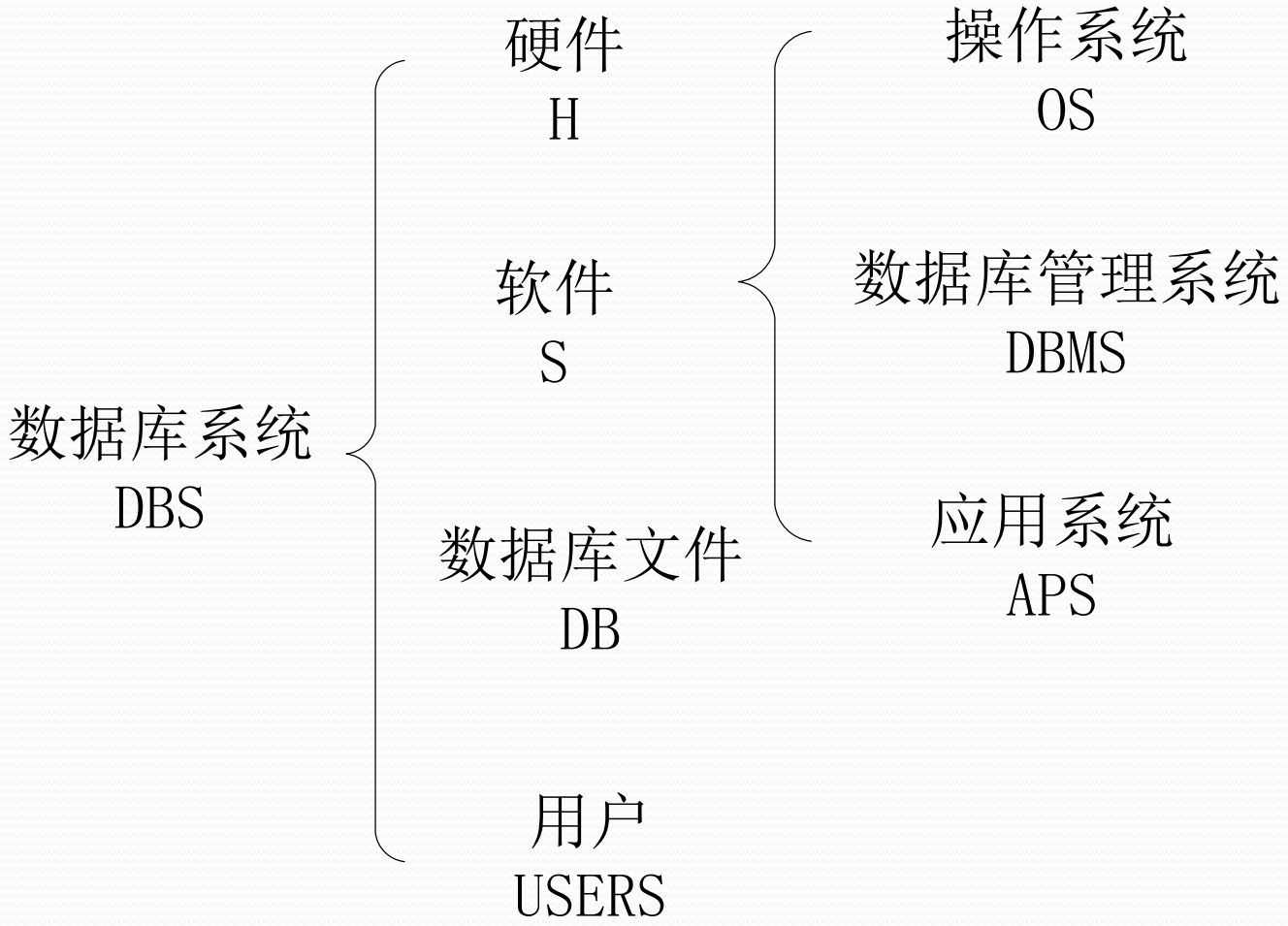
第7章 数据库设计

7.1 概述

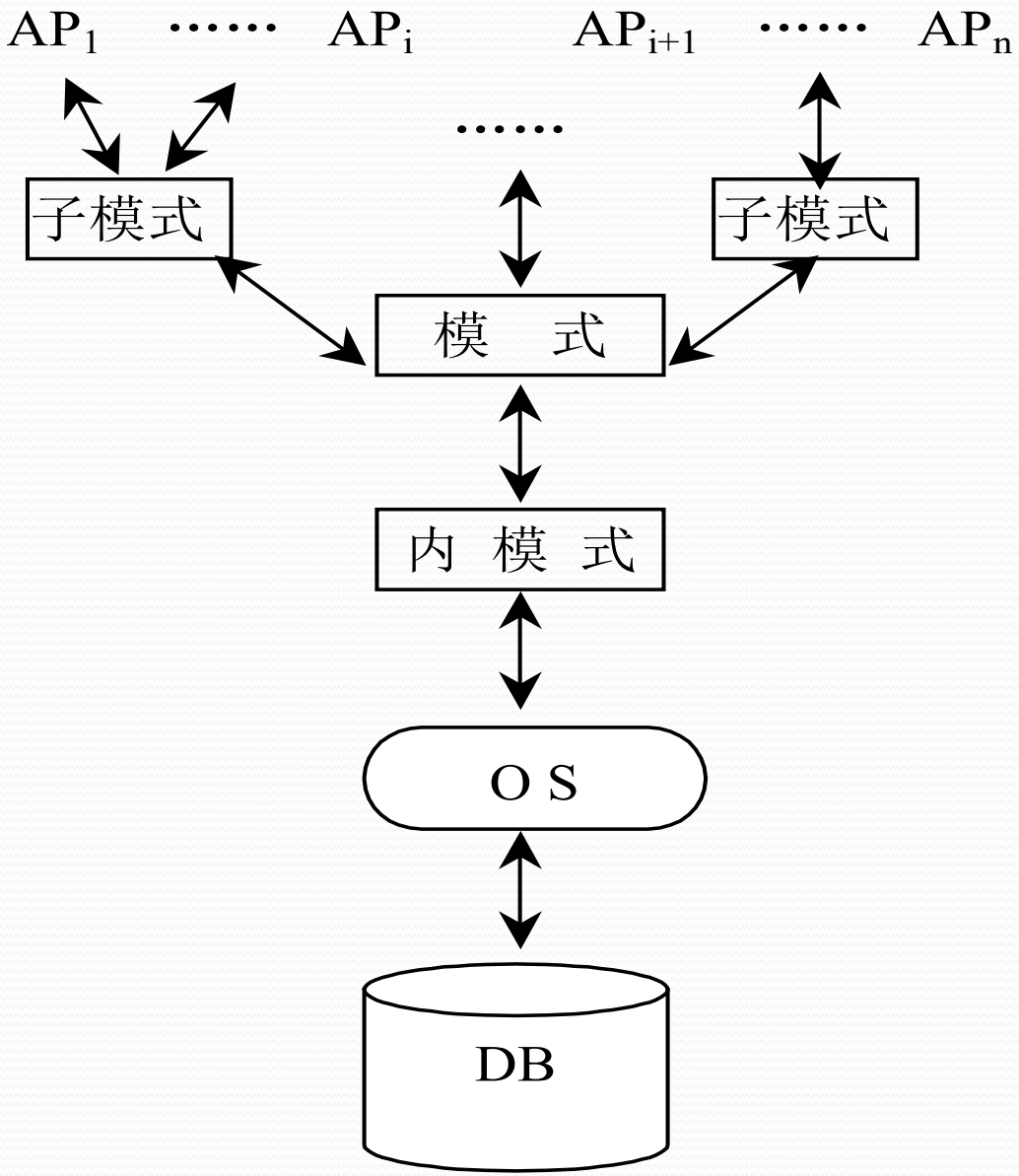
7.1.1 数据库设计概念

对于一个给定的应用环境，构造最优的数据库模式（模式、内模式），建立数据库及其应用系统，使之能够有效的存储和管理数据，满足各种用户的应用需求（信息要求和数据操作要求）。

1）DBS组成


数据库系统
DBS
硬件
H
软件
S
数据库文件
DB
用户
USERS
操作系统
OS
数据库管理系统
DBMS
应用系统
APS

2）DBS结构



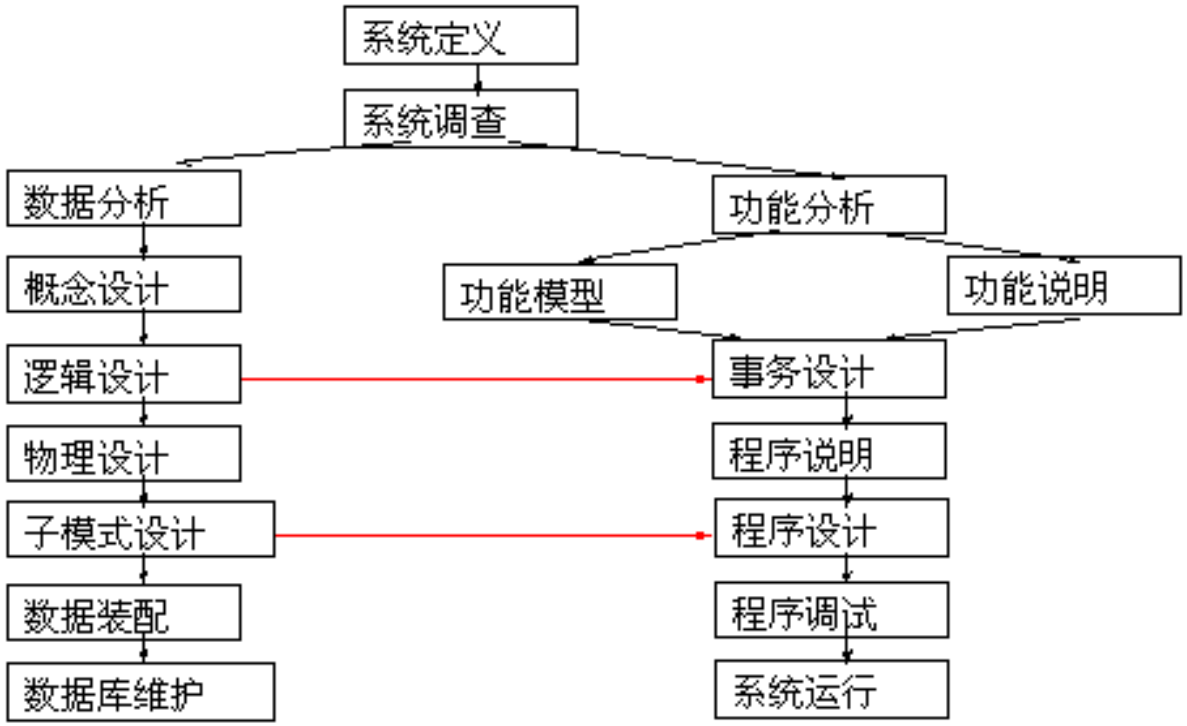
3）功能

从用户应用需求出发，设计并建立数据库的结构（逻辑结构和物理结构）。

7.1.2 数据库设计方法

数据库设计应该和应用系统设计相结合，即系统的结构（数据）设计和行为（处理）设计密切结合。

早期的数据库设计方法是结构与行为分离的。



数据库设计方法目前可分为四类：

1）直观设计法（也叫手工试凑法）

2）规范设计法

3）计算机辅助设计法

4）自动化设计法。

至今，人们经过努力探索，提出了各种规范设计方法，其基本思想都是过程迭代和逐步求精。

规范设计方法举例

新奥尔良法

1978年10月，来自三十多个国家的数据库专家在美国新奥尔良（New Orleans）市专门讨论了数据库设计问题，他们运用软件工程的思想和方法，提出了数据库设计的规范，这就是著名的新奥尔良法，它是目前公认的比较完整和权威的一种规范设计法。

新奥尔良法将数据库设计分成：

需求分析（分析用户需求）

概念设计（信息分析和定义）

逻辑设计（设计实现）

物理设计（物理数据库设计）。

基于E-R模型的数据库设计方法

由P.P.S.chen于1976年提出的数据库设计方法，其基本思想是在需求分析的基础上，用E-R（实体—联系）图构造一个反映现实世界实体之间联系的企业模式，然后再将此企业模式转换成基于某一特定的DBMS的模式。

基于3NF的数据库设计方法

由S·Atre提出的结构化设计方法，其基本思想是在需求分析的基础上，确定数据库模式中的全部属性和属性间的依赖关系，将它们组织在一个单一的关系模式中，然后再分析模式中不符合3NF的约束条件，将其进行投影分解，规范成若干个3NF关系模式的集合。

基于视图的数据库设计方法

先从分析各个应用的数据着手，为每个应用建立自己的视图，然后再把这些视图汇总起来合并成整个数据库的概念模式。

除了以上几种方法外，规范化设计方法还有实体分析法、属性分析法和基于抽象语义的设计方法等。

计算机辅助设计法

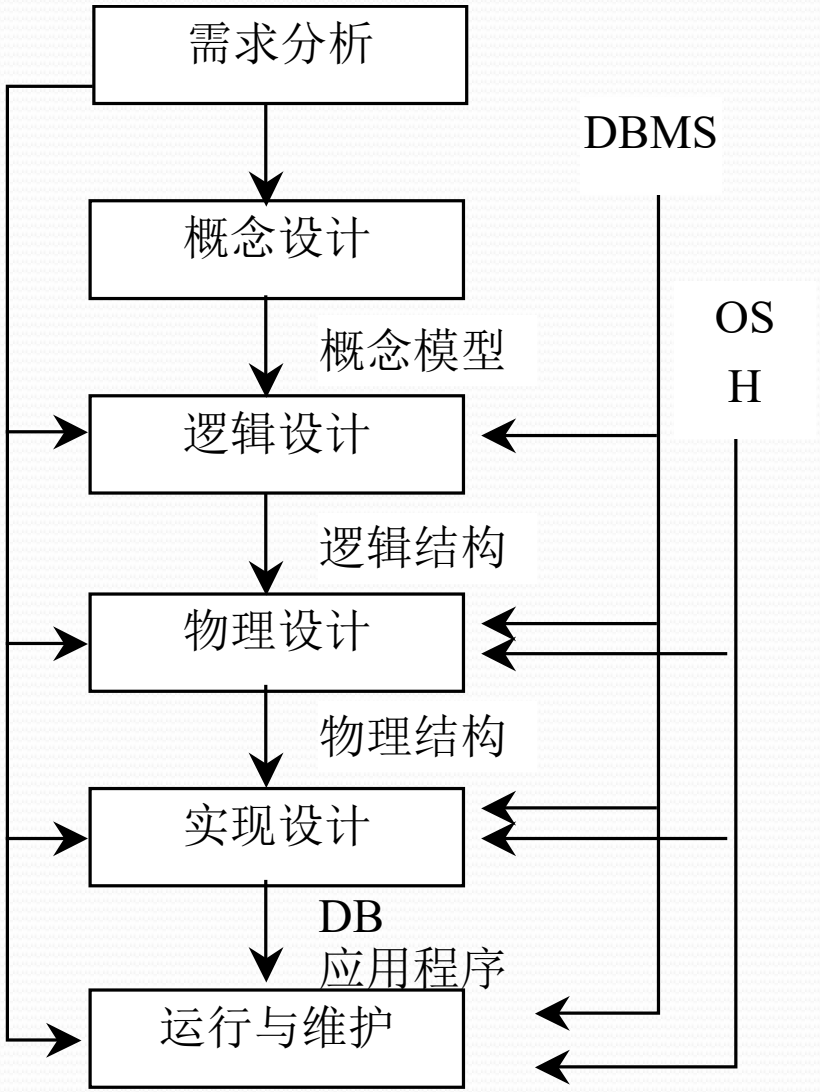
是指在数据库设计的某些过程中模拟某一规范化设计的方法，并以人的知识或经验为主导，通过人机交互方式实现设计中的某些部分。

目前许多计算机辅助软件工程（Computer Aided Software Engineering，CASE）工具可以自动或辅助设计人员完成数据库设计过程中的很多任务。例如：

SYSBASE公司的PowerDesigner、

Oracle公司的Designer。

7.1.3 DB设计步骤


需求分析
概念设计
概念模型
逻辑设计
逻辑结构
物理设计
物理结构
实现设计
DB
应用程序
运行与维护
DBMS
OS
H

7.2 需求分析

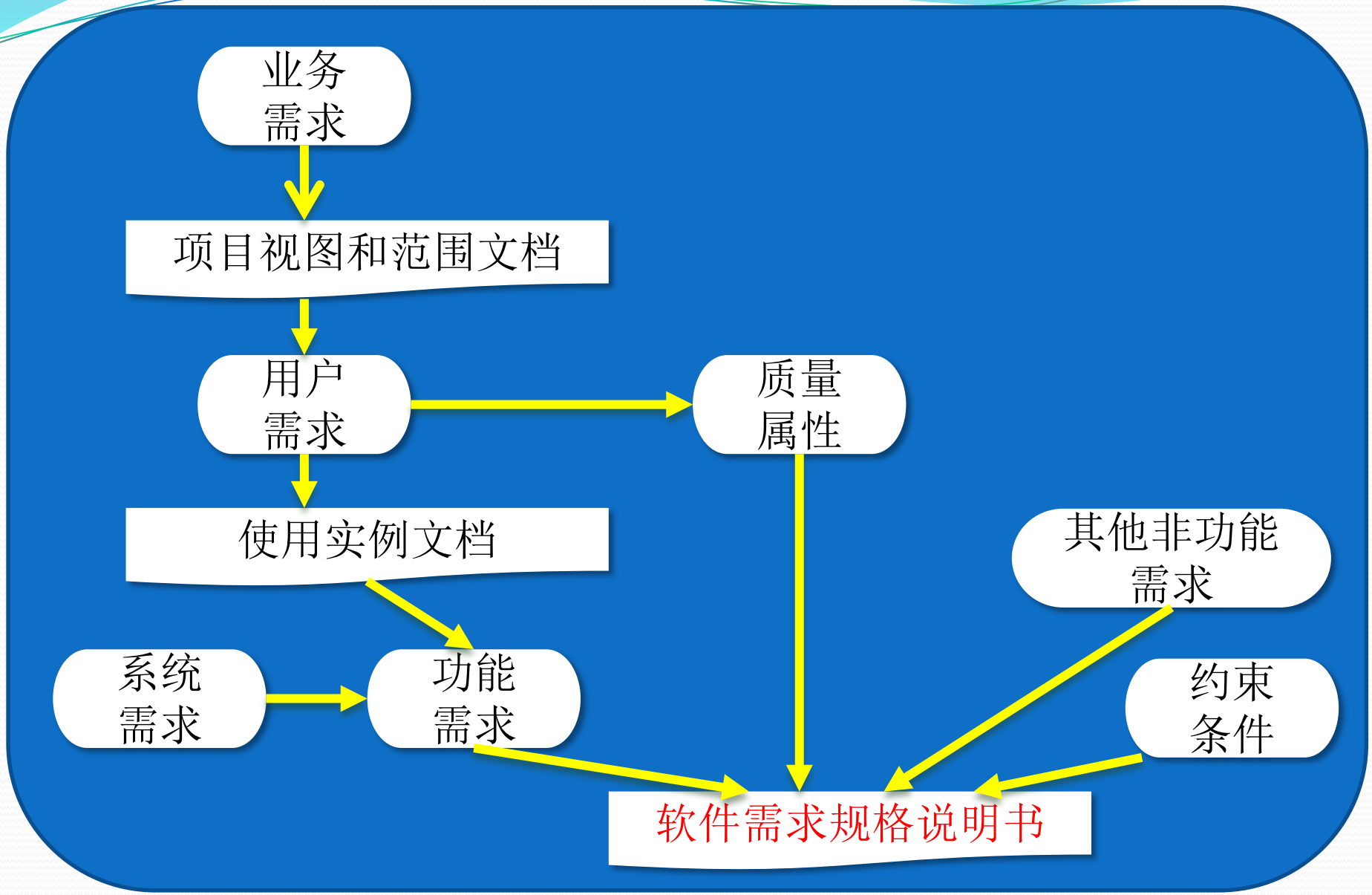
7.2.1 任务

1、掌握用户功能需求；
2、掌握用户数据需求；
3、掌握用户处理需求；
4、掌握用户的约束需求。

响应时间

处理方式：联机、批处理

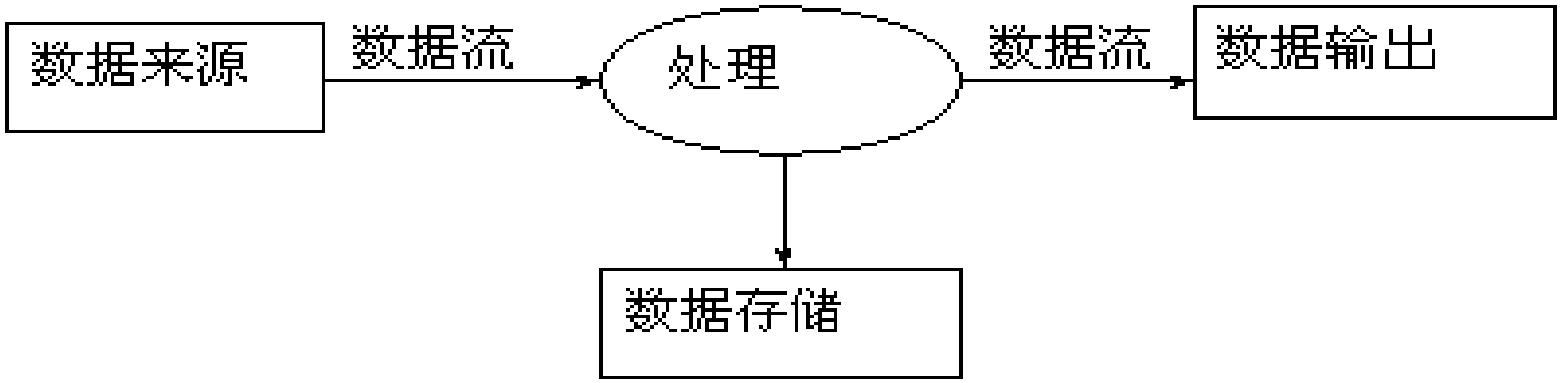
时间？空间？数据语义规范？


业务需求
项目视图和范围文档
用户需求
质量属性
使用实例文档
其他非功能需求
系统需求
功能需求
约束条件
软件需求规格说明书

7.2.2 方法

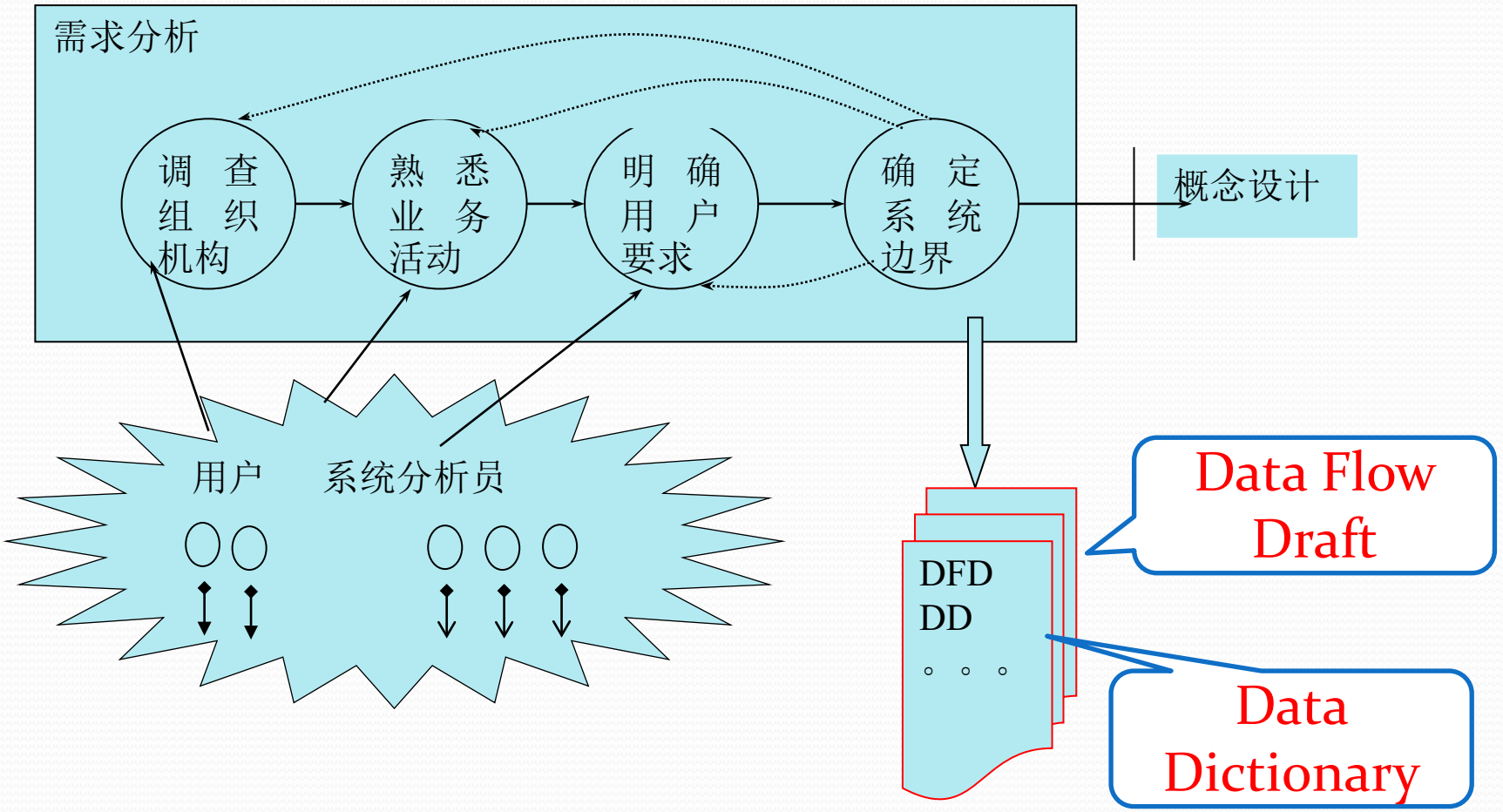
1、目标

全面、准确、客观、详细。



结构化分析（Structured analysis, SA）方法

需求分析过程


需求分析
调查组织机构
熟悉业务活动
明确用户要求
确定系统边界
概念设计
用户
系统分析员
DFD
DD
Data Flow Draft
Data Dictionary

2、策略

1）自顶向下；2）自底向上。

3、步骤

1）调查机构与职能。

2）业务处理流程；

① 部门与数据；

② 数据与数据；

③ 处理与数据。

3、步骤（续）

3）DB约束

① 时间；② 空间；③ 完整性；④ 安全性。

4）建立需求分析说明书。

4、调查方式

1）报告；2）座谈；3）跟班作业；4）阅读。

5、技巧

1）调查提纲； 2）友好气氛；3）导航。

7.2.3 需求分析说明书

7.2.3.1 构成

① 引言；

② 机构与功能；

③ DFD(Data Flow Draft)；

④ DD(data dictionary)

7.2.3.2 数据字典

——数据的收集和分析的结果。

① 数据项

名、含义、类型、长度、取值范围、取值含义、与其他数据项的逻辑关系、备注。

② 数据结构

一般由若干数据项组成。

③ 数据流

数据流名、来源、去向、流量（平均流量、高峰期流量）

④数据存储

数据存储名（数据流的来源和去向如手工或计算机文档）、输入流、输出流、数据量、存取频度、存取方式。

⑤ 处理过程

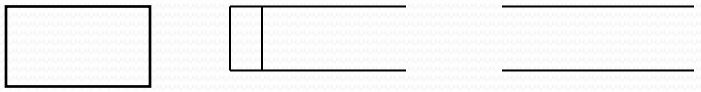
过程名、功能、使用数据、频率、处理算法、备注。

数据流程图 （可参见第四版教材216～218页例子，自上而下细化）

- 处理过程

- 数据流

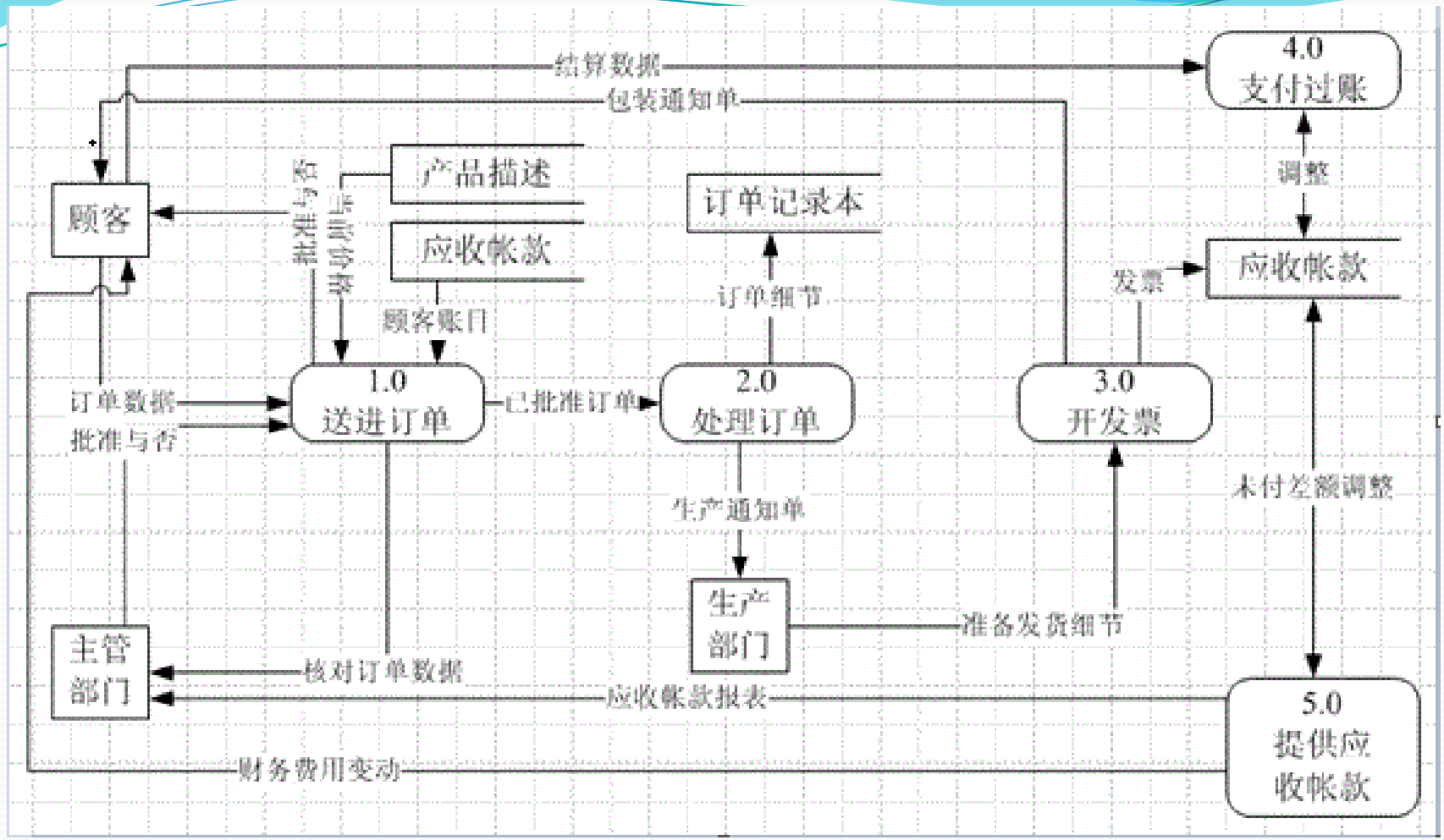
- 数据流的终点或源点



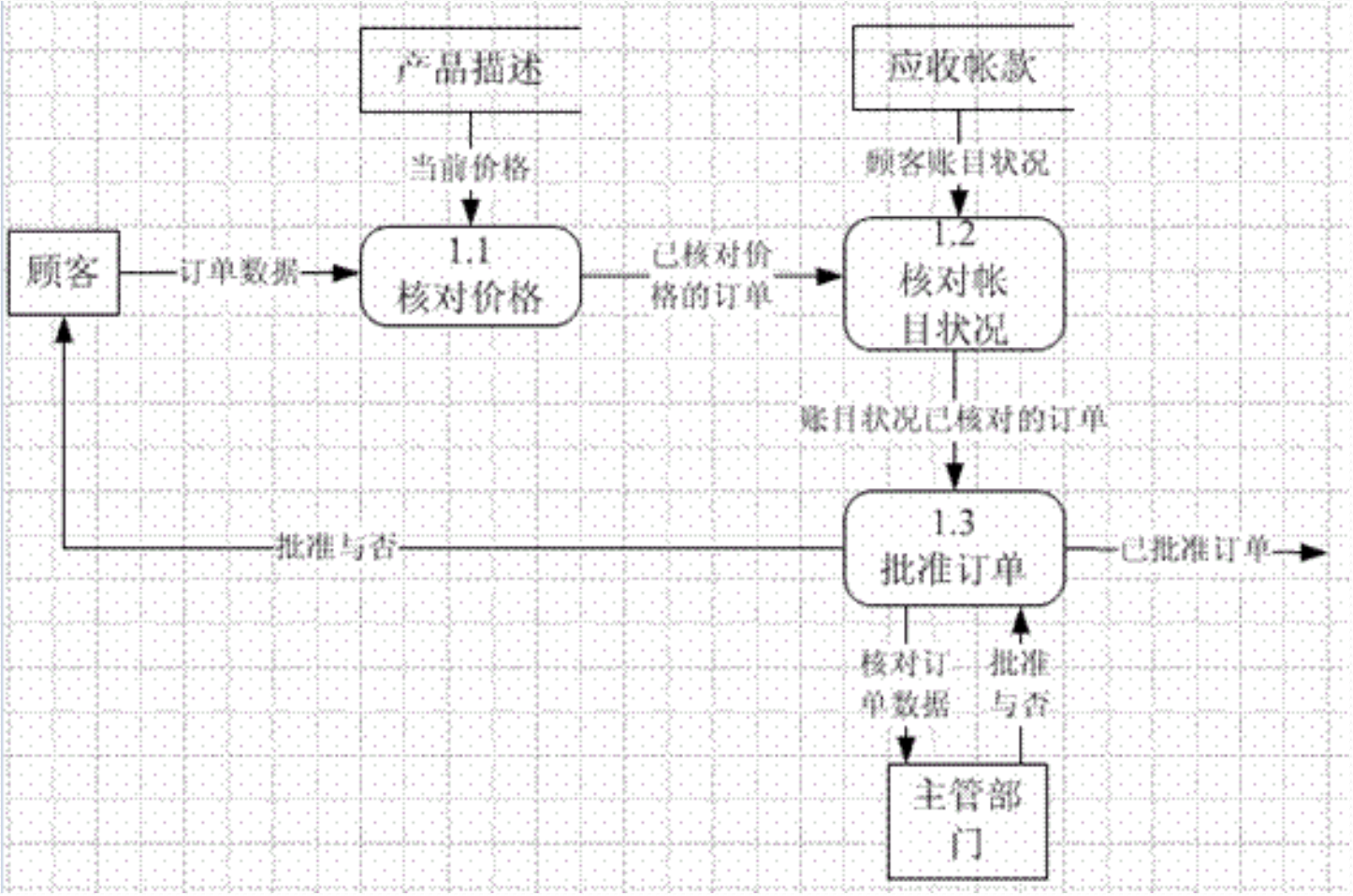
- 存储池

7.2.3.3 要求

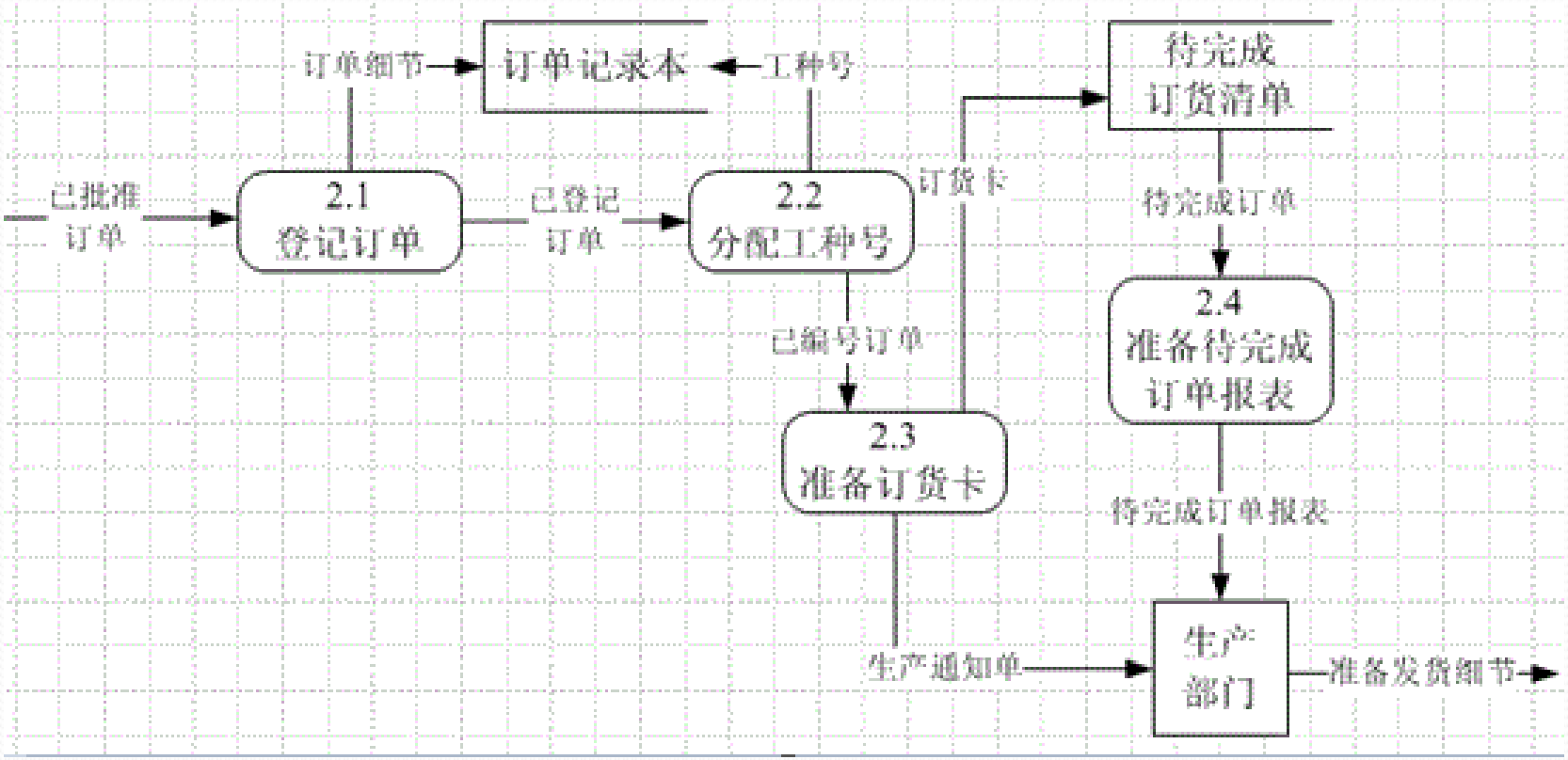
① 准确；② 全面；③ 详细；④ 客观；⑤无二义性。


结算数据
包装通知单
4.0
支付过账
调整
顾客
产品描述
应收帐款
订单记录本
应收帐款
发票
订单细节
顾客账目
订单数据
批准与否
1.0
送进订单
已批准订单
2.0
处理订单
3.0
开发票
未付差额调整
生产通知单
生产
部门
准备发货细节
主管
部门
核对订单数据
应收帐款报表
5.0
提供应
收帐款
财务费用变动

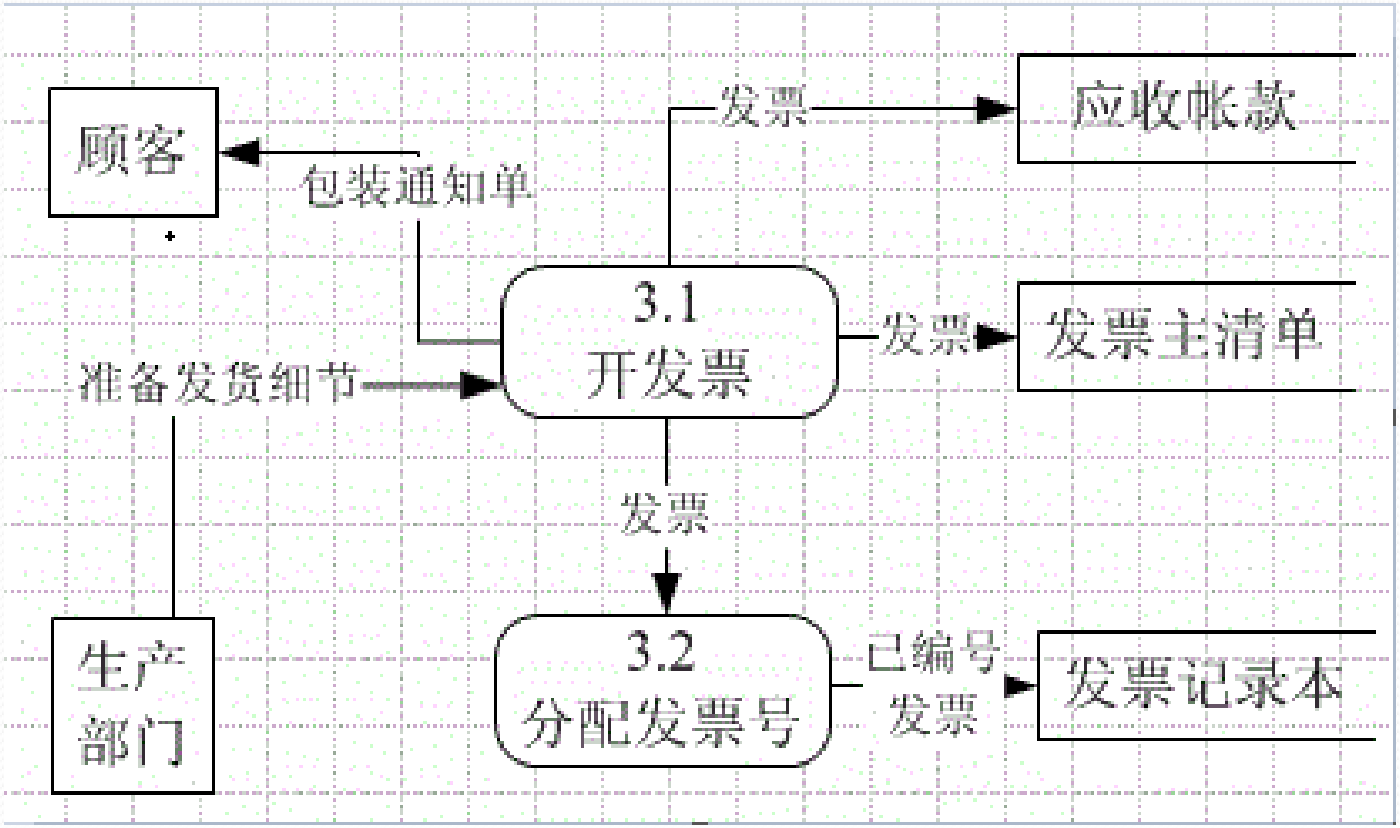
第一层 数据流图


产品描述
应收帐款
当前价格
顾客账目状况
顾客
订单数据
1.1
核对价格
已核对价格的订单
1.2
核对帐目状况
账目状况已核对的订单
1.3
批准订单
已批准订单
批准与否
核对订单数据
批准与否
主管部门

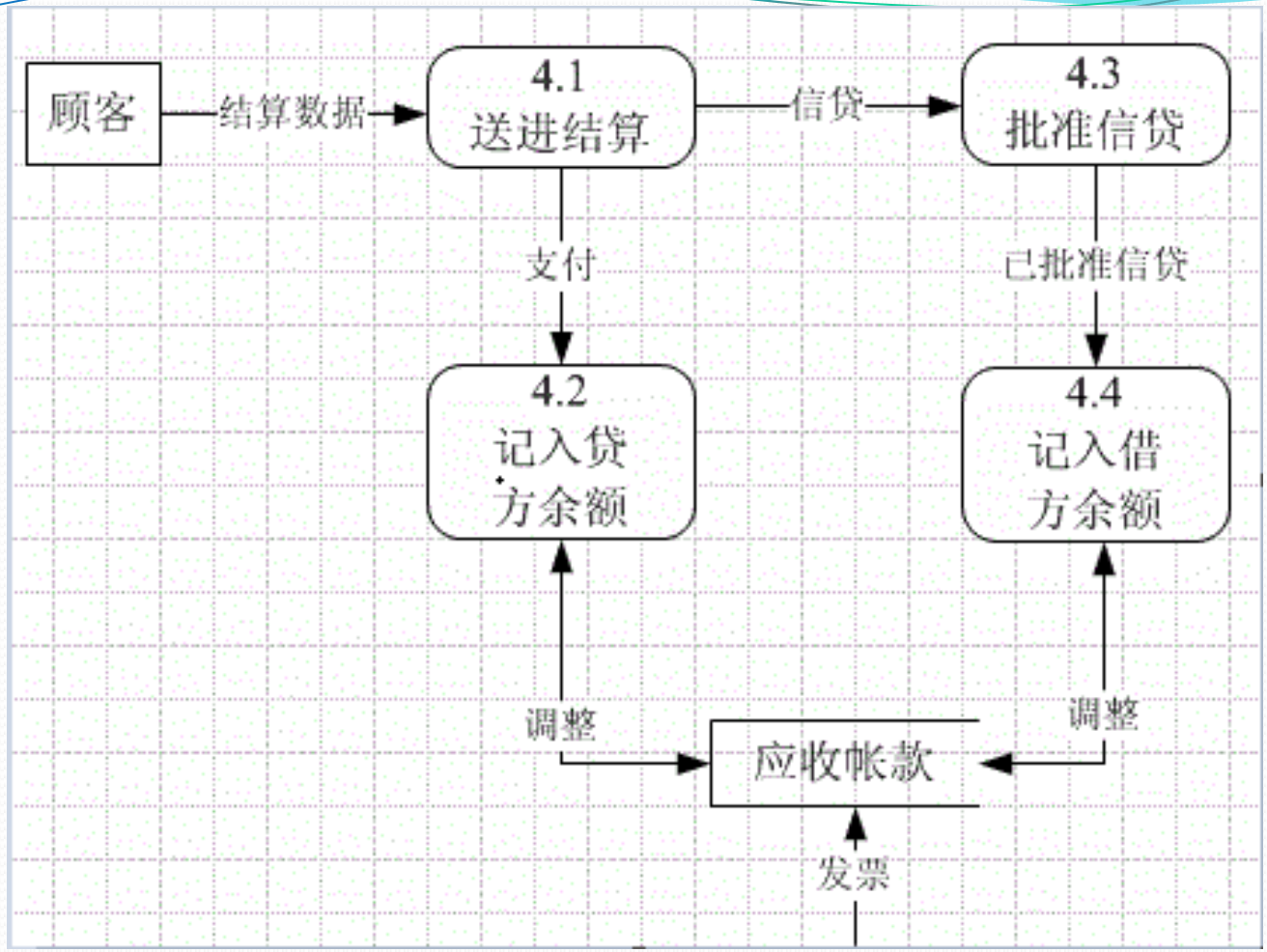
第二层　送进订单


订单细节
订单记录本
工种号
待完成
订货清单
已批准
订单
2.1
登记订单
已登记
订单
2.2
分配工种号
订货卡
待完成订单
2.4
准备待完成
订单报表
已编号订单
2.3
准备订货卡
待完成订单报表
生产通知单
生产
部门
准备发货细节

第二层　处理订单


顾客
包装通知单
发票
应收帐款
3.1
开发票
发票
发票主清单
准备发货细节
发票
生产
部门
3.2
分配发票号
已编号
发票
发票记录本

第二层　开发票


顾客
结算数据
4.1
送进结算
信贷
4.3
批准信贷
支付
已批准信贷
4.2
记入贷
方余额
4.4
记入借
方余额
调整
调整
应收帐款
发票

第二层　支付过账

需求说明文档（部分举例）

级别编号	模块名称	功能说明	子模块	操作步骤	操纵数据
01	用户登录子模块	1.通过用户名和用户口令来控制该系统的合法用户及其相应的权限。 2.用户分高级用户和普通用户两类，高级用户为综合科工作人员，普通用户包括销售科工作人员和各销售员		1.选择用户名； 2.输入相应口令； 3.系统判断该用户的合法性以及相应权限，并进入相应操作界面	系统用户功能权限表（**）

需求说明文档（部分举例）

级别编号	模块名称	功能说明	子模块	操作步骤	操纵数据
02	接收订单	1.接收顾客提交的订单数据，并完成价格核对、账目核对以及批准的工作。 2.涉及顾客、产品描述、应收账款、订单数据	021 核对价格 022 核对账目状况 023 批准订单	1.用户提交； 2.读取产品信息，核对价格 3.读取顾客账目信息，核对账目状况 4.主管部门审批。	涉及顾客、产品基本信息、应收账款、订单数据

7.3 概念结构设计

7.3.1 概述

7.3.1.1 任务

设计满足用户需求的概念模型。

7.3.1.2 描述工具

E-R图

7.3.1.3 特征

1）独立于硬件；2）独立于软件（OS、DBMS）。

概念结构设计的方法

设计概念结构的E-R模型可采用四种方法。

1) 自顶向下。先定义全局概念结构E-R模型的框架，再逐步细化。

2) 自底向上。先定义各局部应用的概念结构E-R模型，然后将它们集成，得到全局概念结构E-R模型。

3) 逐步扩张。先定义最重要的核心概念E-R模型，然后向外扩充，以滚雪球的方式逐步生成其他概念结构E-R模型。

4) 混合策略。该方法采用自顶向下和自底向上相结合的方法，先自顶向下定义全局框架，再以它为骨架集成自底向上方法中设计的各个局部概念结构。

- 常用的方法是“自底向上”——先自顶向下的进行需求分析，再自底向上地设计概念结构。

7.3.1.4 步骤

1）局部E-R图设计；

2）全局E-R图集成；

3）优化。

7.3.2 局部E-R图设计

7.3.2.1 任务

1、确定局部范围；
2、设计局部E-R图。

7.3.2.2 确定范围原则

1、相对独立；
2、内部联系较紧密；
3、与外部联系相对较少。

7.3.2.3 步骤

1、确立实体；

2、确立联系。

7.3.2.4 确定实体与属性

1. 困难：实体与属性之区别（相对）

2. 任务：① 命名 ② 确定实体码 ③ 确定实体内属性

3. 方法

1）以需求分析说明书中数据字典的数据结构为基础。

2） 可再分者为实体

年龄显然作属性

单位若还可细分为DH、DM、DD，则为实体。

3）实体内部属性不能再与其它实体有联系。

同一实体内两属性间可有联系，但一般不应与其它实体发生联系（这种联系应该是实体间）。

三种抽象方法用于概念结构的提取：

分类（Classification, is member of）、

聚集（Aggregation, is part of）、

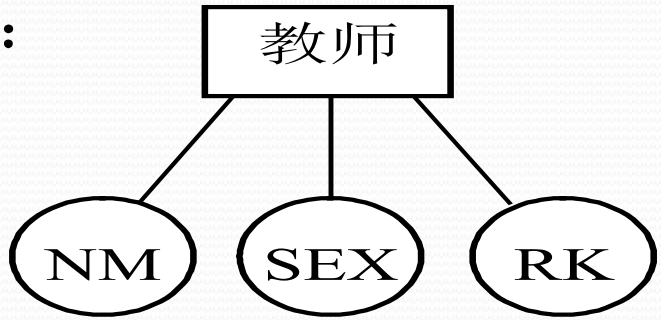
概括（Generalization, is subclass of, is superclass of）。

其中，概括方法可对E-R模型扩充，定义超类和子类实体。

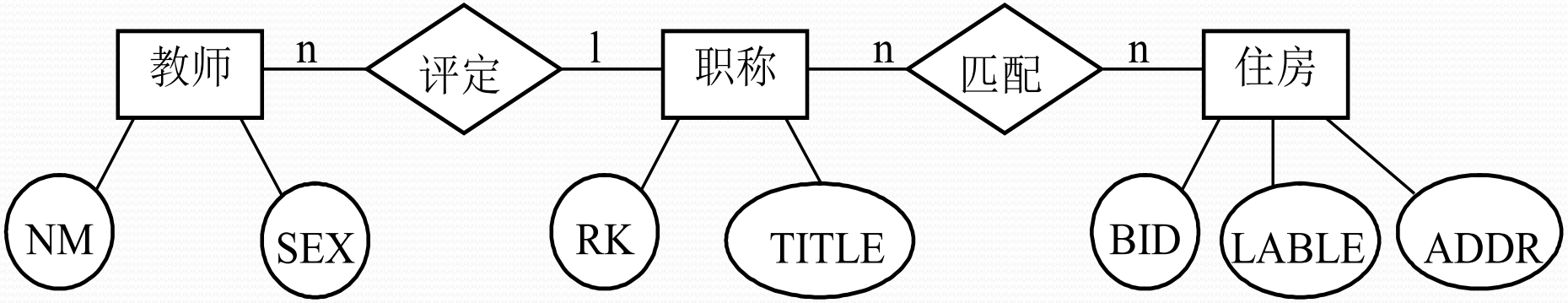
- 实体，属性一般的区分原则
  - •实体一般都有描述信息，而属性不一定
  - •多值的对象类可考虑作为实体
  - •一个对象类的某一个描述项如果和另一个对象类存在“多”的关系，那么即使它本身没有描述信息，也可将这个描述项作为实体（工厂——产品 ；
    运动员—— 项目）
  - •使用组合标识的对象类，如果组成这个标识的成分都是其他对象类的标识，一般应定义为联系，但如果不是，则可根据情况定义为实体。

例如：人名（first name+second name）
取款或缴费记录（终端机编号+该机的流水号）
教室（教学楼编号+教室编号）

例：一般“职称”作属性：



若职称与住房有联系则可作为实体：



4) 属性间存在m:1联系且m大时，“m”方宜作实体。
工厂（工厂名 ， 工厂地址， 工厂电话，帐号， 产品名）；
若一个工厂生产产品较多，则应将产品单独作为一实体。
5）勿轻易引入导出属性
例如：小计、合计……
6）应用规范化理论

7.3.2.5 确定联系

1. 任务
1）发现联系
2）确定联系方式
3）命名
4）确定KEY

联系举例

- 存在性联系

  系有学生，专业有必修课程，课程有参考书

- 功能性联系

  教师指导学生，技师负责维修机械

- 事件联系

  顾客发出订单，学生借书，学生听课，技师维修了某个机械，值班员检查了某个设备

2. 方法

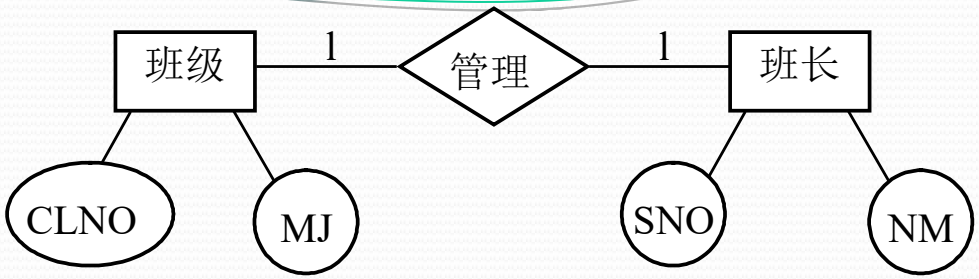
1）以DFD为基础

联系、方式

管理员——管理——用户？？？
应用需求是什么？要查询此类联系么？

2）KEY的确定

① 1：1时：取任何一方



② 1：m时：取“m”方KEY为联系实体KEY。

③ m：n时：取“双方”KEY为联系实体KEY。

7.3.3 视图集成

目的要求：

控制冗余、消除冲突、 实施集成

- 用分析法消除冗余属性
- 用关系理论消除冗余联系（最小依赖）
- 消除命名冲突（包括同名异义和异名同义）
- 消除属性冲突（包括域类型冲突和取值冲突）
- 消除结构冲突（联系方式、属性个数、顺序、抽象级别。。。）

7.3.3.1 方式

1. 多元集成法　　2. 二元集成法

7.3.3.2 步骤

1. 消除冲突　　2. 实施集成

7.3.3.3 冲突及处理

1. 命名冲实

1）同名异义（课堂→课堂头、课程）

2）同义异名（服务员、员工→职工）

2. 特征冲突（属性冲突）

1）属性值、类型、取值范围冲突。

2）属性取值单位冲突

米、尺、公斤、吨/升/元、人民币、比特币、Q币、美元、……

3. 结构冲突

1）联系方式冲突（未必能消除，要看语义能否合并）

1：1， 1：m， m：n

2）属性个数冲突

Student (XH，XM)

Student (XH，XM，XB，YL)

一般取多为结果，注意主关键字，注意效率影响。

3）属性次序冲突

Student （SNO，SNM，SEX）　　Student （SNM，SEX，SNO）

统一顺序

4）抽象级别冲突

同一对象在一局部应用中作实体，在另一局部应用中作属性。

按前述确定实体方法或合并为实体，或作为属性。

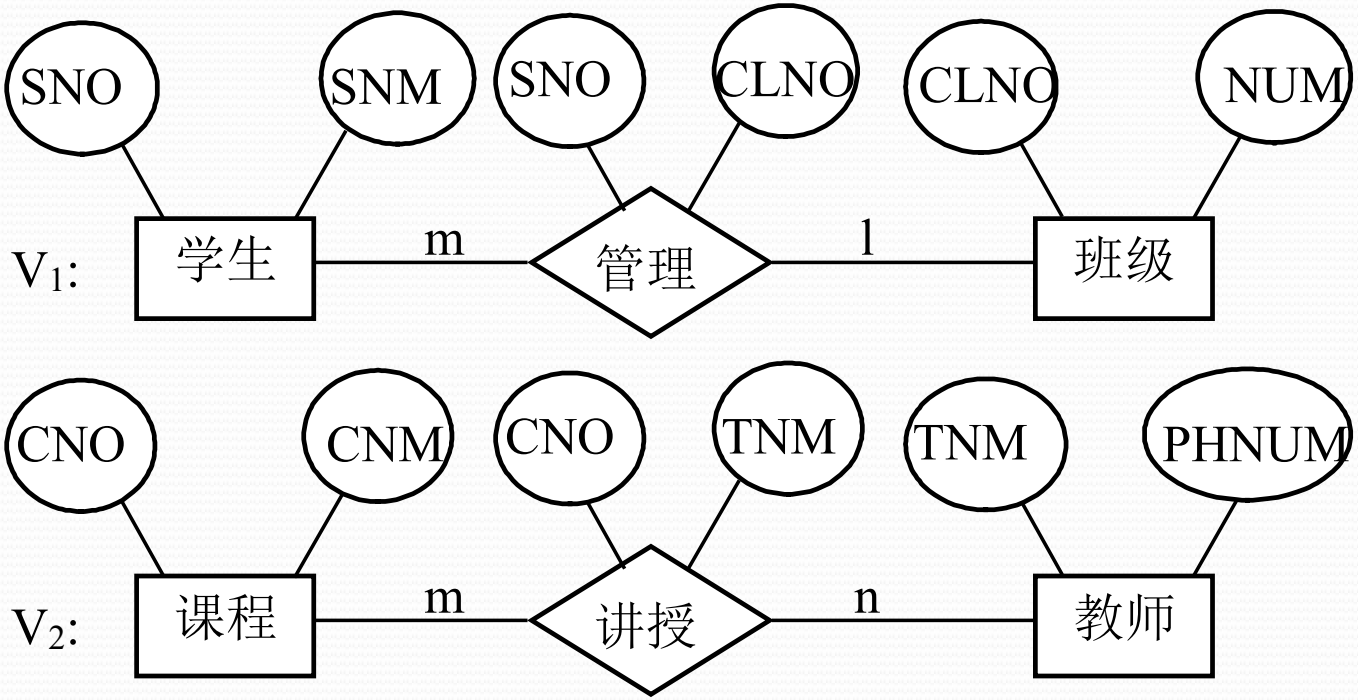
区别?

（区别之一在于是否有动态变化的需求，一旦确定为属性，则信息项个数固定，若保留为实体，则信息项还可增加，

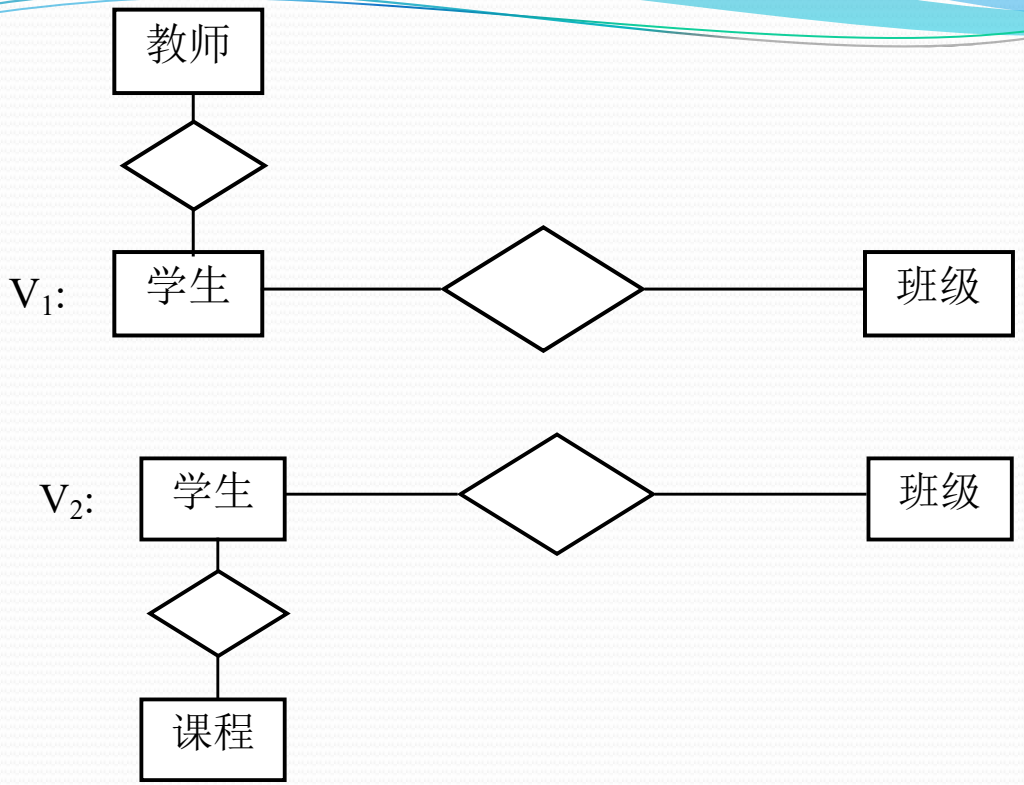
例如：成绩（平时成绩、上机成绩、考试成绩、中期测验、课堂提问。。。）

7.3.3.4 集成手段

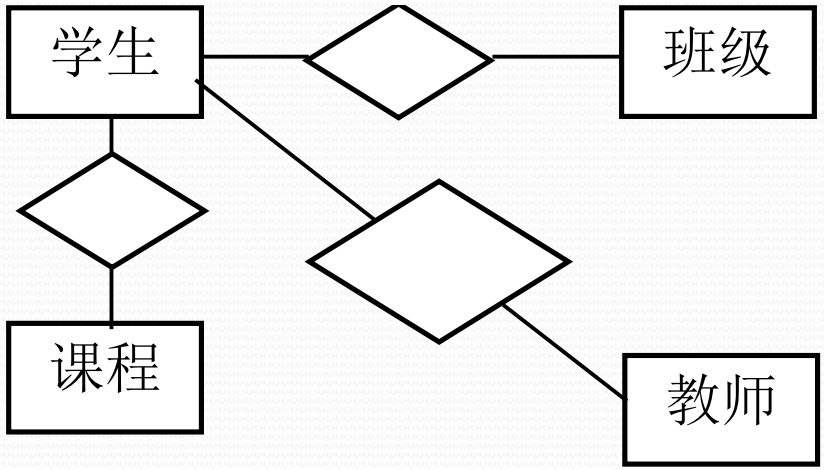
1. 合并



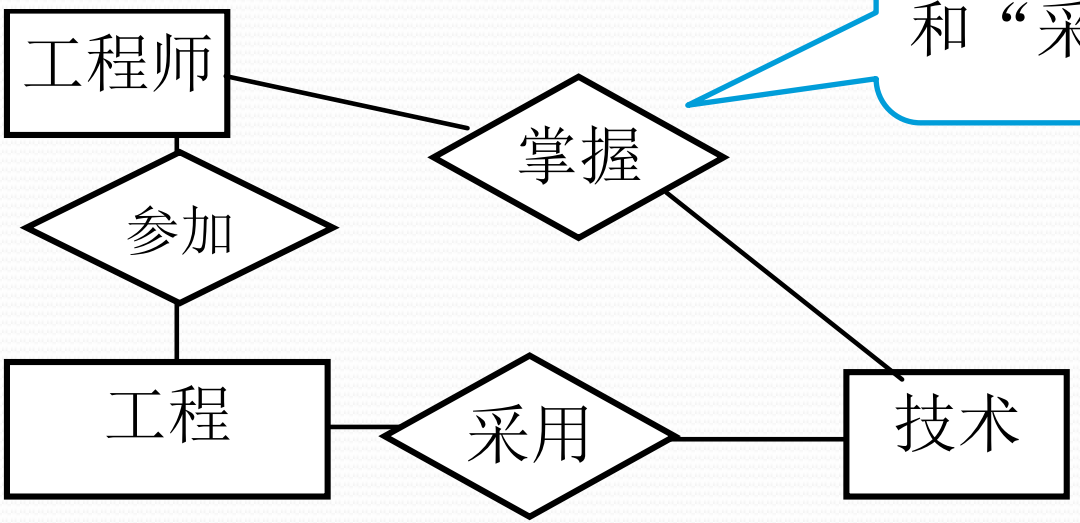
2. 找交集



合并后：

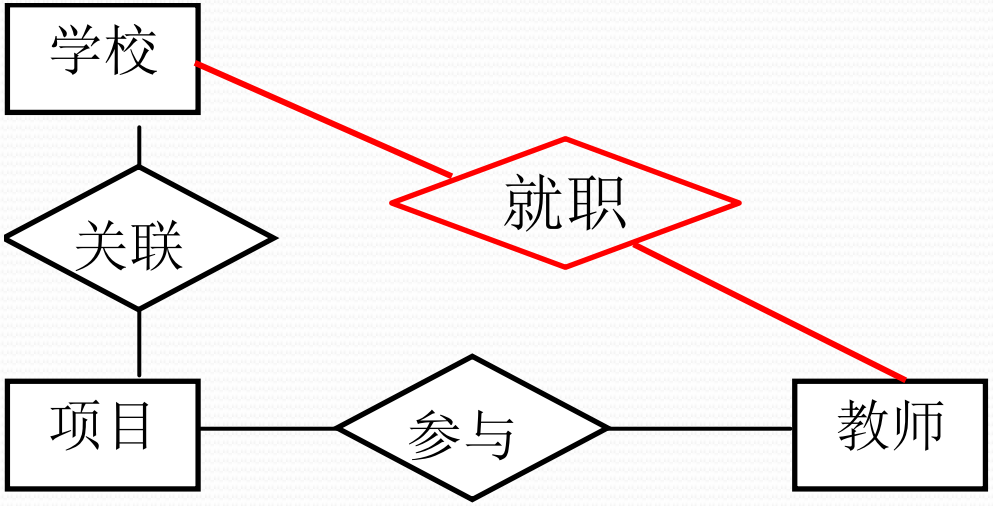


3、避免冗余联系



“掌握”可从“参加”和“采用”中推导出来。

4、避免联系陷井



若无就职联系，
教师直属学校
不能表达。

7.4 逻辑结构设计

7.4.1 任务

E-R图转换为等价的关系模型结构。

7.4.2 步骤

1、E-R⇒一般RDMS

2、一般RDMS⇒特定RDMS

3、优化

7.4.3 E-R图向一般RDMS转换

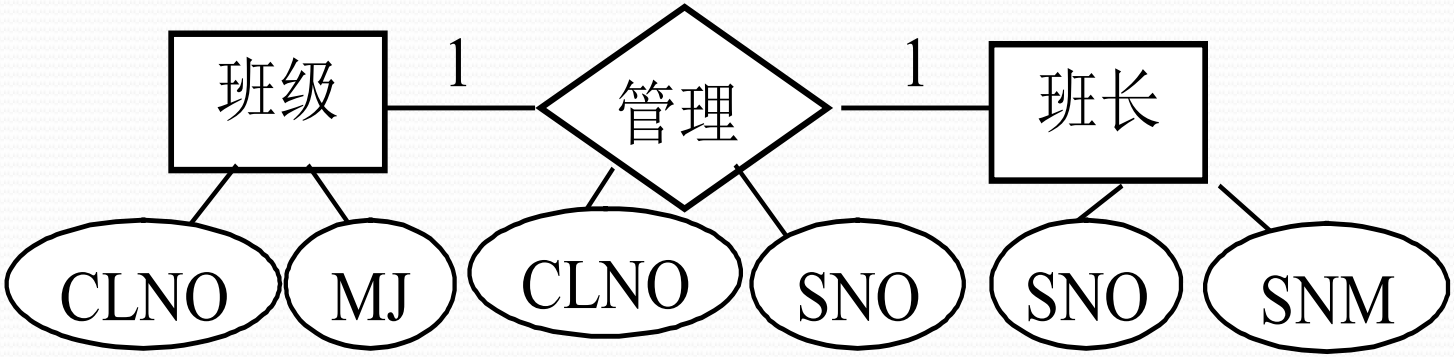
1、1：1

1）基本实体⇒一个关系模式；

实体的属性转换为关系的属性，实体的码=====>关系的码

2）联系⇒一个关系模式；

3）联系可消（优）化。



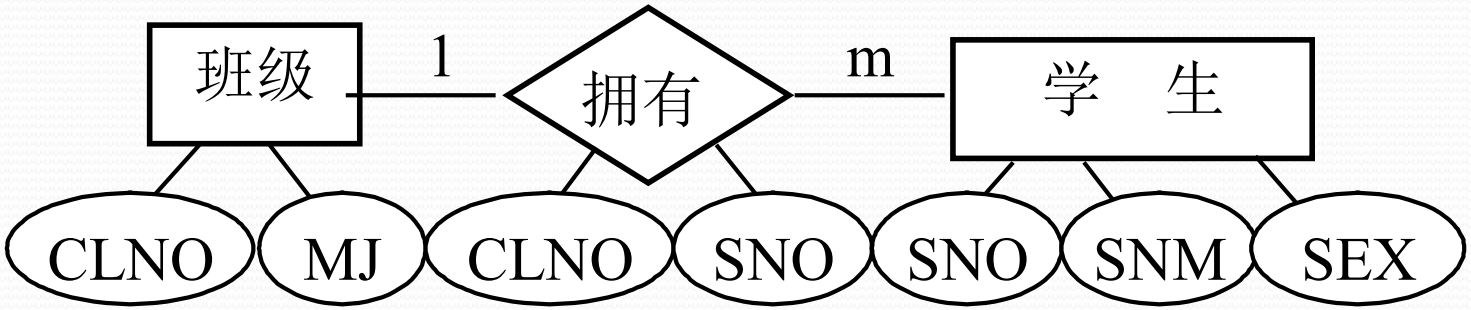
班级（CLNO，MJ）；
班长（SNO，SNM）；
管理（CLNO，SNO）。
消化联系后可为：
班级（CLNO，MJ，SNO）；
班长（SNO，SNM）。
或
班级（CLNO，MJ）；
班长（SNO，SNM，CLNO）。

2、1：m

1）基本实体⇒一个关系模式；

2）联系实体⇒一个关系模式；

3）联系实体可消化到“m”方中去。



班级（CLNO，MJ）；

学生（SNO，SNM，SEX）；

拥有（CLNO，SNO）。

消化联系后为：

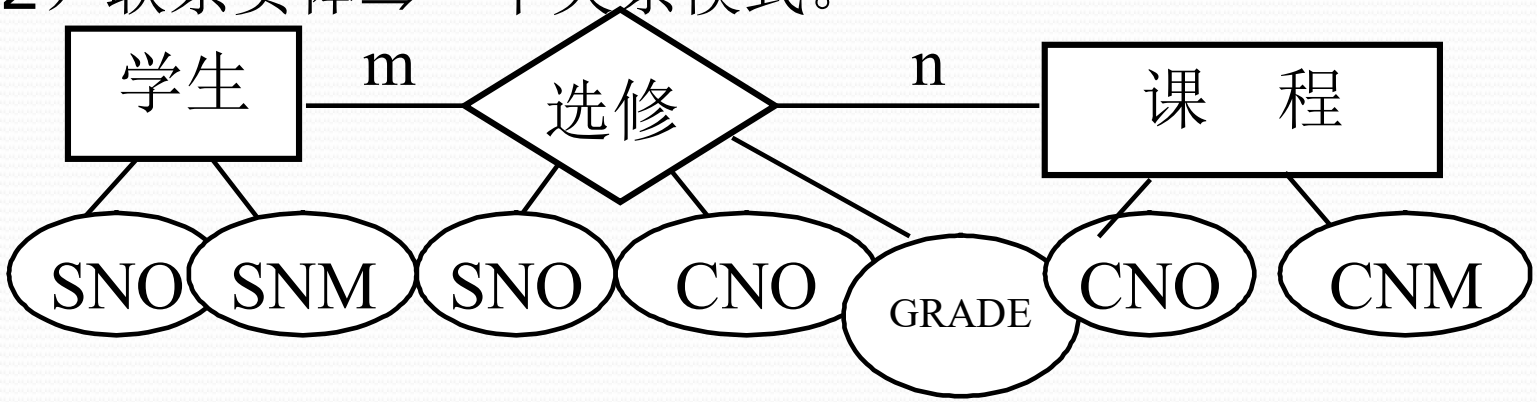
班级（CLNO，MJ）；

学生（SNO，SNM，SEX，CLNO）。

3、m：n

1）基本实体⇒一个关系模式；

2）联系实体⇒一个关系模式。

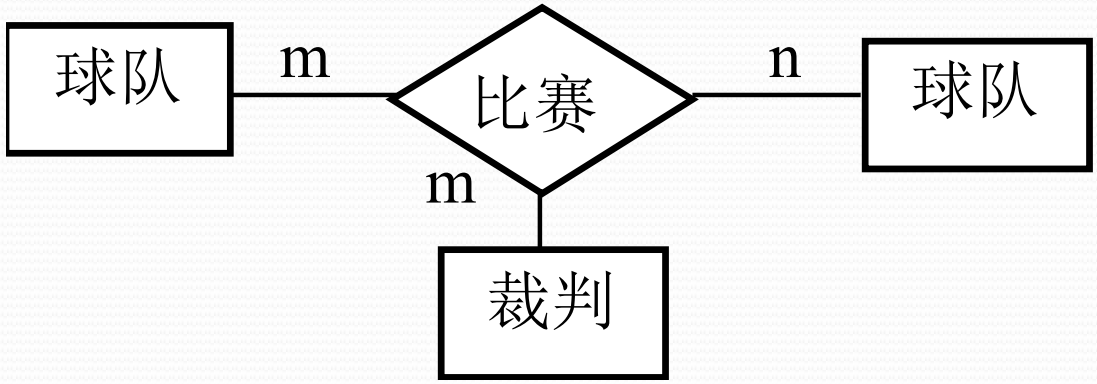


学生（SNO，SNM）；

课程：（CNO，CNM）；

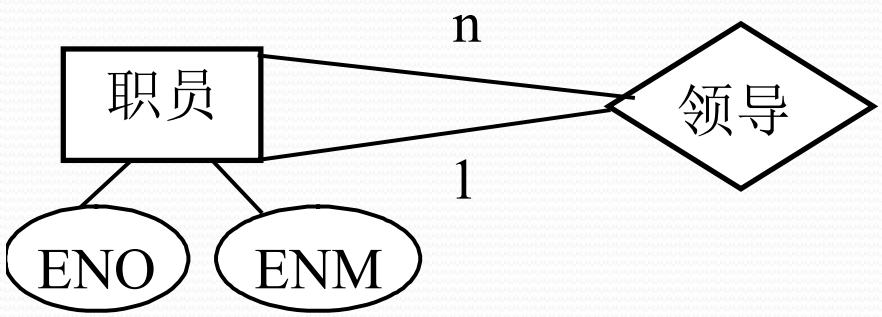
选修（SNO，CNO，GRADE）。

4、多元联系



1）基本实体⇒一个关系模式；
2）联系⇒一个关系模式。

5、自联系



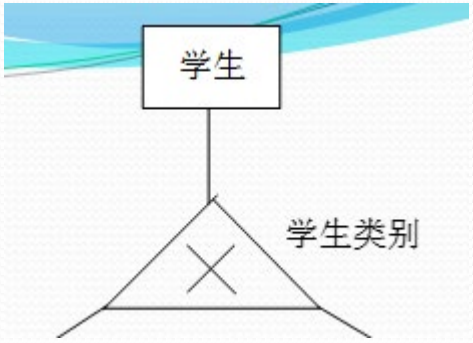
一个领导可领导多个职员

职员（ENO，ENM）
领导（ENO1，ENO2）
其中：
ENO1：领导者职员号
ENO2：被领导者职员号 （主码）

主码？
多的那一方

6、具有相同码的关系模式可以合并。

*扩展的E-R模型



1. ISA联系

1）分类属性（三角形符号）

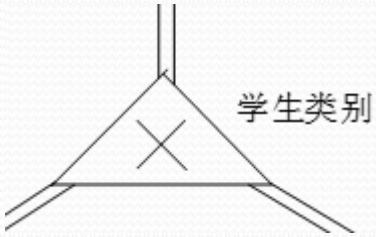
2）不相交约束 /可重叠约束

（三角形符号内部含叉号 /不含叉号）

3）完全特化/部部分特化

完全特化使用双实线连接，表示父类中的一个实体必须是某个子类中的实体。

部分特化用单实线连接。

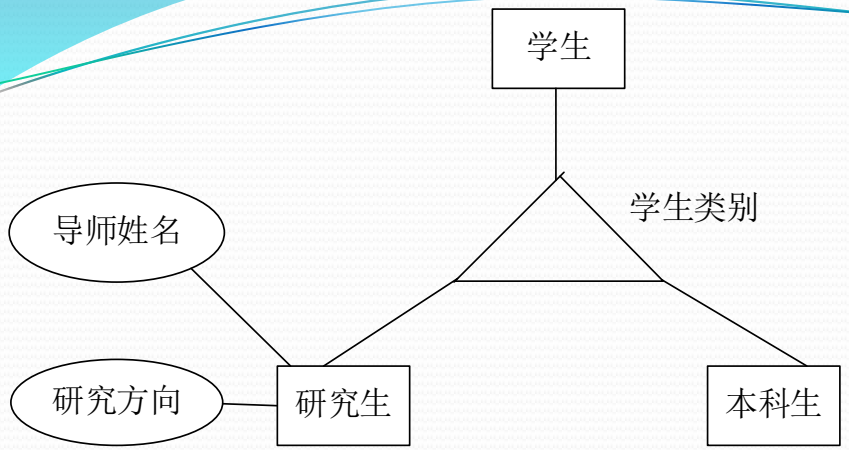


2. 基数约束

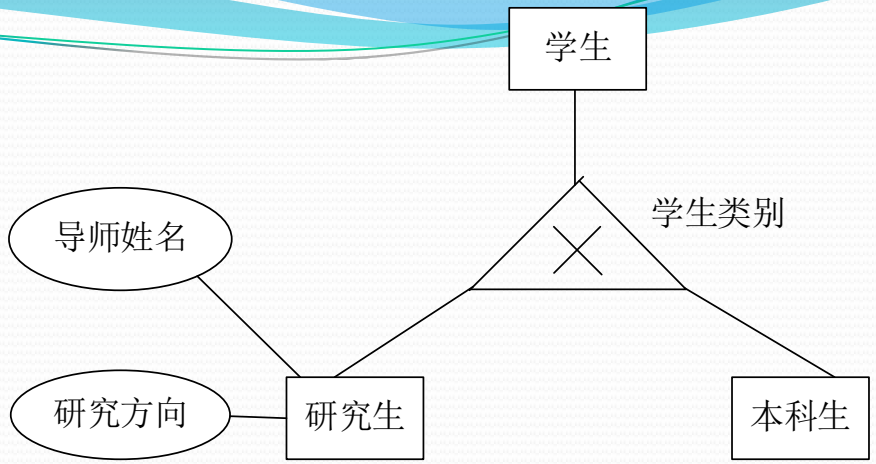
min..max表示，*表示无穷大，

强制参与约束——min=1。

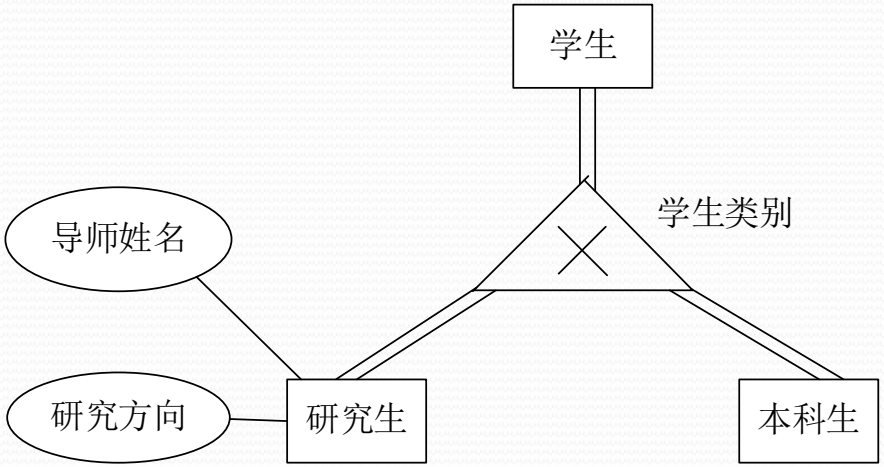
非强制参与约束——min=0。



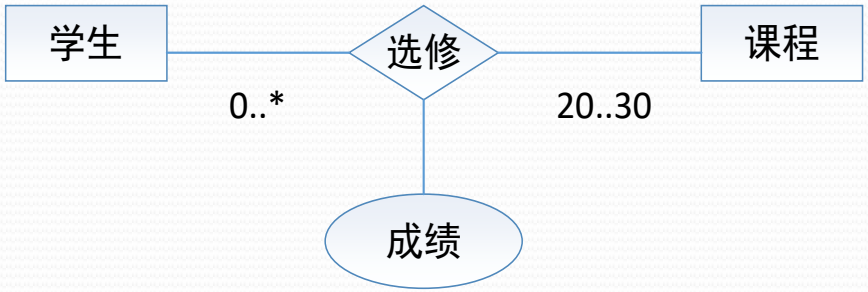
子类和分类属性



不相交约束和可重叠约束



完全特化和部分特化



基数约束min，max

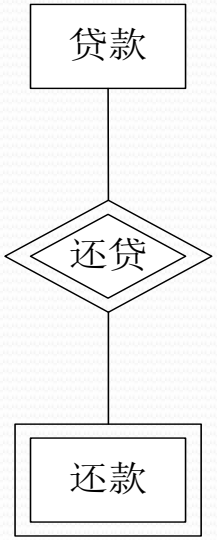
Min=1强制性约束，0非强制性

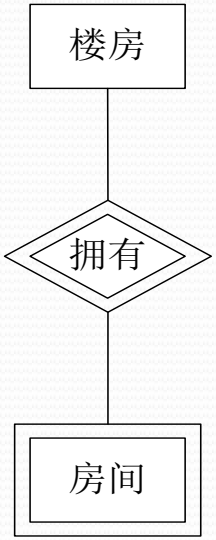
3. Part-of联系

强实体类型

非独占的Part-of联系，部分实体可独立存在。

独占的Part-of联系，部分实体不能独立存在，对应**弱实体型**（双矩形框） 和**识别联系**（双菱形框）。





*使用UML统一建模语言描述E-R图

UML的类大致对应E-R图中的实体。

7.4.4 一般RDMS⇒特定RDMS

1、实体数量、命名；
2、实体内属性数量；
3、属性命名，数据类型，值域限制。

7.4.5 逻辑设计的优化

1、目标：提高速度、节约空间
2、方法：关系规范化理论应用，合并与分解（纵向与横向）。

7.4.6 抽取子模式

根据局部应用需求，结合具体DBMS特点设计用户外模式，即关系DBMS提供的视图。

1．指导思想

注重局部用户的习惯与方便

安全性控制

简化使用、方便维护

具体方法

- 使用更符合用户习惯的别名
- 对不同级别的用户定义不同的视图，以保证系统的安全性
- 简化用户对系统的使用

如某些用户的复杂查询，涉及多个关系模式，可为其定义专门视图，大大地简化用户地使用。

7.5 物理结构设计

7.5.1 功能

1、确定数据库存储结构（存放位置、系统存储配置、。。。。）；
2、确定数据库存取方法。

7.5.2 目标

1、提高速度；2、节约space；3、事务吞吐率大；。。。

7.5.3 存取方法选择

7.5.3.1 存取路径相关

索引，提高速度。

（常用三类索引方法：B+树、聚簇（cluster）、HASH方法，其中，HASH方法适用于等值比较或者等值连接较多的关系，包含静态HASH和动态HASH）

7.5.3.2 建立聚簇

减少I/O，适应单、多个关系集（相当于预连接）。。。。。

可建立聚簇的情况：

1）经常进行连接的关系；

2）单关系的一组属性常出现在等值比较条件中；

3）单关系的属性组的值重复率很高。

聚簇可显著提高ORDER BY、GROUP BY、UNION、DISTINCT等子句的效率，可省去对结果集的排序操作。

聚簇的局限：

1）经常进行全表扫描的关系不适宜使用聚簇；

2）查询属性的更新操作远多于连接操作的关系不适宜使用聚簇；

3）一个关系只能使用一个聚簇。

7.5.3.3 存放位置

1）磁盘/磁带；

2）阵列磁盘；

3）磁盘前位置/后位置。

7.5.3.4 系统配置

参数：用户数，缓冲区大小及个数，页面大小，时间片，……

7.5.3.5 存取方法确定

1）与存储结构有相关性；

2）商用关系DBS透明。

7.6 实现设计

1）定义数据库；
2）初始数据装载；
3）应用程序编制与调试；
4）数据库试运行
修改与完善

7.7 运行与维护

1）安全性、完整性控制
依需求、实际情况设计与调整
2）转储与恢复
通常制定计划
3）性能监测、分析与改进
观察性能参数值

4）数据库重组

物理结构的一些改变，但不涉及逻辑结构。

（例如重新安排存储位置、回收磁盘碎片、减少指针链、创建索引、修改索引类型）

5）数据库重构

逻辑结构的改变→修改模式和内模式。

数据库设计案例-社区疫情防控

疫情防控期间，数据库技术记录社区级防控

疫情期间，某社区有若干小区，每个小区有多个楼栋单元，一个单元有多个家庭住户，每户人家有多位居民。

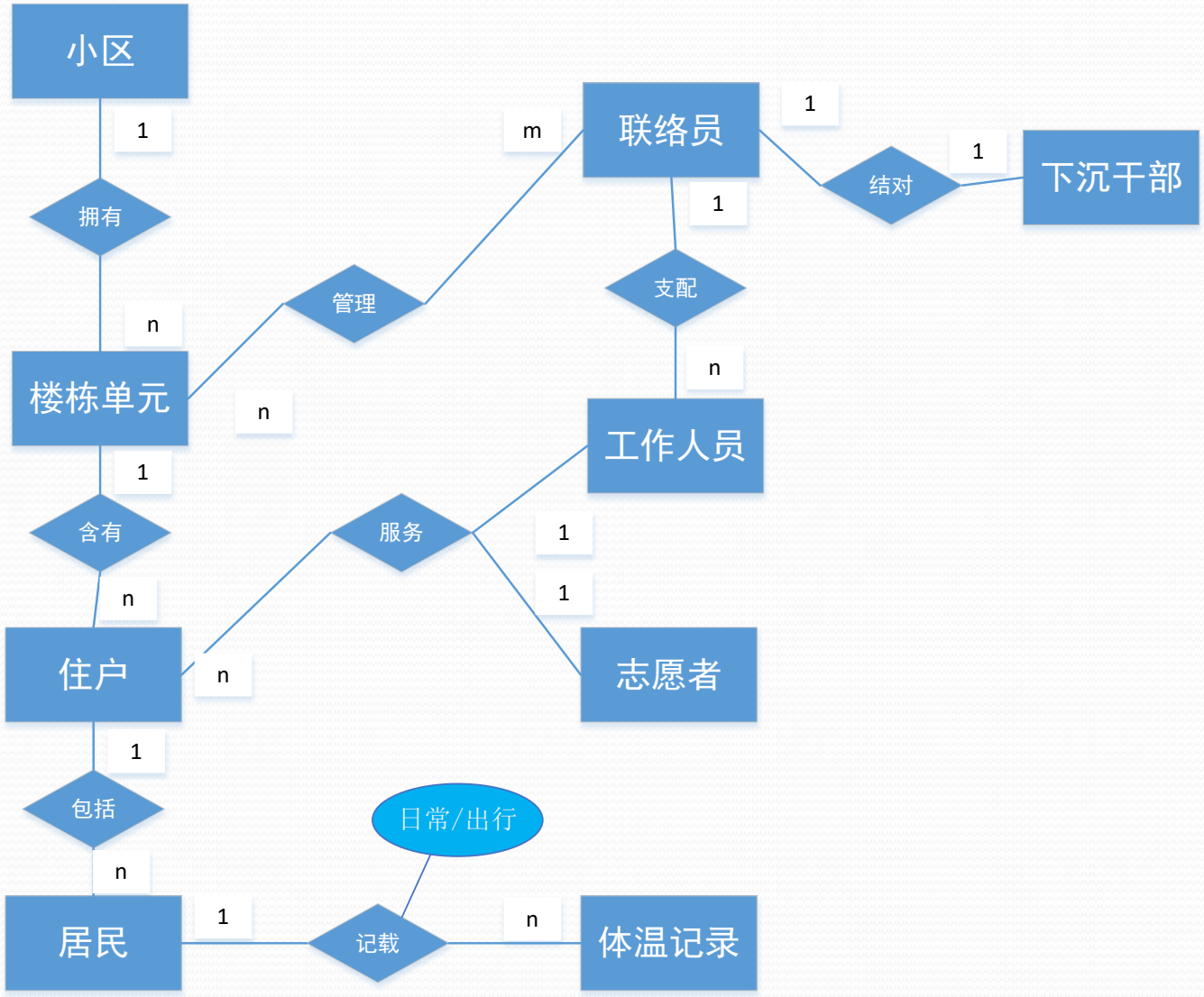
每位居民每天必须上报自己的体温数据，居民出入小区也必须测温并记录在案。

社区派出多名联络员，每名联络员需要负责多个楼栋单元的疫情管控，一个楼栋单元也需要多名联络员分别处理不同疫情事务。

社区接收了若干上级单位的下沉干部，并建立了下沉干部与联络员的一对一工作关系，一名联络员负责某个小区多名工作人员的管理工作。

社区还建立了共产党员志愿者团队，志愿者和小区工作人员采用一对一结对方式为若干住户提供日常生活物资服务。

数据库设计案例-社区疫情防控



数据库设计案例-社区疫情防控

- 小区（小区ID，小区名称，地址，物业电话）
- 楼栋单元（单元ID，单元名称，*小区ID*）
- 住户（住户ID，住户地址，户主姓名，联系电话，*单元ID*，*人员ID*，*志愿者ID*）
- 居民（居民身份证号，姓名，联系电话，*住户ID*）
- 体温记录（记录ID，居民身份证号，记录类型（出入小区、日常测温），记录日期时间）
- 联络员（联络员ID，联络员姓名，联系电话）
- 管理（单元ID，联络员ID）（两个字段同时也为外码）
- 下沉干部（干部ID，干部姓名，*联络员ID*）
- 工作人员（人员ID，服务部门，联系电话，*联络员ID*）
- 志愿者（志愿者ID，志愿者姓名，联系电话）

慕课讨论题

- 如何评价一个E-R图设计的好坏？

E-R图具有较直观的表现形式，是设计人员和需求表分析人员之间沟通的良好工具，如何评价一个E-R图设计的好坏？

- 数据库访问性能与数据库设计之间的问题。

如果一个数据库的访问性能很差，有可能是数据库设计的哪些步骤出了问题？可以考虑哪些优化手段？