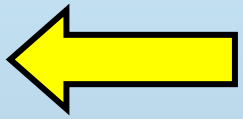
第7章 数据库设计

金培权（jpq@ustc.edu.cn）

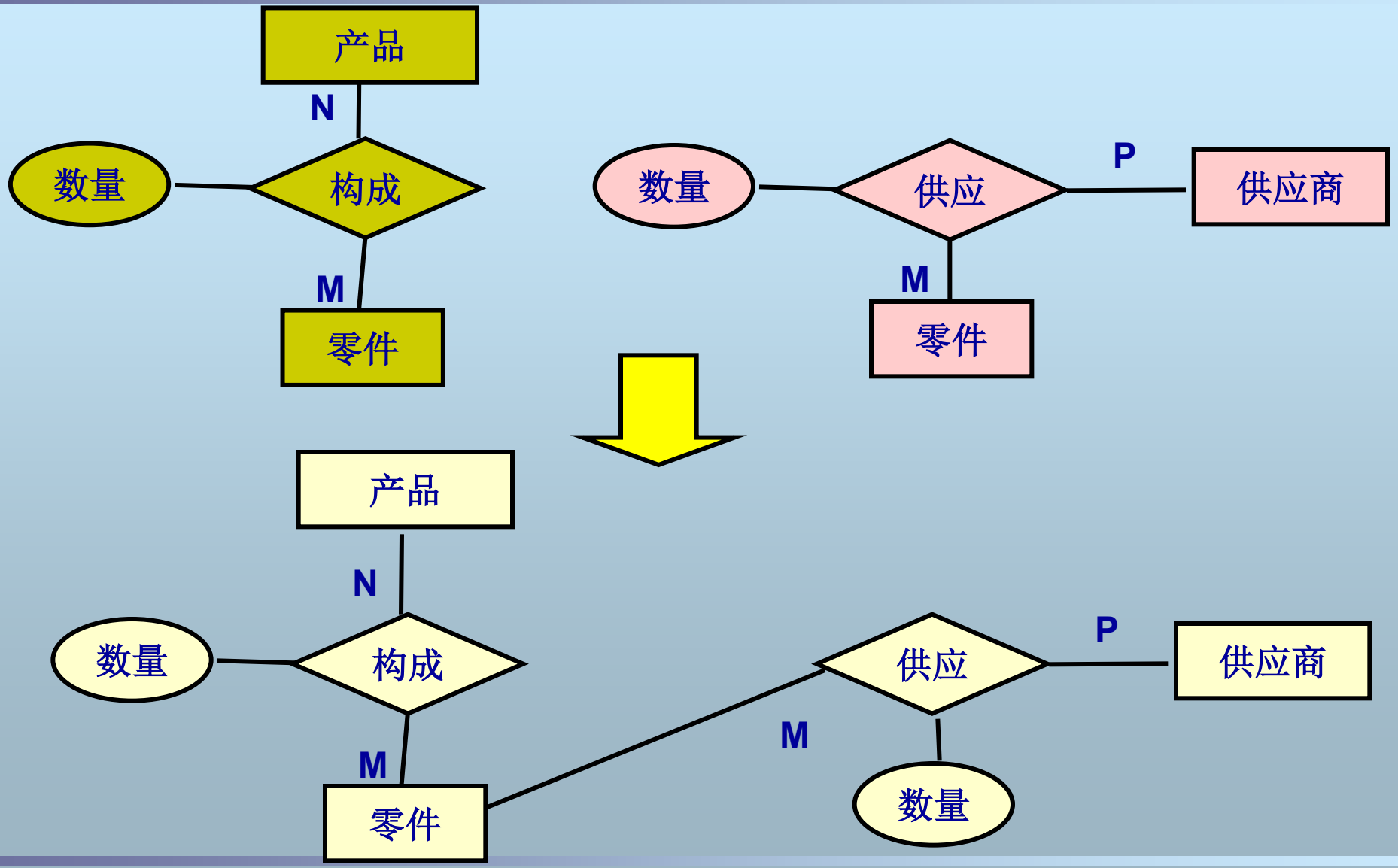
2、ER设计的步骤

- 自顶向下进行需求分析，自底向上进行ER设计
  - 分ER模型设计（局部ER图）
  - ER模型集成 
  - ER模型优化

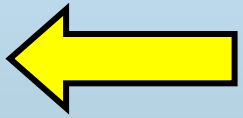
ER集成

- 确定公共实体
- 合并分ER图
- 消除冲突
  - 属性冲突：类型冲突、值冲突
    - 例如性别、年龄
  - 结构冲突：实体属性集不同、联系类型不同、同一对象在不同应用中的抽象不同
  - 命名冲突：同名异义、异名同义
    - 实体命名冲突、属性命名冲突、联系命名冲突

ER集成示例



2、ER设计的步骤

- 自顶向下进行需求分析，自底向上进行ER设计
  - 分ER模型设计（局部ER图）
  - ER模型集成
  - ER模型优化 

ER模型的优化

- 目标
  - 实体个数要少，属性要少，联系尽量无冗余
- 具体优化手段
  - 合并实体类型
  - 消除冗余属性
  - 消除冗余联系

A）合并实体

- 一般1:1联系的两个实体可以合并为一个实体
- 如果两个实体在应用中经常需要同时处理，也可考虑合并
  - 例如病人和病历，如果实际中通常是查看病人时必然要查看病历，可考虑将病历合并到病人实体中
    - 减少了连接查询开销，提高效率

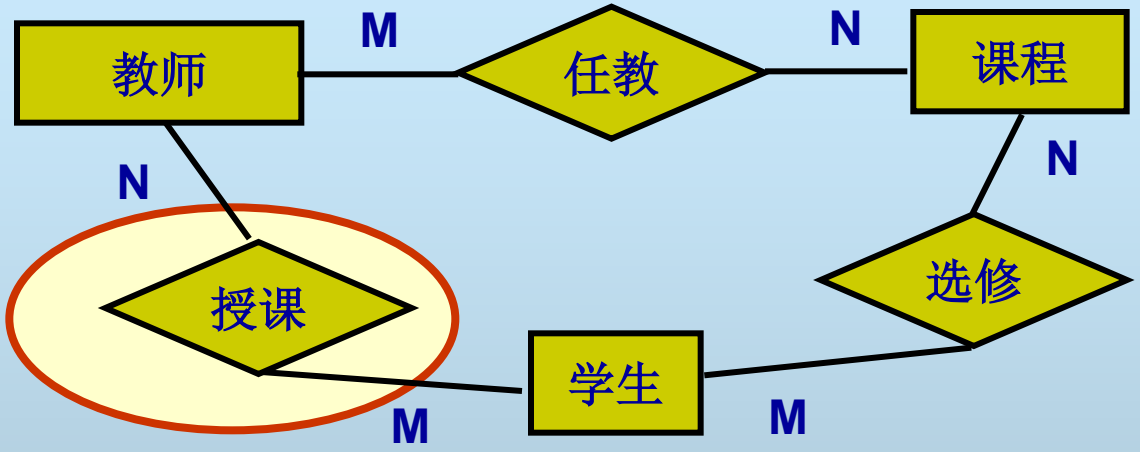
B）消除冗余属性

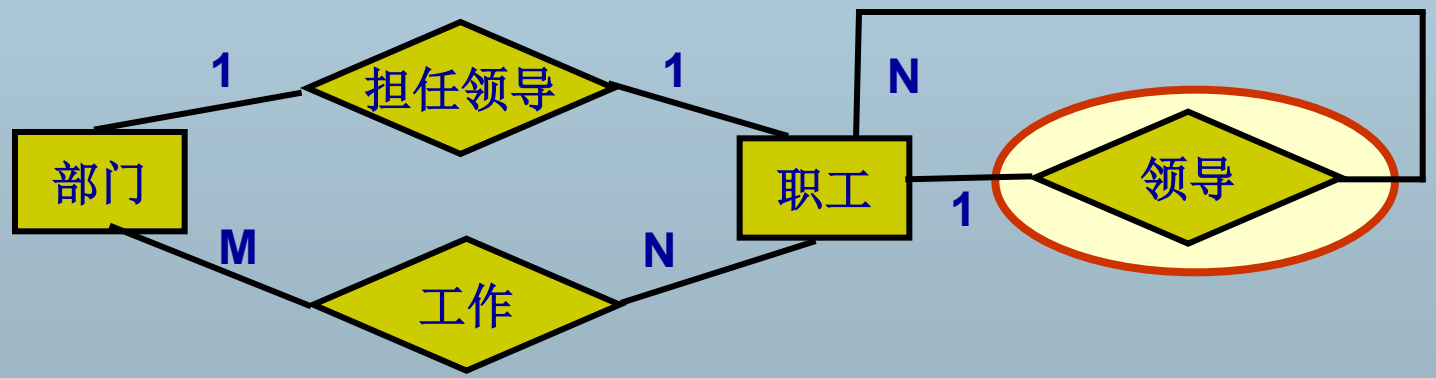
- 分ER图中一般不存在冗余属性，但集成后可能产生冗余属性
  - 例如，教育统计数据库中，一个分ER图中含有高校毕业生数、在校学生数，另一个分ER图中含有招生数、各年级在校学生数
  - 每个分ER图中没有冗余属性，但集成后“在校学生数”冗余，应消除

B）消除冗余属性

- 冗余属性的几种情形
  - 同一非码属性出现在几个实体中
  - 一个属性值可从其它属性值中导出
    - 例如出生日期和年龄

C）消除冗余联系

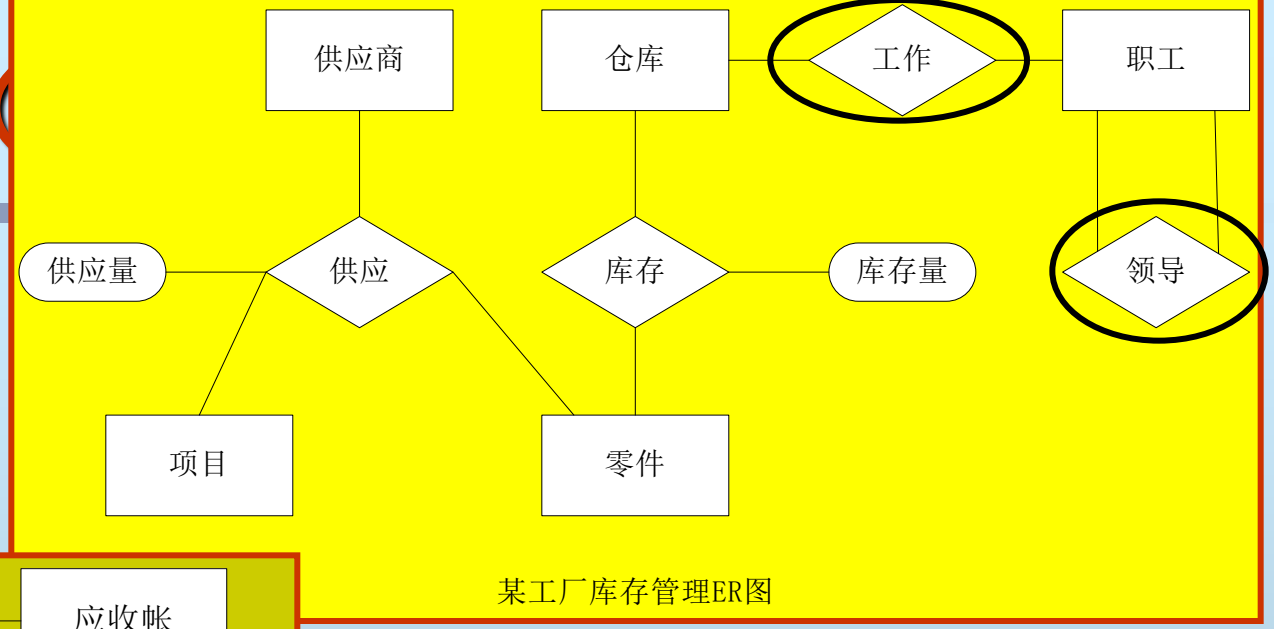




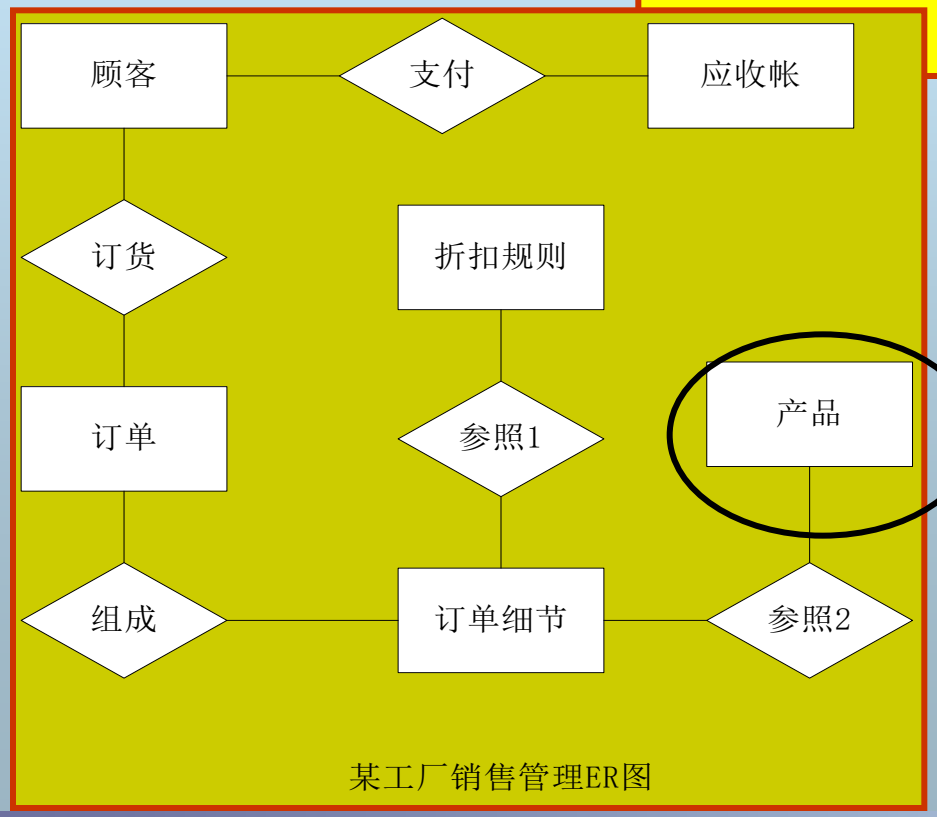
项目＝产品

职工与仓库的联系冗余

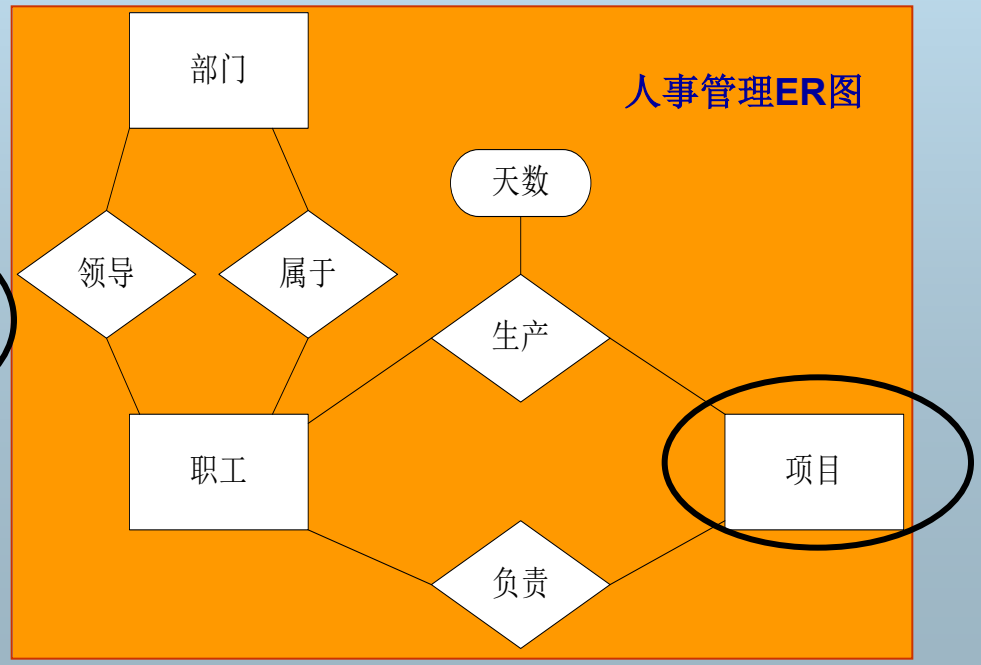
职工与职工的领导联系冗余

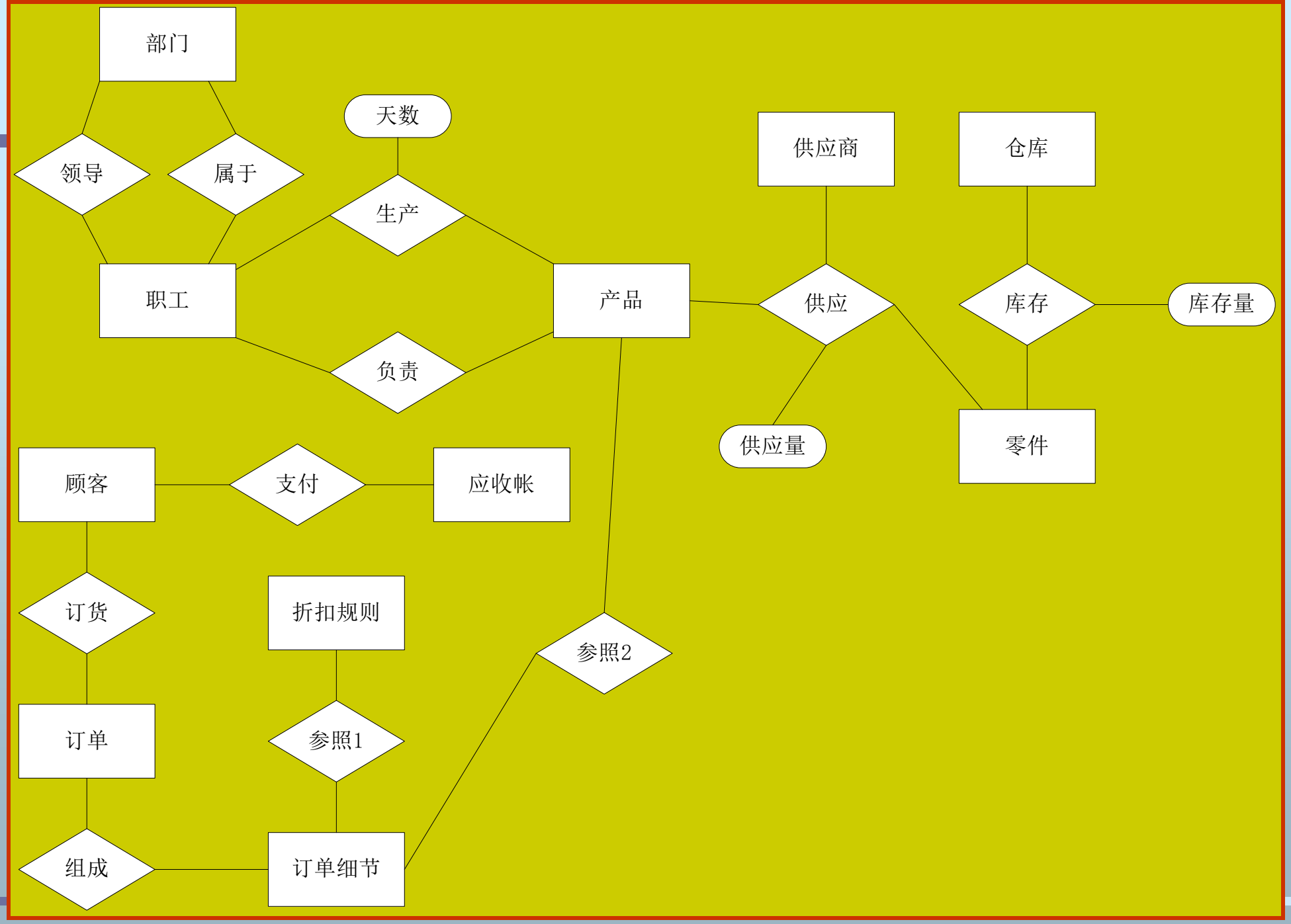


某工厂库存管理ER图



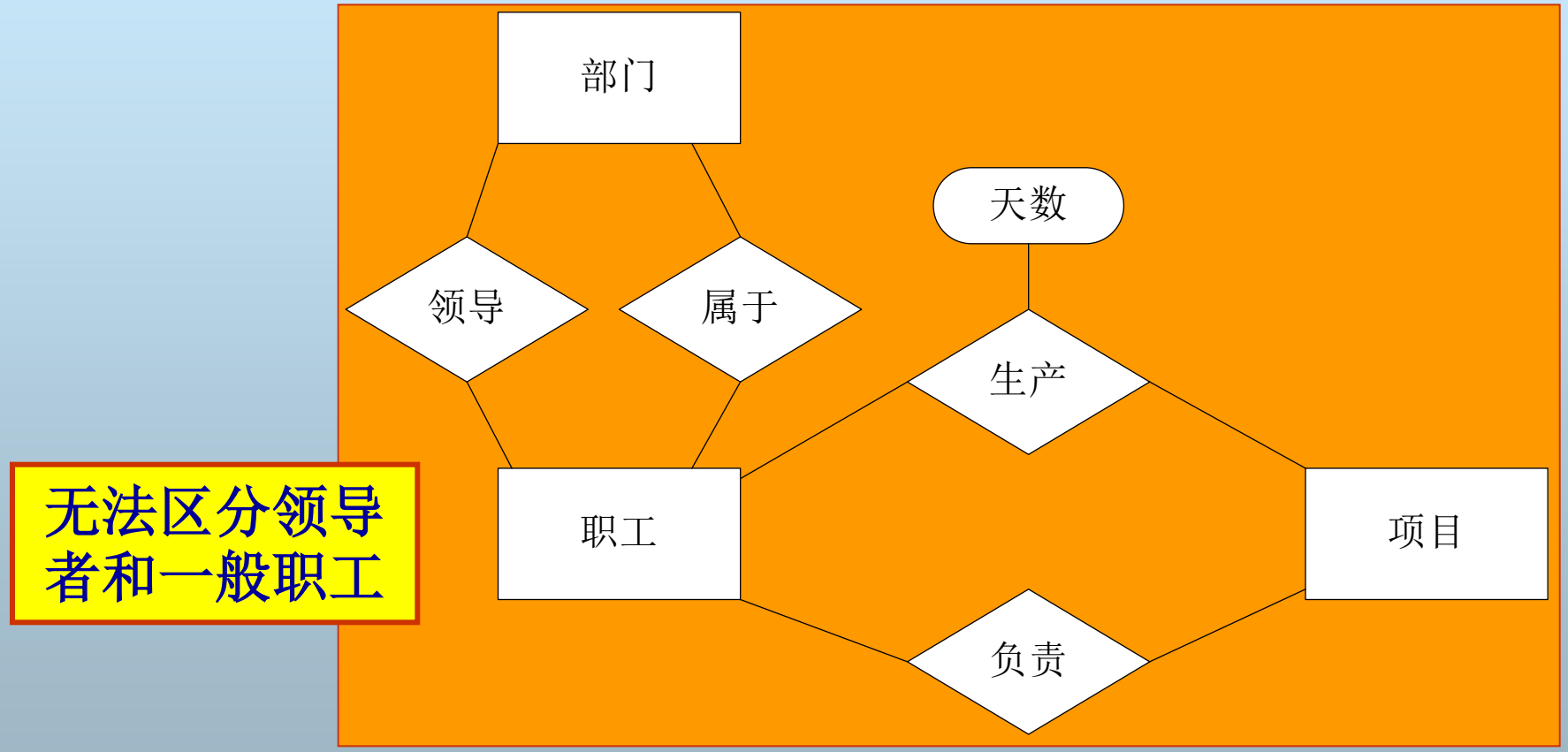
某工厂销售管理ER图




部门
领导
属于
天数
生产
职工
产品
负责
供应商
仓库
供应
库存
库存量
供应量
零件
顾客
支付
应收帐
订货
订单
组成
折扣规则
参照1
参照2
订单细节

3、ER模型的扩展

- 传统的ER模型无法表达一些特殊的语义



3、ER模型的扩展

- 弱实体
- 子类（特殊化）与超类（一般化）

（1）弱实体（weak entity）

- 一个弱实体的存在必须以另一实体的存在为前提
  - 弱实体所依赖存在的实体称为常规实体（regular entity）或强实体（strong entity）
  - 弱实体有自己的标识，但它的标识只保证对于所依赖的强实体而言是唯一的。在整个系统中没有自己唯一的实体标识

（1）弱实体（weak entity）

- 弱实体的例子
  - 一个公司的人事系统中，需要管理职工和职工的子女信息
  - 子女是弱实体，职工是强实体
  - 是否弱实体要看具体应用：例如在社区人口管理系统中，子女就不是弱实体，即使双亲都不存在了，子女仍应存在于人口系统中

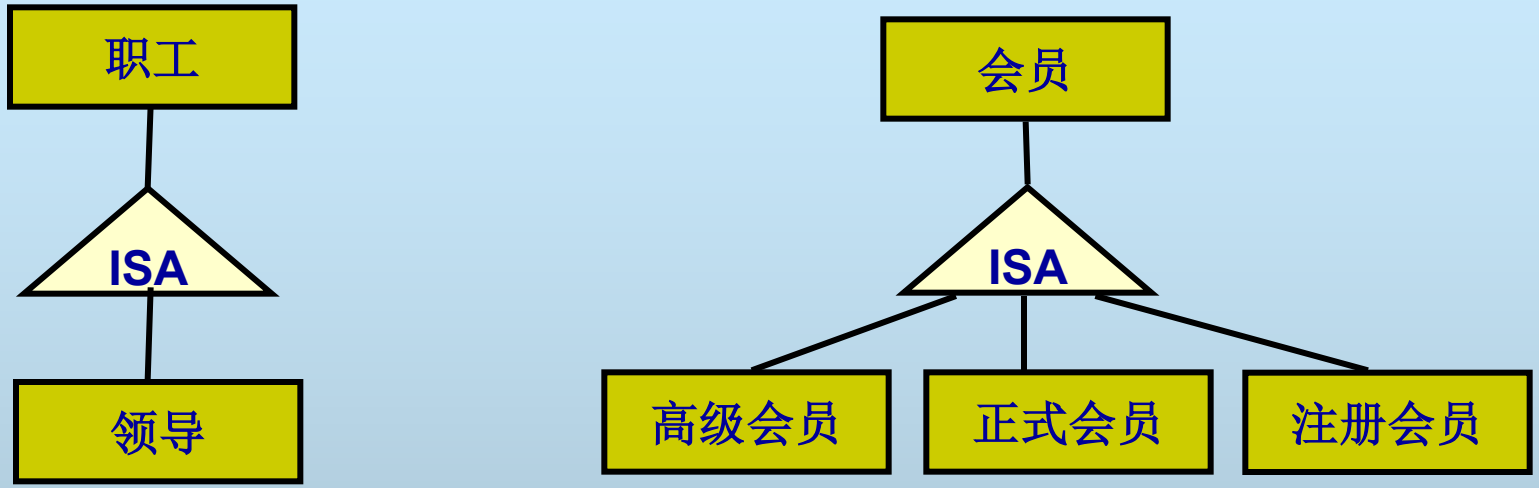
（2）弱实体的表示

弱实体用双线矩形表示，存在依赖联系用双线菱形表示，箭头指向强实体

（3）子类（特殊化）与超类（一般化）

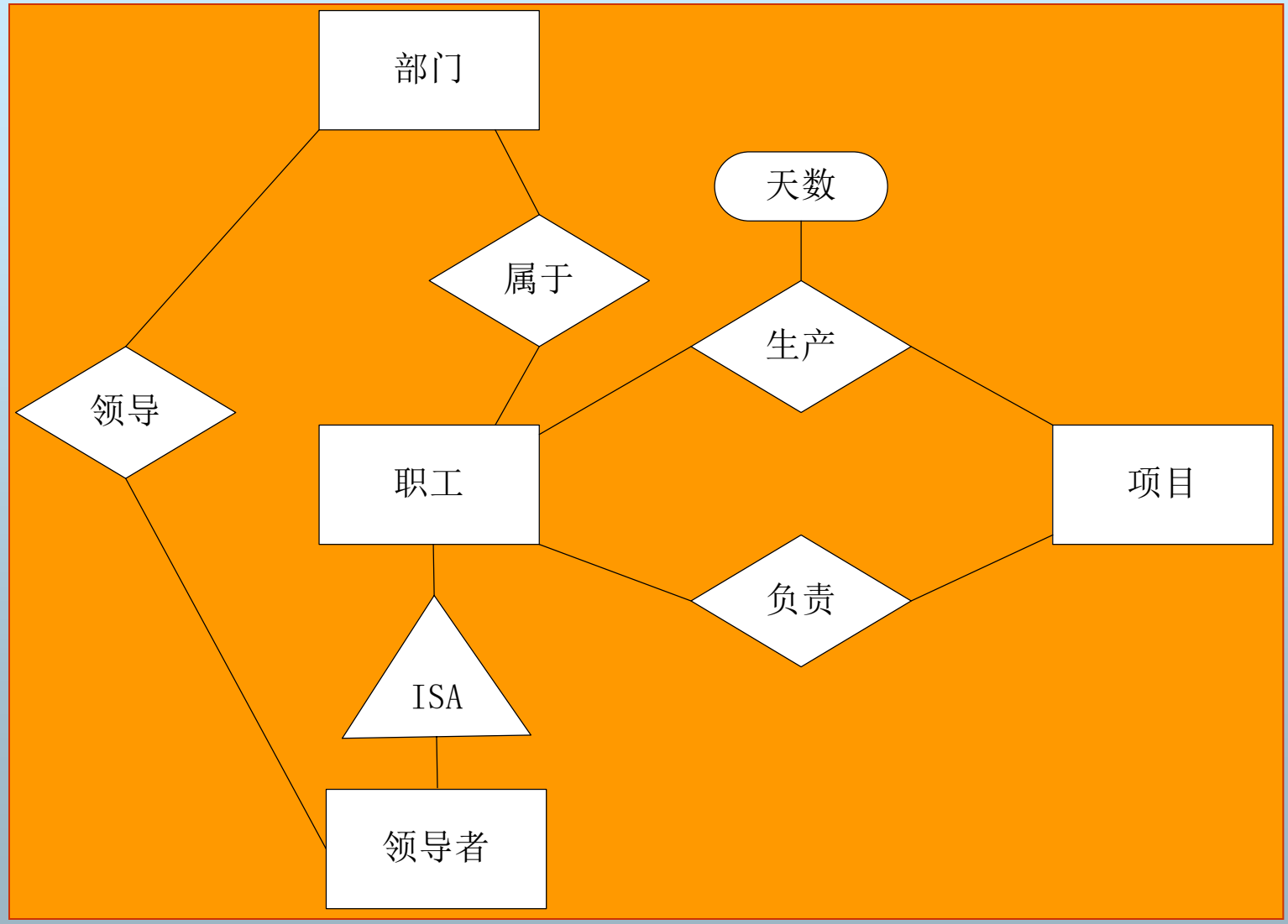
- 子类（Subtype）和超类（Supertype）
  - 两个实体A和B并不相同，但实体A属于实体B，则A称为实体子类，B称为实体超类
  - 子类是超类的特殊化，超类是子类的一般化
  - 子类继承了超类的全部属性，因此子类的标识就是超类的标识
  - 例如，研究生是学生的子类，经理是职工的子类
- 在ER设计时，可以根据实际情况增加子类，也可以根据若干实体抽象出超类

（4）子类符号

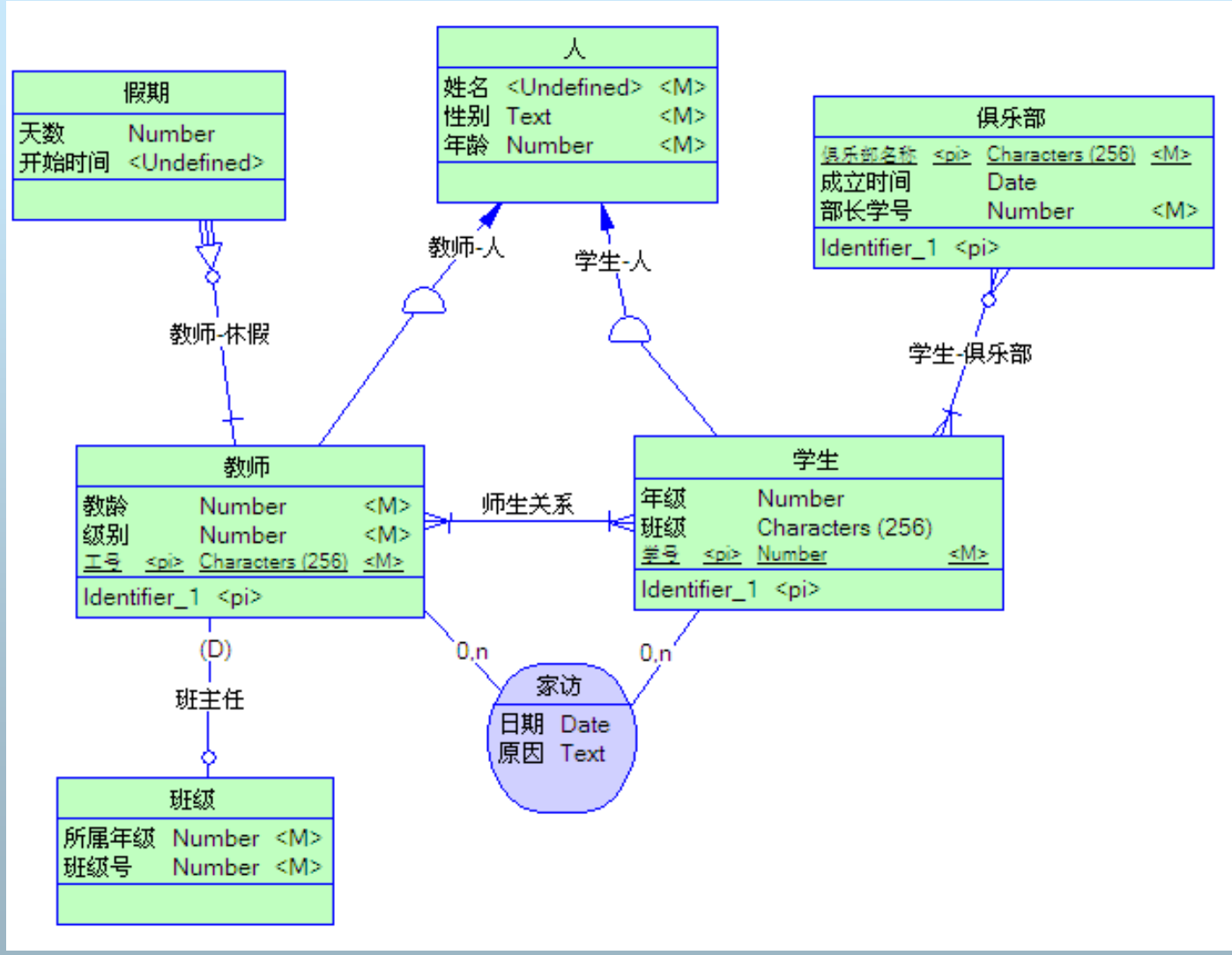


ISA表示子类与超类关系

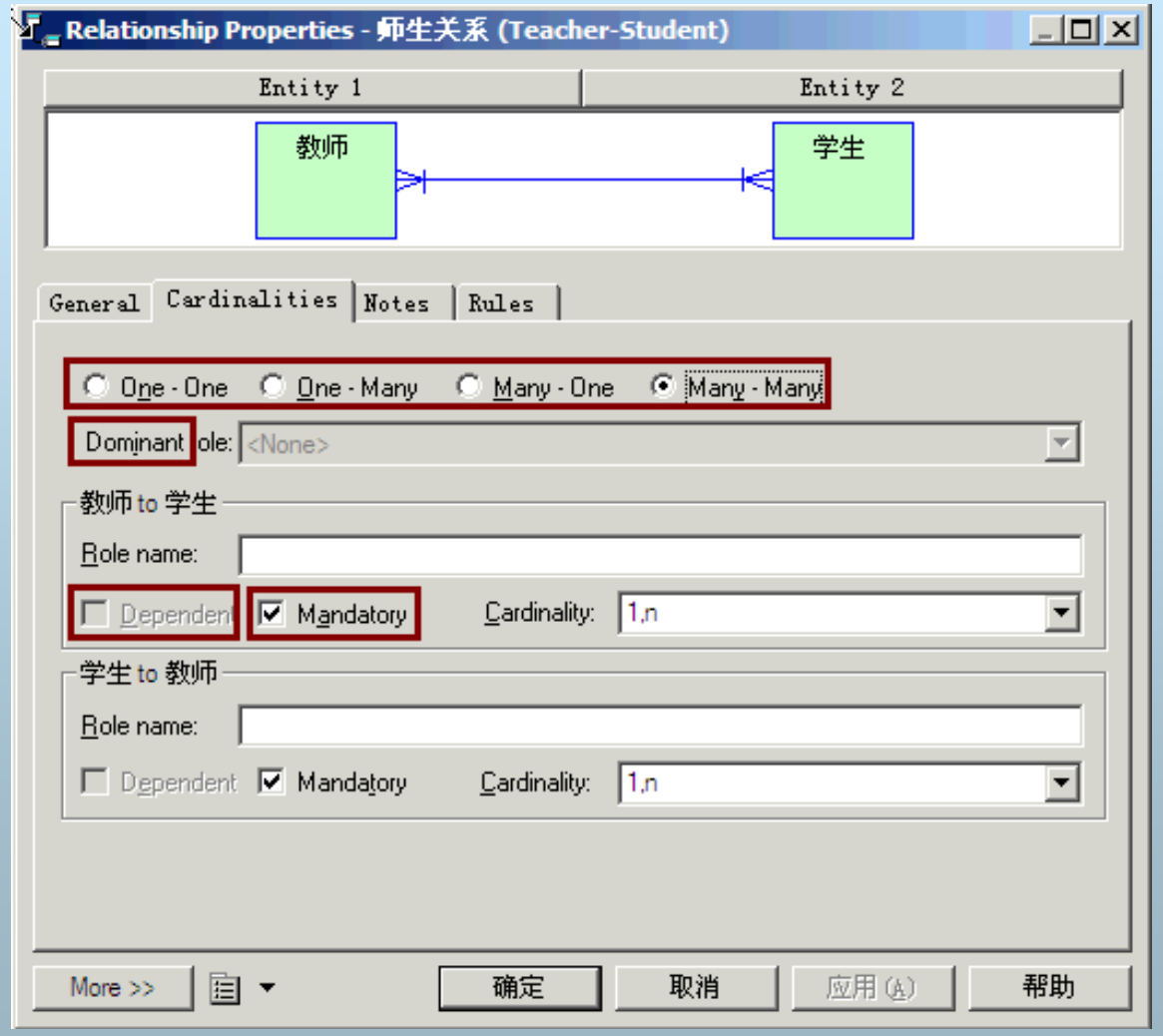
（5）子类例子



Power Designer中的符号



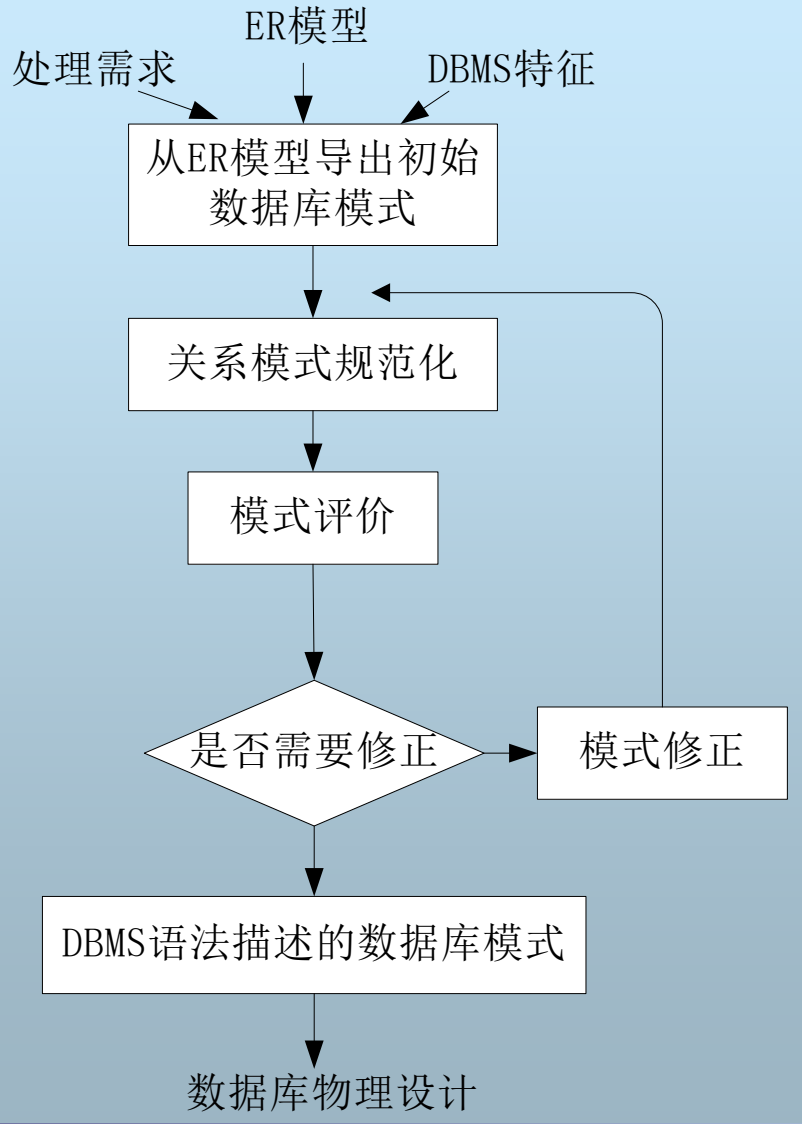
Power Designer中的符号



五、数据库逻辑设计

- 根据概念模型设计出与DBMS支持的数据模型相符合的数据库逻辑结构
- 主要工作
  - ER模型向关系模型的转换
  - 关系模型优化
  - 关系模型修正

1、数据库逻辑设计步骤



① **ER模型转换成关系数据库模式**

② 关系数据库模式的规范化

③ 模式评价

④ 模式修正

⑤ 最终产生一个优化的全局关系数据库模式

⑥ 子模式设计

2、ER模型向关系模型转换

- 基本ER模型的转换
- 扩展ER模型的转换

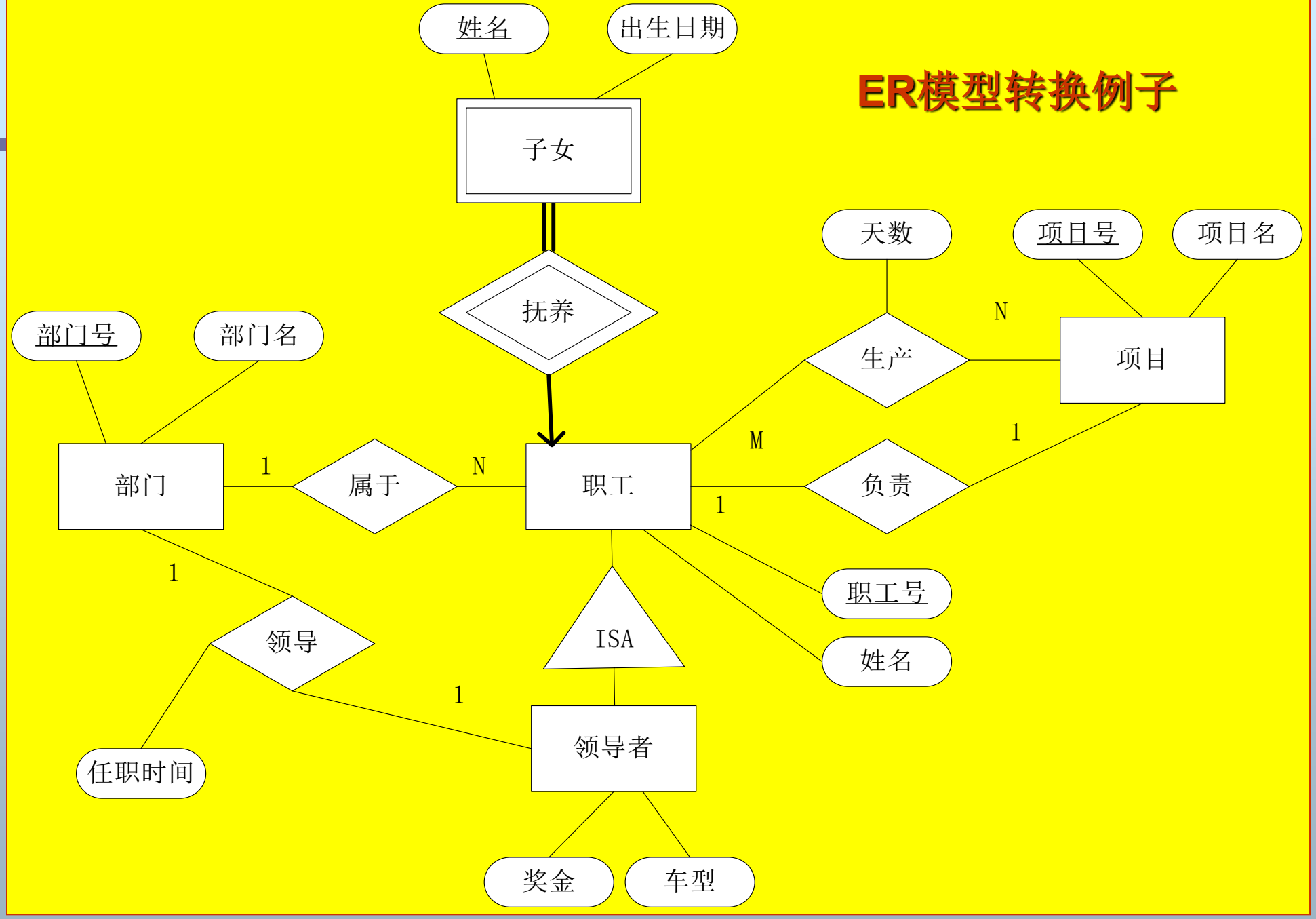
（1）基本ER模型转换到关系模型

- 实体转换
  - 每个实体转换为一个关系模式，实体的属性为关系模式的属性，实体的标识成为关系模式的主码
- 联系转换
  - **1:1**：将任一端的实体的标识和联系属性加入另一实体所对应的关系模式中，两模式的主码保持不变
  - **1:N**：将1端实体的标识和联系属性加入N端实体所对应的关系模式中，两模式的主码不变
  - **M:N**：新建一个关系模式，该模式的属性为两端实体的标识以及联系的属性，主码为两端关系模式的主码的组合

（2）扩展ER模型转换到关系模型

- 弱实体转换
  - 每个强实体转换为一个关系模式，强实体的属性成为关系模式的属性，实体标识成为主码
  - 每个弱实体转换为一个关系模式，并加入所依赖的强实体的标识，关系模式的主码为弱实体的标识加上强实体的标识
- 子类转换
  - 父类实体和子类实体都各自转换为关系模式，并在子类关系模式中加入父类的主码，子类关系模式的主码设为父类的主码

ER模型转换例子


姓名
出生日期
子女
抚养
天数
项目号
项目名
部门号
部门名
生产
项目
N
部门
1
属于
N
职工
M
负责
1
1
1
职工号
领导
ISA
姓名
1
领导者
任职时间
奖金
车型

转换实体

实体转换为关系模式：

1、部门（部门号，部门名）

2、职工（职工号，姓名）

3、项目（项目号，项目名）

4、领导者（奖金，车型）

5、子女（姓名，出生日期）

先考虑弱实体“子女”，加入“职工号”，并修改主码为“职工号＋姓名”

5、子女（姓名，出生日期，职工号）

在考虑子类“领导者”，加入父类标识“职工号”作主码

4、领导者（奖金，车型，职工号）

转换联系

实体转换得到关系模式：

1、部门（部门号，部门名）

2、职工（职工号，姓名）

3、项目（项目号，项目名）

4、领导者（奖金，车型，职工号）

5、子女（姓名，出生日期，职工号）

考虑每个联系：

1、部门:领导者（1:1）：领导者（奖金，车型，职工号，部门号，任职时间）

或者　部门（部门号，部门名，职工号，任职时间）

2、部门:职工（1:N）：职工（职工号，姓名，部门号）

3、职工:项目（1:1）：项目（项目号，项目名，职工号）

4、职工:项目（M:N）：增加模式：职工_项目（项目号，职工号，天数）

得到初步的关系数据库模式

1. 部门（部门号，部门名）
2. 职工（职工号，姓名，部门号）
3. 项目（项目号，项目名，职工号）
4. 领导者（奖金，车型，职工号，部门号，任职时间）
5. 子女（姓名，出生日期，职工号）
6. 职工_项目（项目号，职工号，天数）