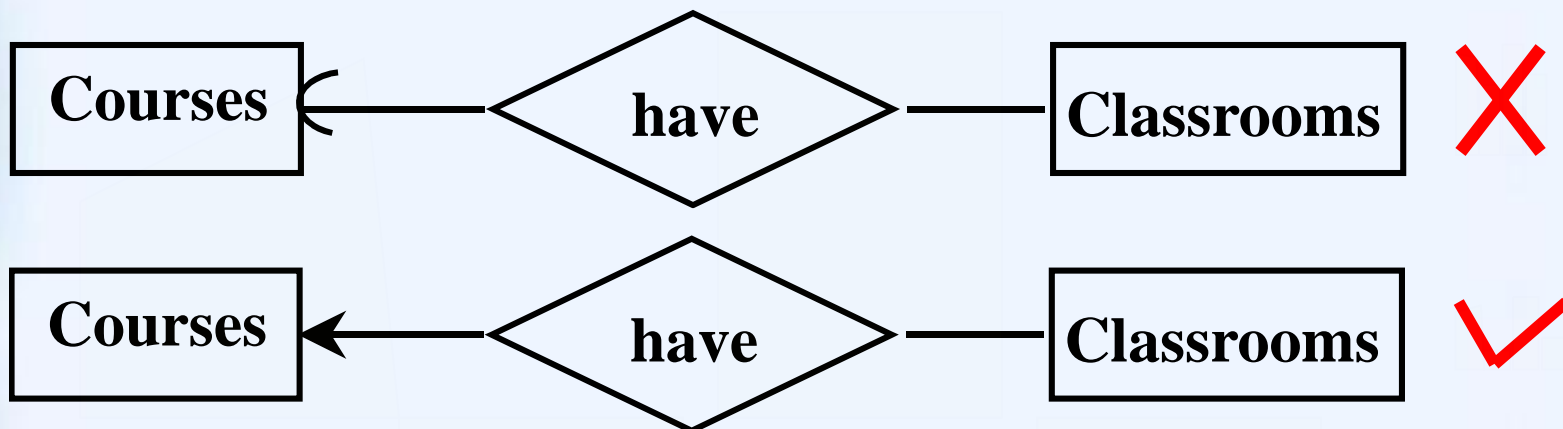
- ❖ 2、**参照完整性约束**(reference integrity constraints) 要求由某个实体引用的另一个实体集中的实体在数据库中确实存在。
- 假定R是一个从实体集E到实体集F的联系。如果E和F之间存在**参照完整性**，则用一个指向F的**圆箭头**来表明不仅联系R是由E到F的多对一或一对一的联系，而且对于实体集E的一个给定实体，要求与之相关的实体集F的实体一定存在。



❗ 多对一联系或一对一联系并不总要满足参照完整性约束。



E/R图中的约束



❖ 3、联系度的约束 比如一个同学至少要有10门课程的成绩，这种限制就属于联系的度的约束。

在E/R模型中，可以把一个极限数附在联系和实体集的连线旁边，以表明限制与有关实体集的任何一个实体相连的实体数。



E/R图注意事项

◆在做E/R图相关的题目时，注意：

- 一个信息不要重复表达！
- 首先要分析题意，看看究竟有哪几个对象需要作为实体集处理。
- 然后分析题目所要求保存的各实体集之间的联系，是多对一，还是一对一，还是多对多。
- 分析是否存在约束，包括主键、参照完整性等。
- 作图。
- 最后检查有没有漏掉信息。

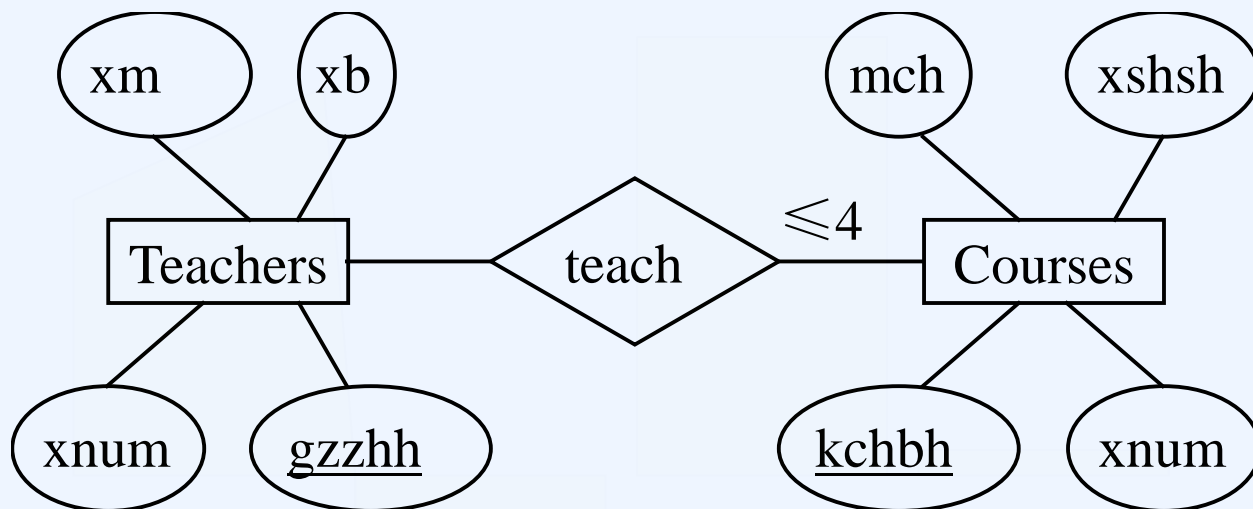


例 课程数据库

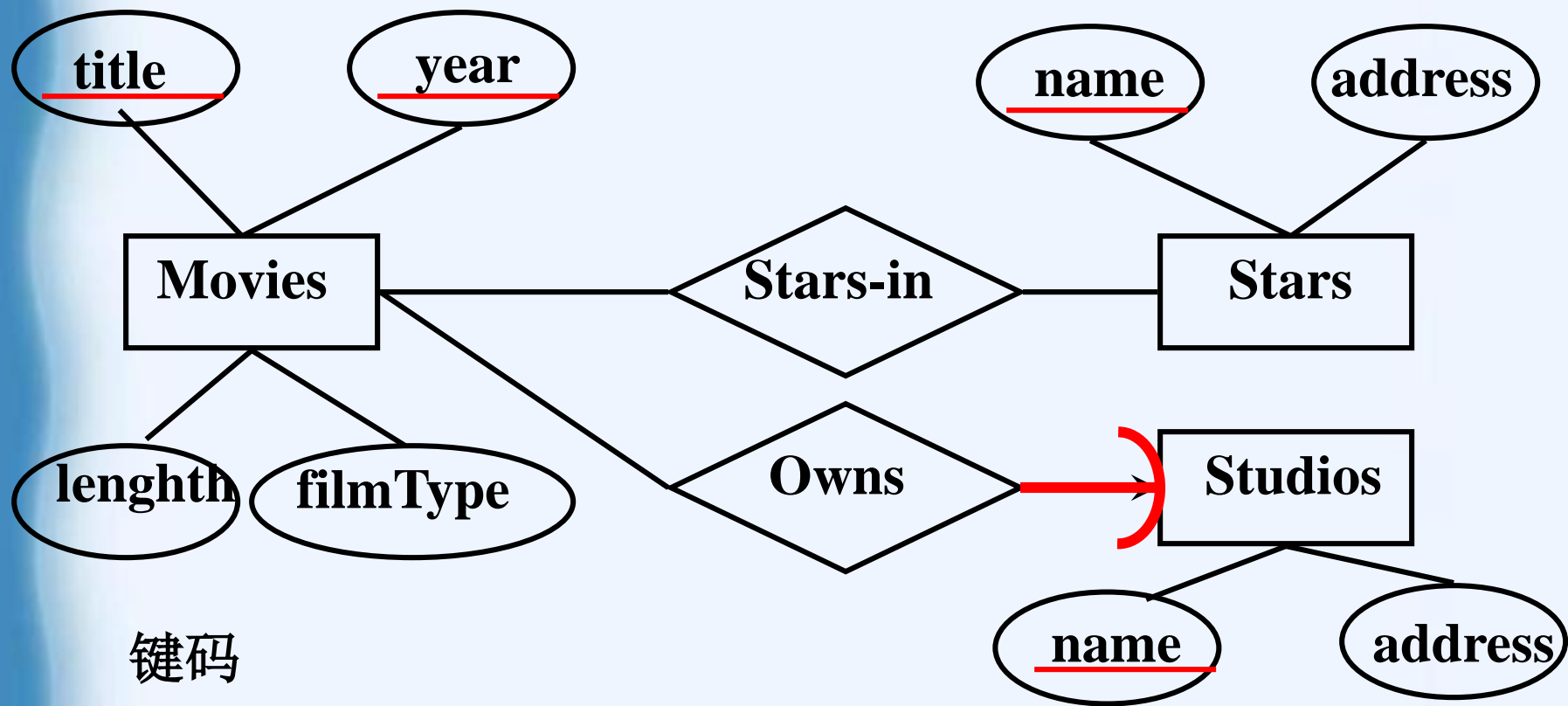
为学校建立课程数据库，要求如下：

- 记录课程的信息：课程名称、课程学时数、课程所属系别和任课教师。
- 记录任课教师信息：姓名、性别、系别、所教课程。
- 每位任课教师最多只能讲授4门课程。

对这个数据库用E/R图进行描述，并且设计键码。



E/R图 例



键码

参照完整性约束

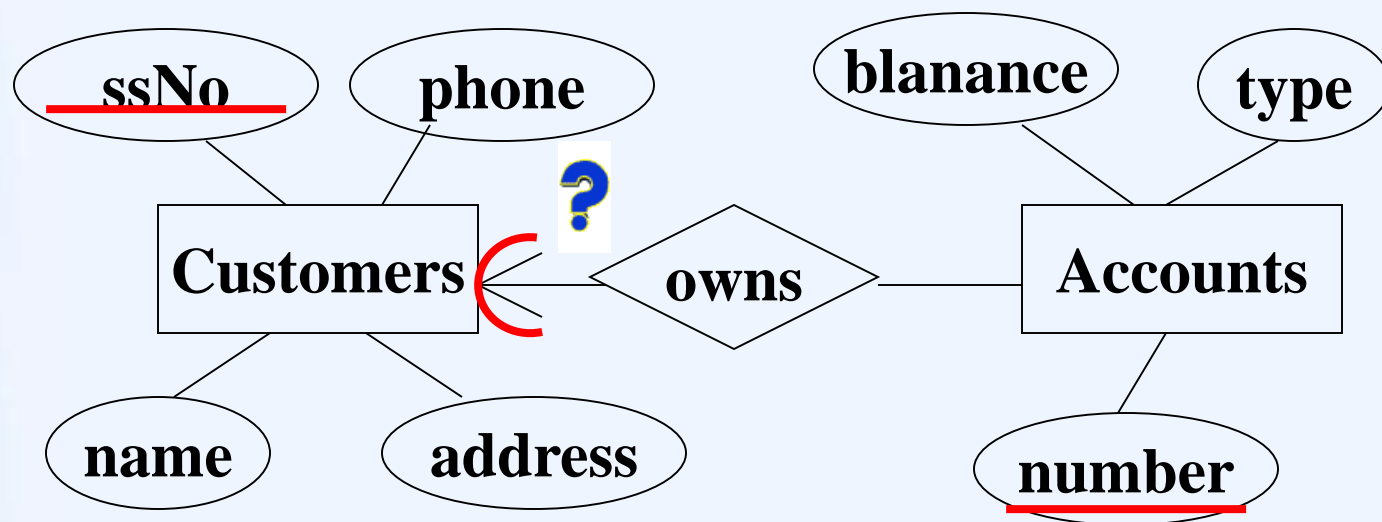


E/R图 例

为银行设计一个数据库，要求记录如下信息：

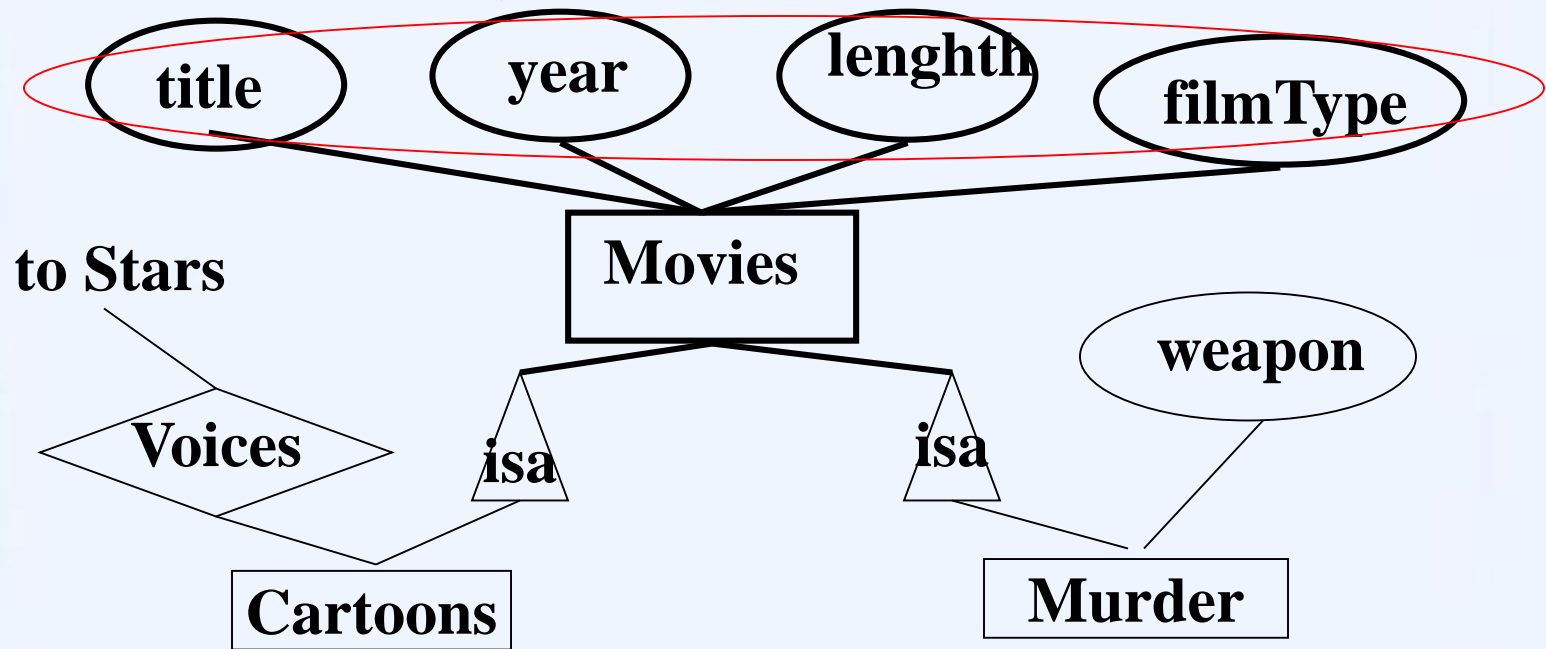
- 顾客的信息包括他们的姓名、地址、电话、社会保险号。
- 帐户包括编号、类型、结余和相应的顾客。

对这个数据库用E/R进行描述，并且设计键码。



E/R图中的子类

- ◆在E/R图中，如果实体集C具有实体集D的所有属性和联系，同时又具有一些其他属性，则可将实体集C描述为D的子类，
- ◆方法是：利用**isa**联系将类C和D的两个实体集相连。**isa**联系用两条直线和一个等腰三角形来表示。三角形的顶点指向超类。专用词“**isa**”放在三角形中。

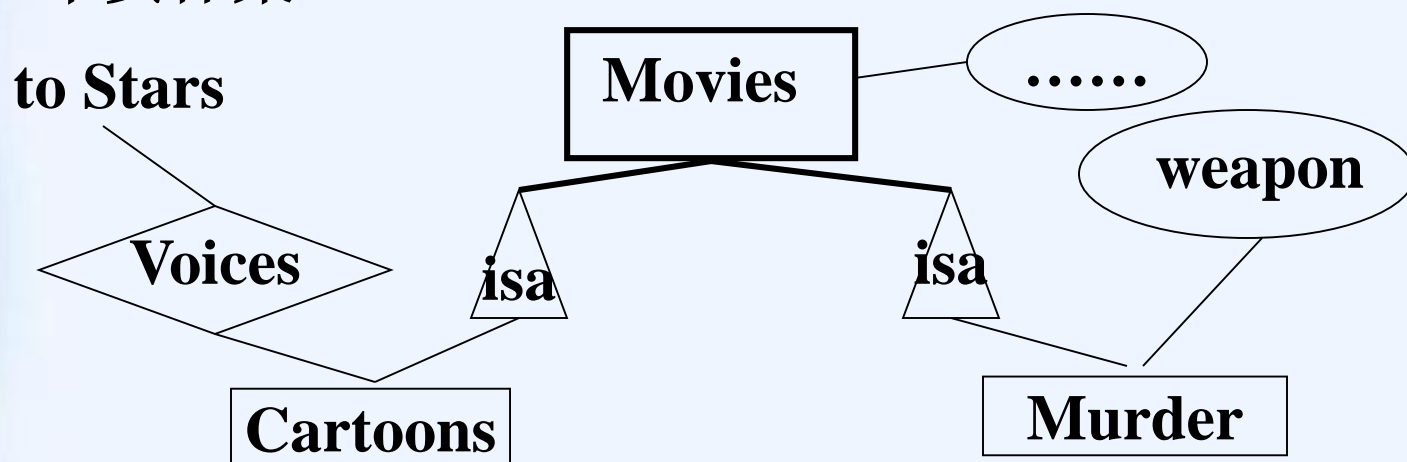


E/R图中的子类

!注：ODL和E/R在子类的概念上是有区别的。

- 在ODL中，对象必须是一个类的成员。
- 在E/R模型中，一个实体可以属于几个实体集，它具有这几个实体集所有的属性。

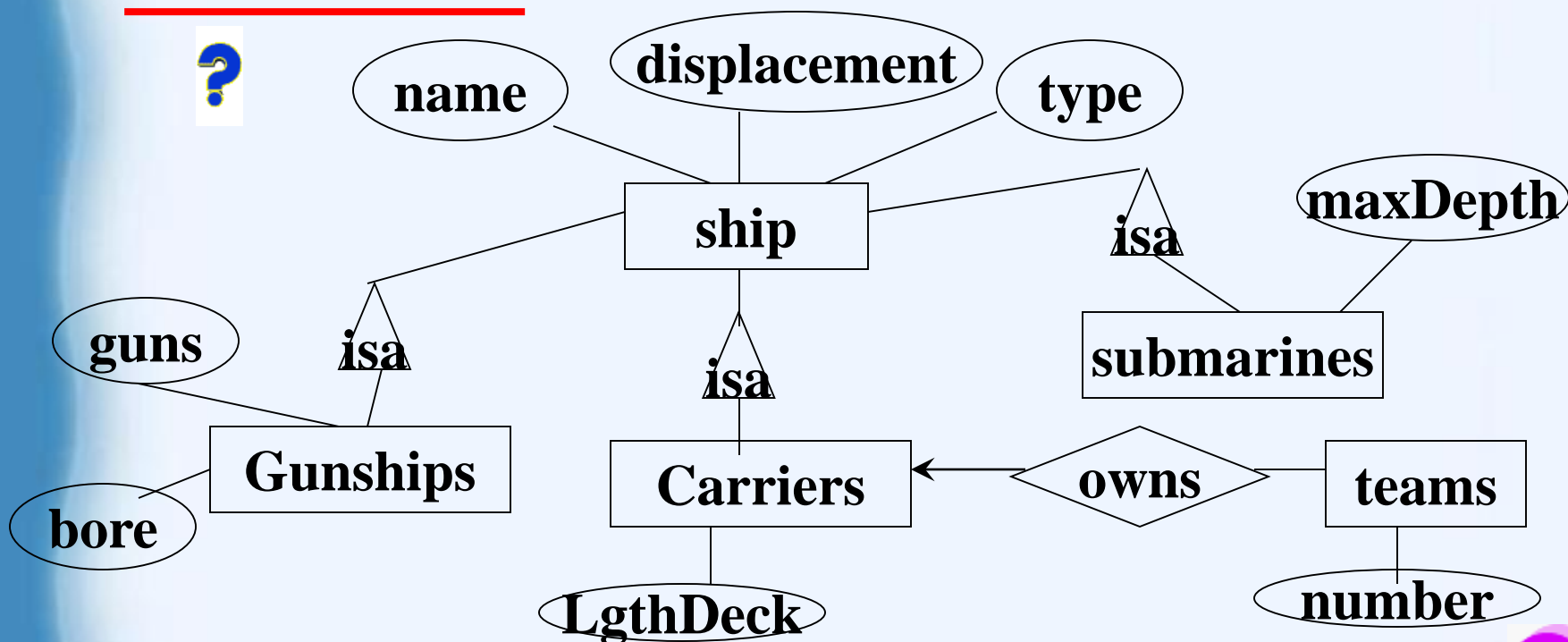
例如：如果要记录一类影片，既是卡通片，又是谋杀片，而且没有自己独有的特性，则在E/R图中，不需要单独建立一个实体集。



E/R图中的子类 例

用E/R描述军舰数据库。每艘军舰有信息：名称、排水量、类型。另外，有下列具有某些其他信息的特殊类型的舰艇：

- 炮舰：携带大型火炮的舰艇，记录主炮的数量和口径。
- 航空母舰：记录飞行甲板的长度和航空大队的集合。
- 潜艇：记录最大安全深度。
- 攻击型航空母舰：既是炮舰又是航空母舰。



E/R图中的弱实体集

◆ **定义** 如果组成一个实体集键码的属性中的一些或全部属于另一个实体集，那么这个实体集称为**弱实体集**。

◆ 弱实体集表示方法：

- 用双边矩形表示一个实体集是弱的。
- 用双边菱形表示连接它和提供其键码属性的其他实体集的多对一联系。
- 对为实体集提供键码的属性加下划线。

❓ 弱实体集是否与子类的概念相同？区别在哪里？

- 弱实体集不具有为它提供键码的实体集的所有属性。而子类具有超类的所有属性。



E/R图中的弱实体集

◆ **弱实体集的键码要求** 如果E是一个弱实体集，那么为E提供键码属性的每个实体集F必须通过联系R和E相连，并且

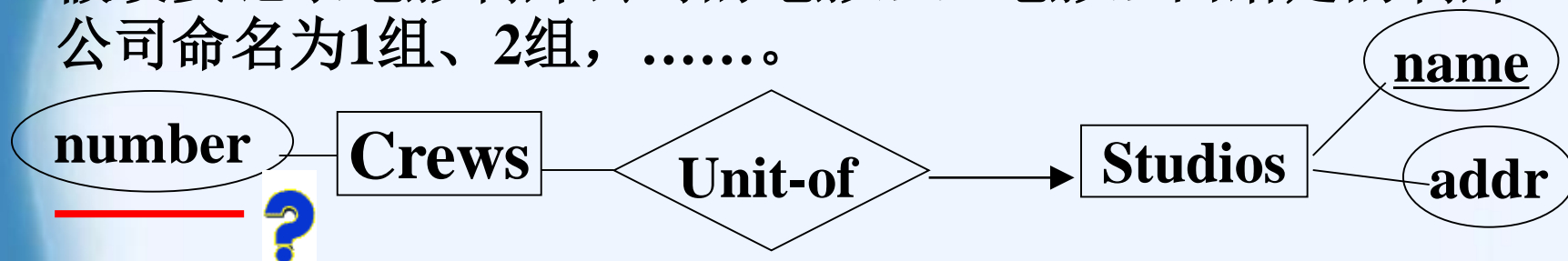
- R必须是从E到F的二元的多对一的联系。
- F为E的键码提供的属性必须是F的键码属性。
- 若实体集F本身是弱实体集，那么F提供给E的键码属性包括为F提供键码属性的实体集的属性。
- 如果有几个从E到F的多对一联系，那么每个联系将提供一次F的键码属性，共同构成E的键码。



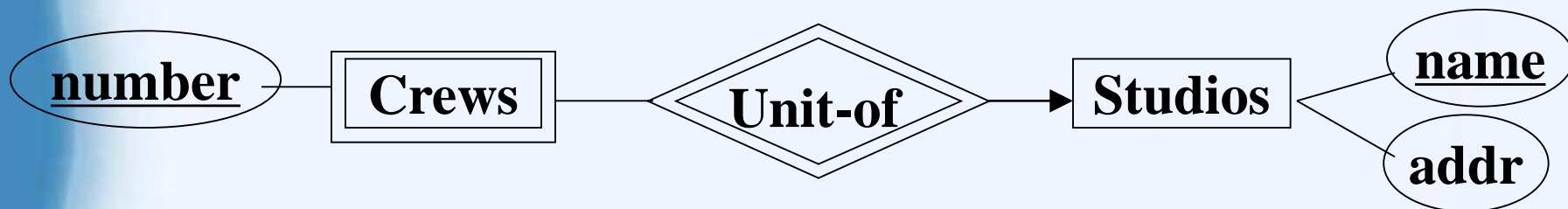
E/R图中的弱实体集 例

◆情况一：实体集属于一种层次结构

假设要记录电影制片公司的电影组，电影组由给定的制片公司命名为1组、2组，……。

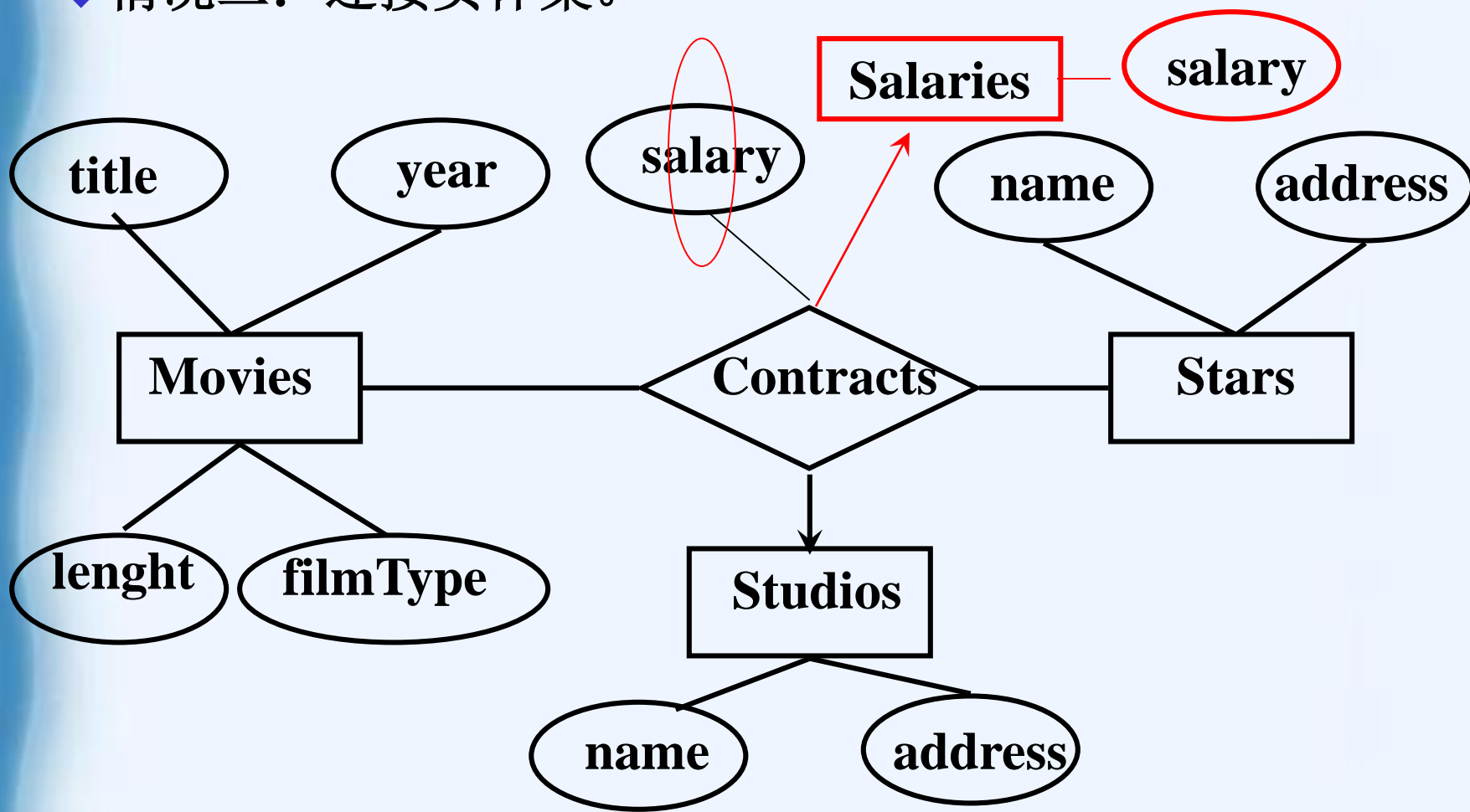


属性number(组号)并不是电影组的键码。为了唯一地标识一个电影组，需要同时给出它所属的制片公司的名字和组号。因此就产生了弱实体集。

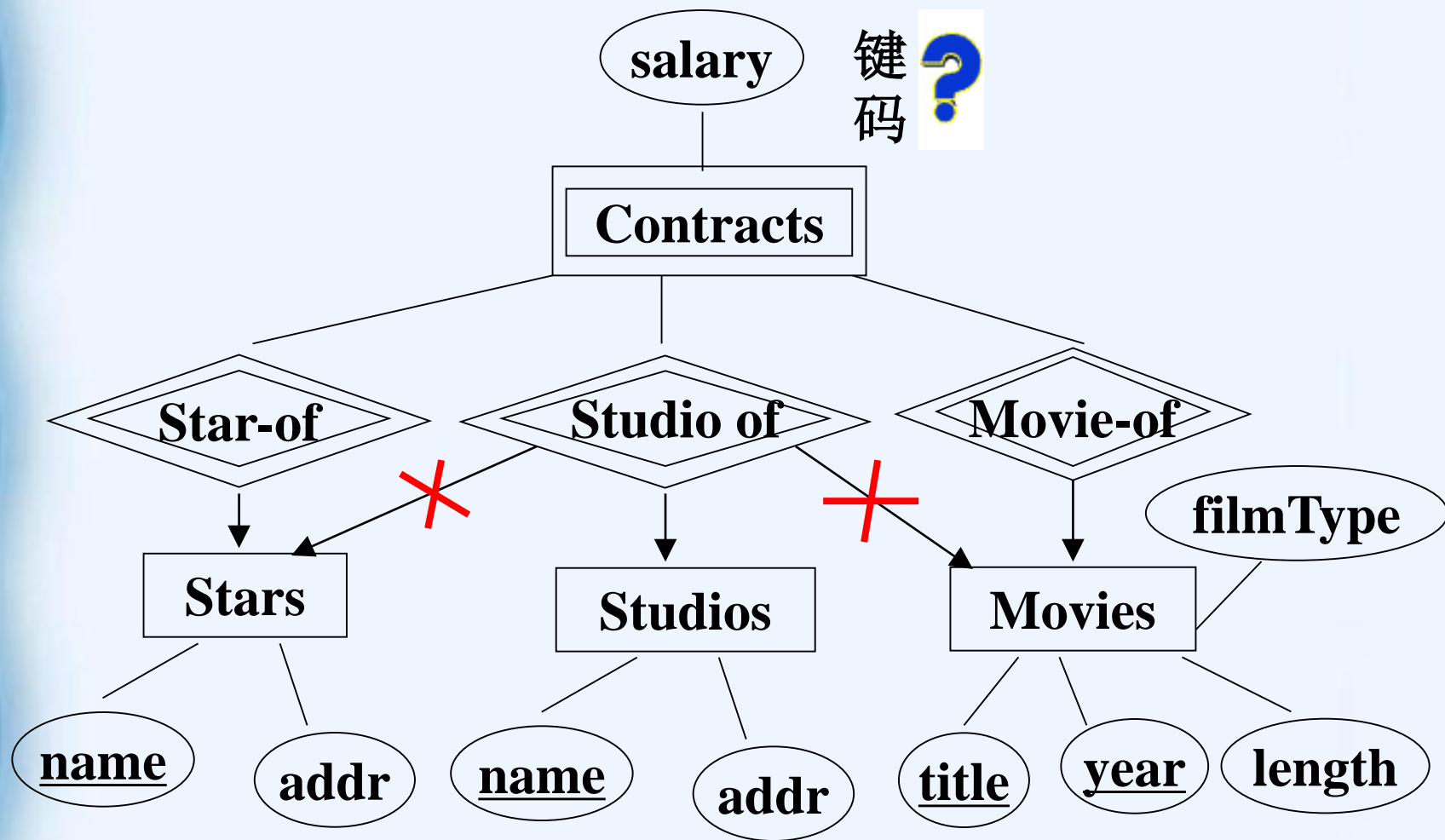


E/R图中的弱实体集 例

◆情况二：连接实体集。



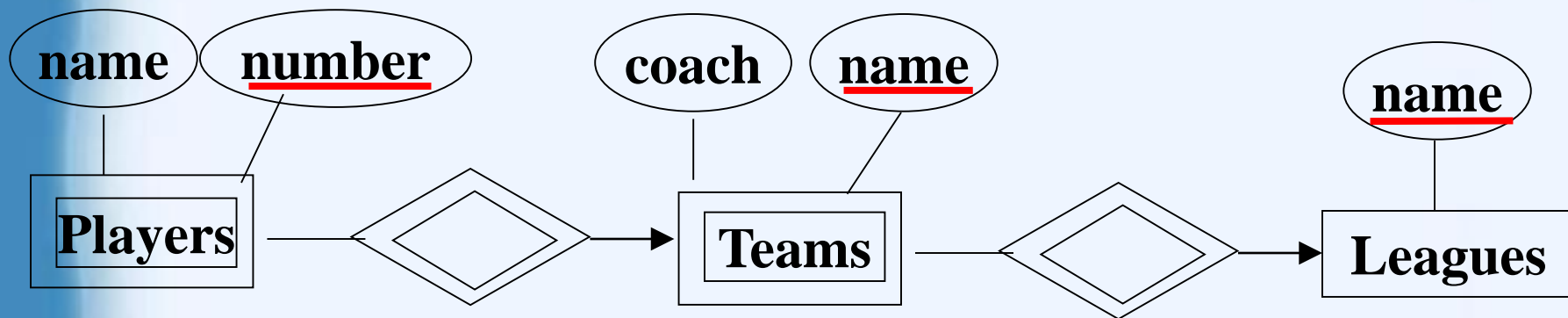
E/R图中的弱实体集



E/R图中的弱实体集 例

画出E/R图并指出实体集的键码，要求记录社团信息(名字)、社团的球队(名字、教练名)和球队的队员信息(名字和队员号)。已知社团的名字是唯一的。一个社团不会有两个同名的球队。一个球队不会有两个同号的队员。

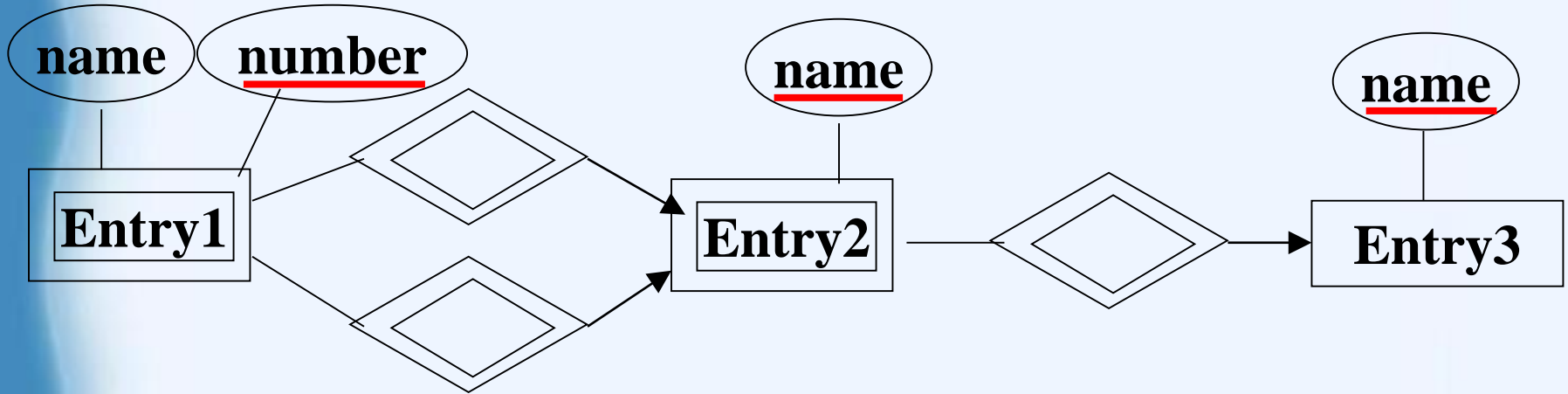
然而，在不同的球队可以有同号的队员，并且在不同的社团可以有同名的球队。



键码 ?



E/R图中的弱实体集 例




Entry1的键码为

number,entry2name1,entry3name1,
entry2name2,entry3name2



E/R图与ODL的区别

E/R模型与ODL的主要区别在于：

- E/R模型中，联系作为独立的概念存在，而ODL中是作为特性嵌套在类定义中。
- ODL中，属性可以是任意的聚集类型。而E/R模型中通常认为数据类型不能是聚集类型。
- E/R模型中，联系可以具有属性，而ODL中没有相应的概念。
- E/R模型中有弱实体集的概念，而ODL中没有。

E/R图转换到关系模型

❖ 对于每个**非弱实体集**，建立一个与之同名而且具有相同属性集的关系。该关系不包含任何联系的信息。

❖ E/R模型中的**联系**转换到关系模型中也用关系表示。对于给定的联系R，它所对应的关系具有以下属性：

- 联系R涉及到的每个实体集的键码属性或属性集。

- 联系R的属性。

如果一个实体集在联系R中出现多次，转换到关系时必须进行改名，以免出现重名属性。

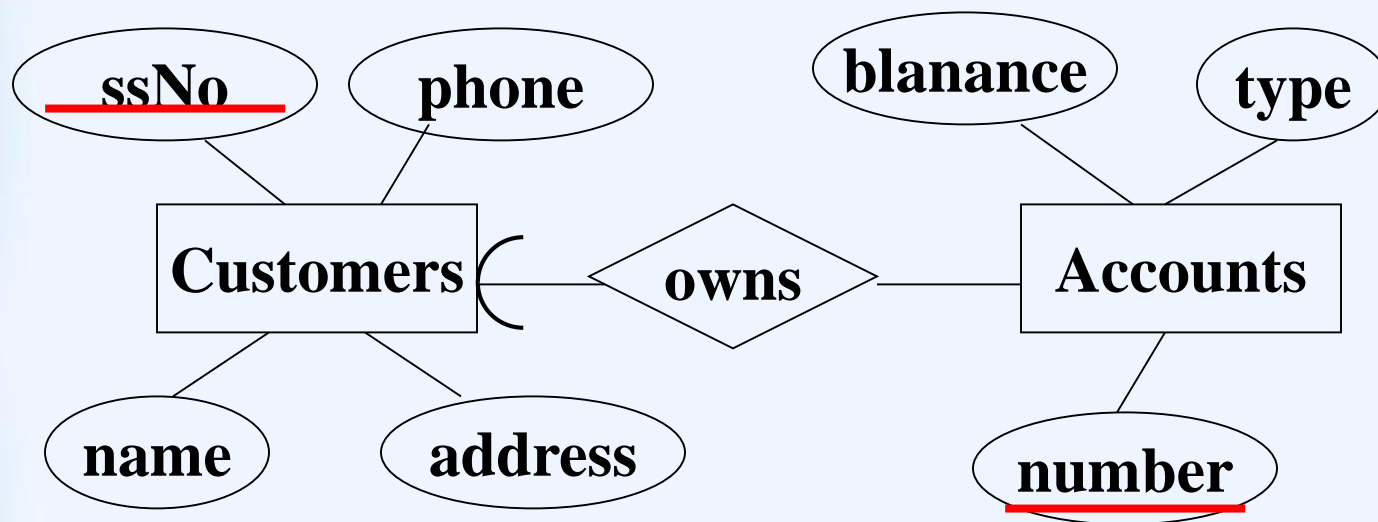


E/R到关系的键码的推断

- ❖ 对于来自实体集的关系，它的键码就是该实体集的键码。
- ❖ 如果关系来自二元联系，有三种情形：
 - 多对多联系 该联系相连的两个实体集的键码都是关系R的键码属性。
 - 从实体集E1到E2的多对一联系 实体集E1的键码是关系的键码，而E2的键码则不是该关系的键码。
 - 一对一联系 联系的任何一个实体集的键码都是该关系的键码，即该关系有两个键码。 两个都画横线
- ❖ 如果关系R来自多向联系，若多向联系R有一个箭头指向实体集E，则转换后的关系中，除了E的键码以外，其他实体集的键码的集合就构成了这个关系的键码。



E/R图转换成关系 例



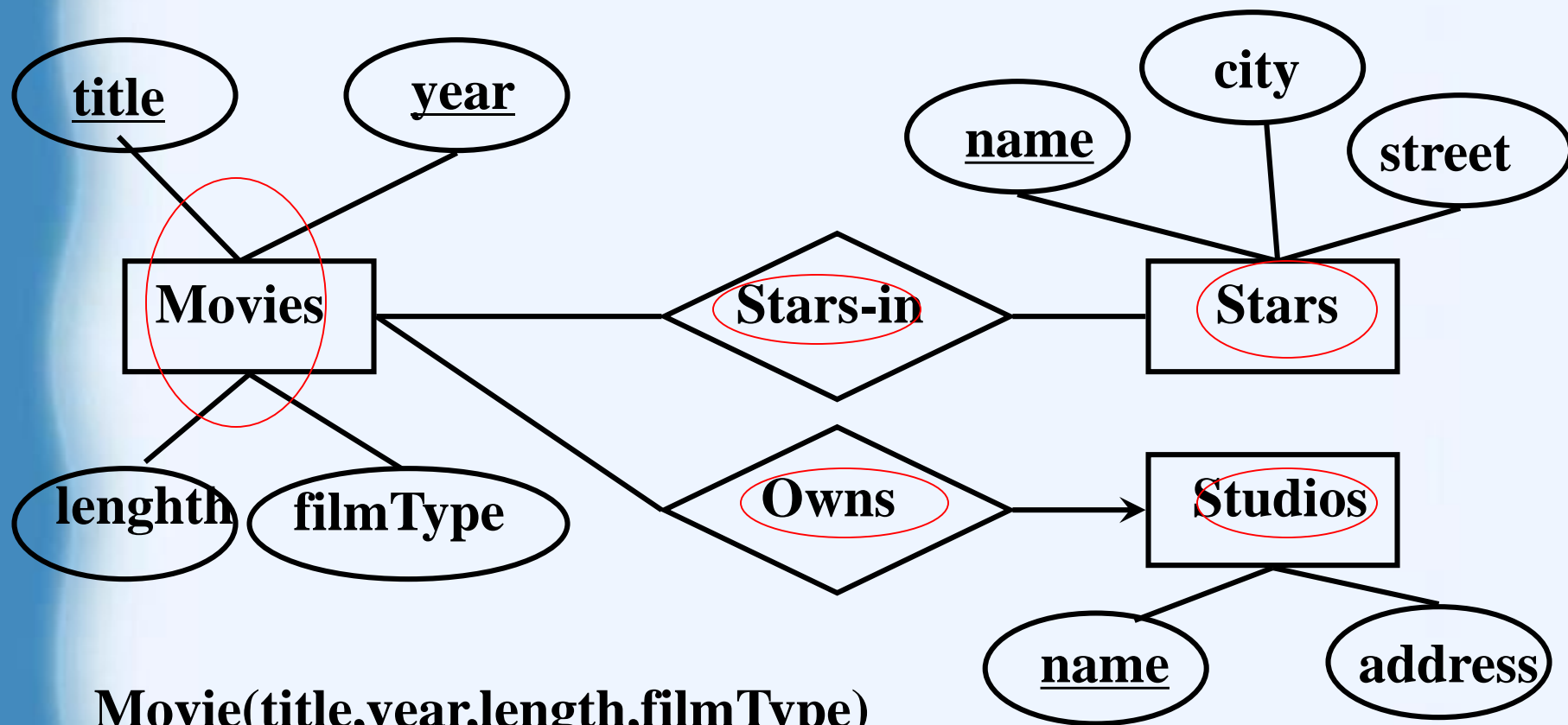
Customers(ssNo,name,address,phone)

Accounts(number,type,balance)

Owns(ssNo,number)



E/R图到关系的转换 例



Movie(title,year,length,filmType)

Star(name,street,city)

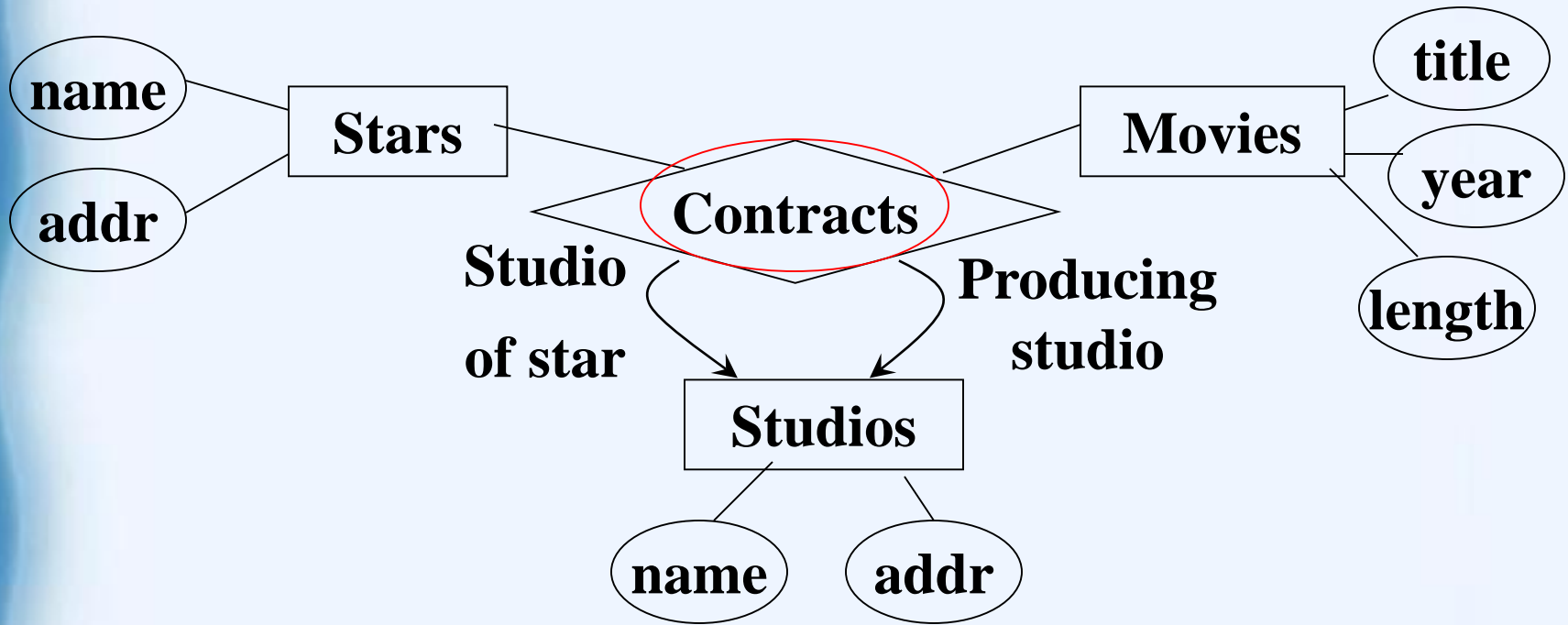
Studio(name,address)

Owns(title,year,studioName)

Stars-in(title,year,starName)



多向联系的转换 例

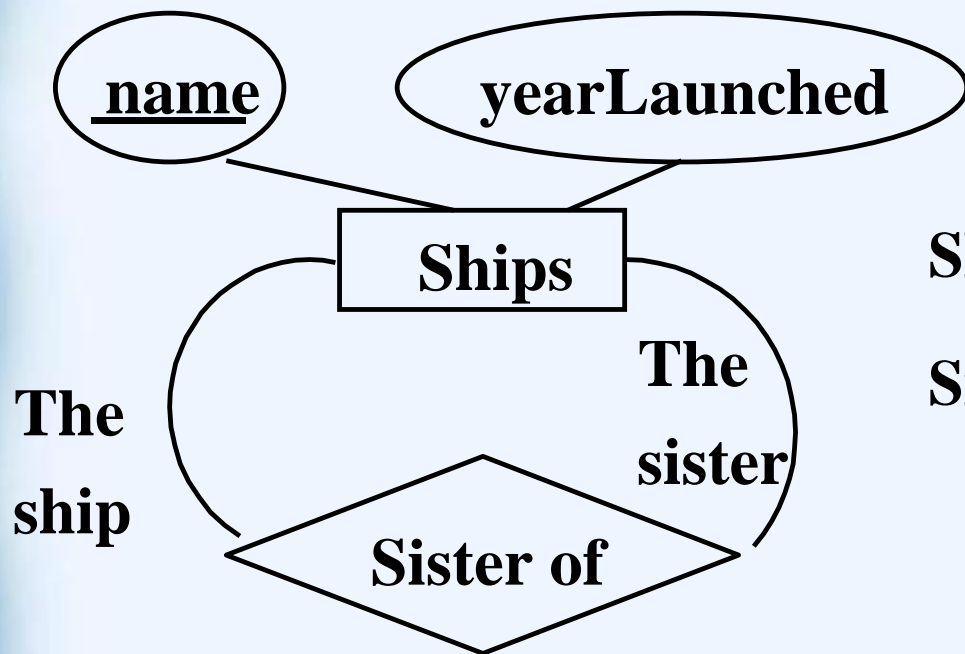


Contracts(title,year,starName,studioOfStar,producingStudio)



E/R图转换成关系 例

下面的E/R图表示舰艇。如果两艘舰艇是根据同一个方案设计制造的，就称它们为“姐妹”舰。把这个E/R图转换为关系数据库模式。



Ships(name,yearLaunched)

SisterOf(name,sisterName)



E/R到关系的转换

❖对E/R图中的**弱实体集**，需要注意：

➤**弱实体集W**转换成关系，该关系既包含W的属性，也包含构成W的键码的其他实体集的键码属性。

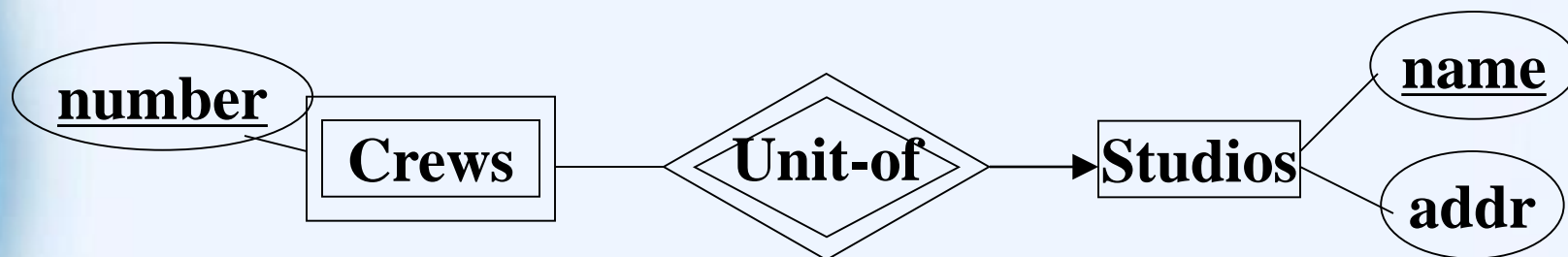
➤与弱实体集W相连的**非双菱形联系**转换成关系，该关系必须包含W的所有键码属性。

➤与弱实体集W相连的**双菱形联系**，不需要转换成关系。

此过程中，必须注意属性不能重名。



弱实体集的处理 例



从该图直观上可以得到三个关系，关系模式分别为：

Studios(name,addr)

Crews(number,studioName)

Unit-of(number,StudioName,name)

? 相同否

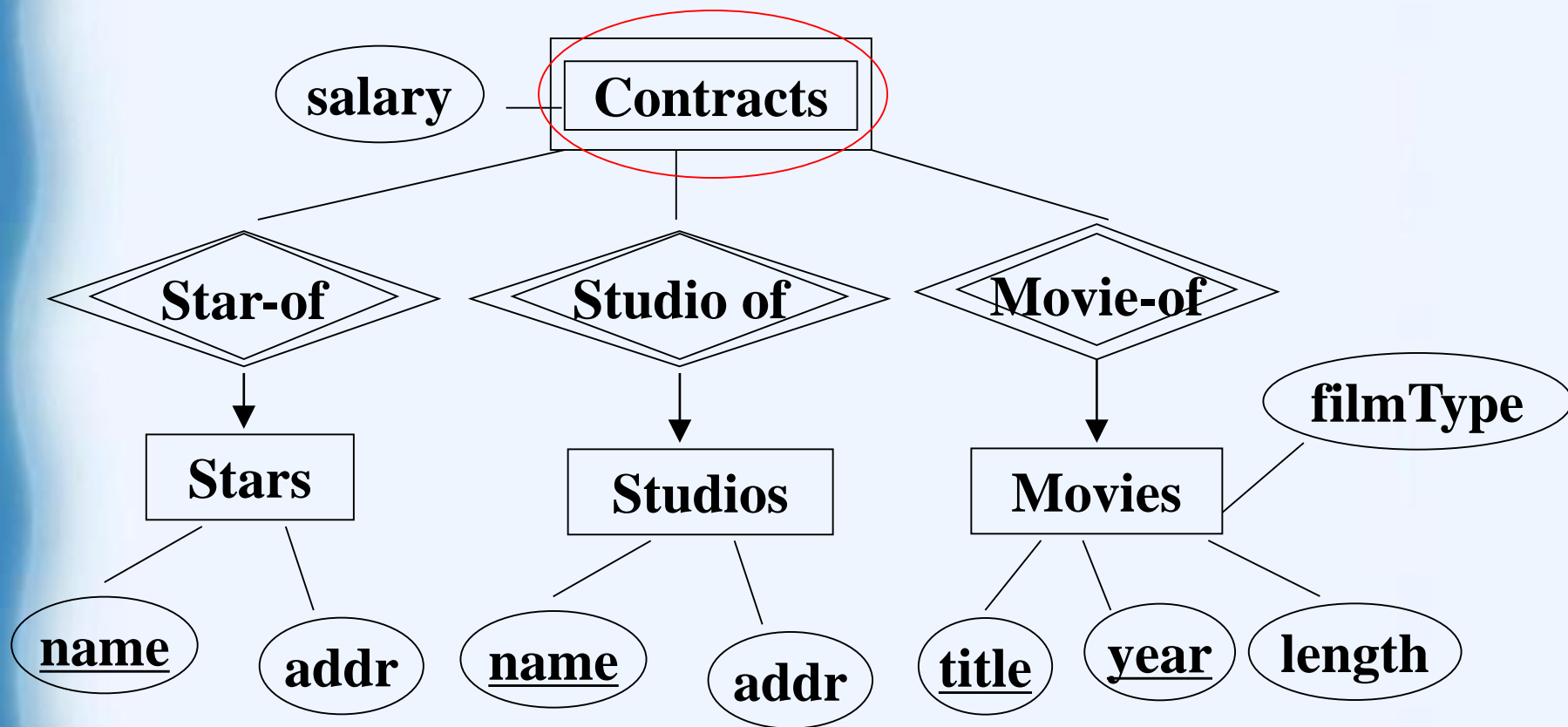
Unit-of关系可以省略。因此最后的转换结果是两个关系：

Studios(name,addr)

Crews(number,studioName)



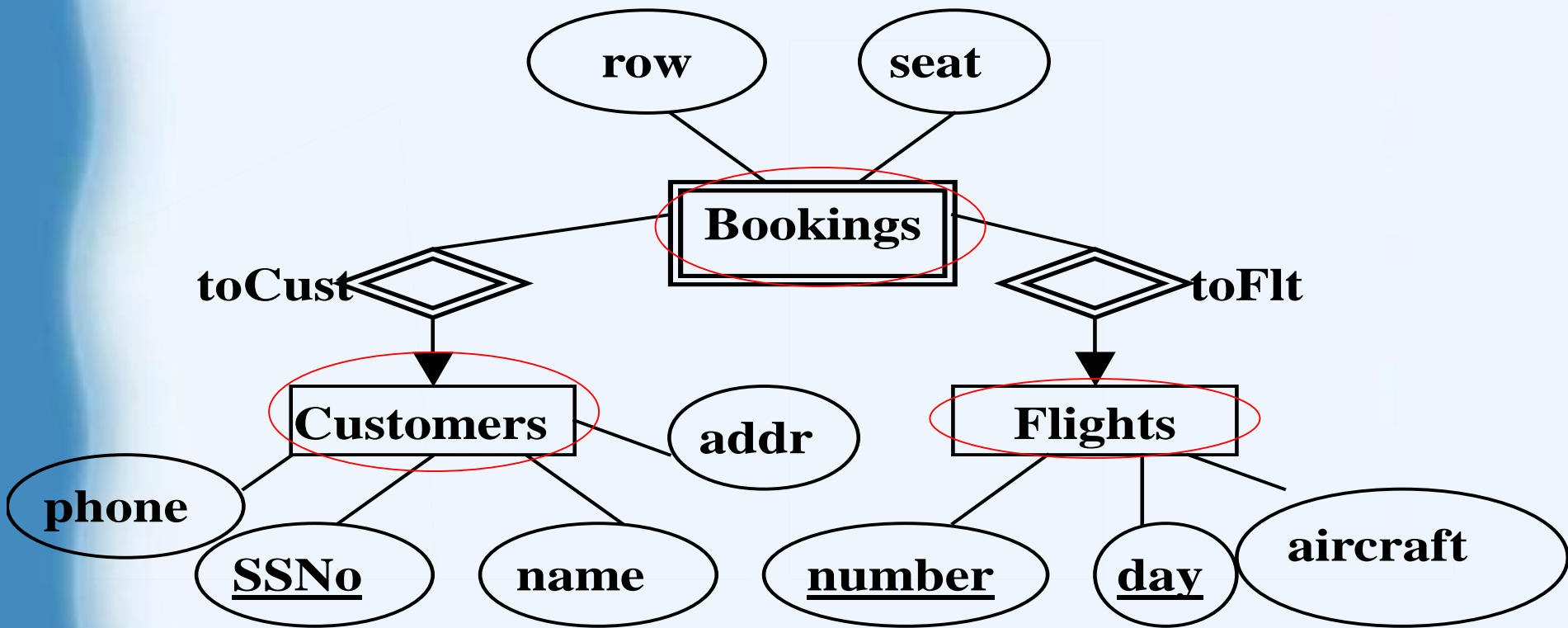
弱实体集的处理 例



弱实体集Contracts(签约)对应的关系Contracts的关系模式为：
Contracts(starName, studioName, title, year, salary)



弱实体集的处理 例



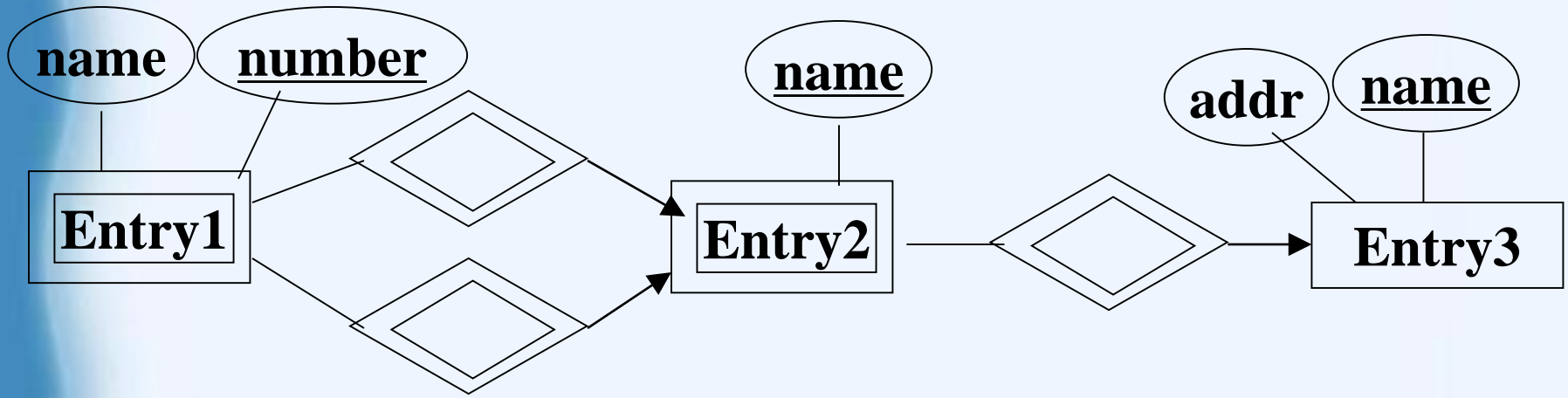
Customers(ssNo,name,address,phone)

Flights(number,day,aircraft)

Bookings(ssNo,number,day,row,seat)



弱实体集的处理 例



Entry3(name,addr)

Entry2(Entry3name,Entry2name,addr) ?

Entry1(Entry1name,number,entry2name1,entry3name1,
entry2name2,entry3name2)



“属于”联系到关系的转换

❖ E/R中isa联系转换到关系时

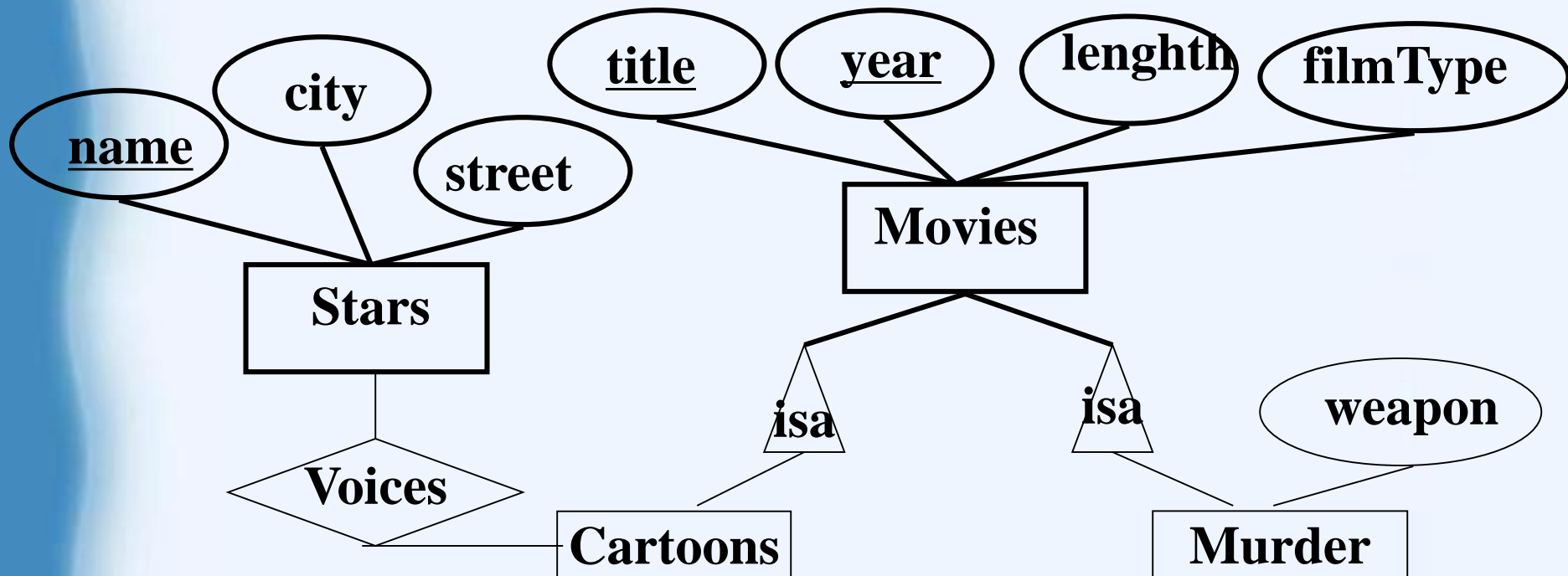
➤ 属于联系本身不需要建立相应的关系。

➤ 对于子类，除了包含自己的属性集外，还应包含超类的键码。不用包含超类的所有属性，只要键码即可。

➤ 如果在元组中允许使用NULL值，就可以用单个关系来表示E/R图中的继承。



继承的转换 例



Movies(title,year,length,filmType)

Muder(title,year,weapon)

Cartoons(title,year)

Voices(title,year,name)

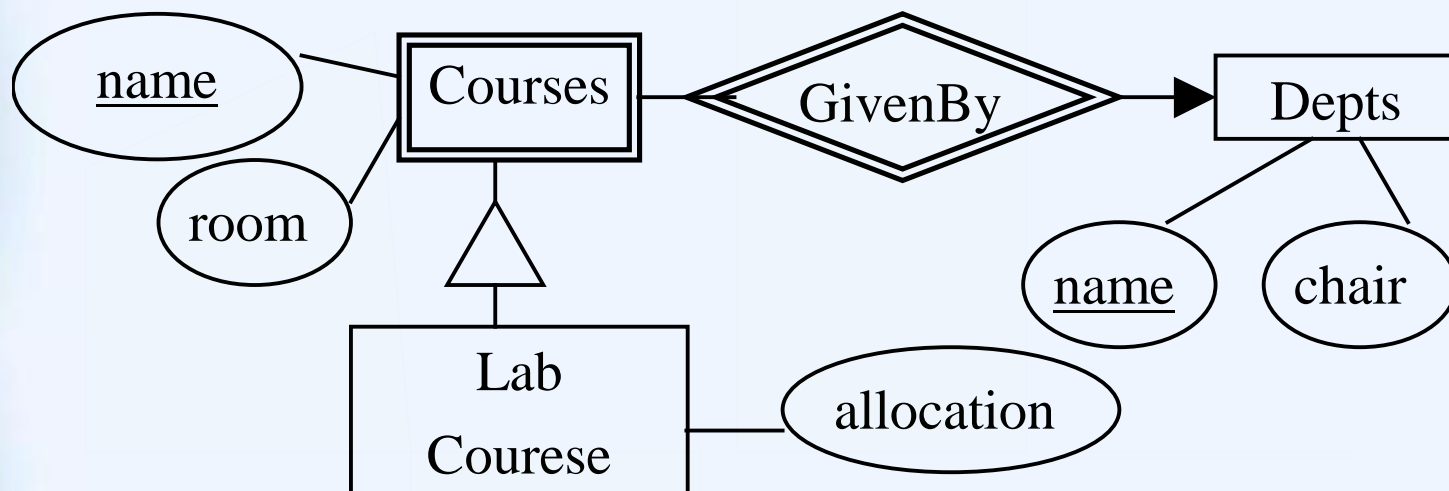
利用NULL合并

→

Movie(title,year,length,
filmType,weapon)



E/R图转换成关系 例



直观上得到: **Depts(name,chair)**

Courses(name,deptName,room)

LabCourses(name,deptName,~~room~~,allocation)



利用NULL合并**Courses**和**LabCourses**得到

Depts(name,chair)

Courses(name,deptName,room,allocation)