# 第七章　数据库设计

# 第七章 数据库设计

# 7.1 数据库设计概述

1、 数据库设计的定义

2、 数据库设计的基本步骤

# 数据库设计概述（续）

## 什么是数据库设计

- 数据库设计是指对于一个给定的应用环境，构造最优的数据库模式，建立数据库及其应用系统，使之能够有效地存储数据，满足各种用户的应用需求（信息要求和处理要求）
- 在数据库领域内，常常把使用数据库的各类系统统称为数据库应用系统。

# 数据库设计人员应该具备的技术和知识

- 数据库的基本知识和数据库设计技术
- 计算机科学的基础知识和程序设计的方法和技巧
- 软件工程的原理和方法
- 应用领域的知识

# 数据库设计的基本步骤

## 一、数据库设计的准备工作

### 选定参加设计的人员

**1.** 数据库分析设计人员

- 数据库设计的核心人员
- 自始至终参与数据库设计
- 其水平决定了数据库系统的质量

# 7.1.4 数据库设计的基本步骤

## 2. 用户

- 在数据库设计中也是举足轻重的
- 主要参加需求分析和数据库的运行维护
- 用户积极参与带来的好处
  - **加速数据库设计**
  - **提高数据库设计的质量**

# 数据库设计的基本步骤（续）

## 3. 程序员

- 在系统实施阶段参与进来，负责编制程序

## 4. 操作员

- 在系统实施阶段参与进来，准备软硬件环境

# 数据库设计的基本步骤（续）

## 二、数据库设计的过程(六个阶段)

### 1.需求分析阶段

- 准确了解与分析用户需求（包括数据与处理）
- 是整个设计过程的基础，是最困难、最耗费时间的一步

# 数据库设计的基本步骤（续）

## 2.概念结构设计阶段

- 是整个数据库设计的关键
- 通过对用户需求进行综合、归纳与抽象，形成一个独立于具体**DBMS**的概念模型

# 数据库设计的基本步骤（续）

## 3.逻辑结构设计阶段

- 将概念结构转换为某个**DBMS**所支持的数据模型
- 对其进行优化

# 数据库设计的基本步骤（续）

## 4.数据库物理设计阶段

- 为逻辑数据模型选取一个最适合应用环境的物理结构（包括存储结构和存取方法）

# 数据库设计的基本步骤（续）

## 5.数据库实施阶段

- 运用DBMS提供的数据语言、工具及宿主语言，根据逻辑设计和物理设计的结果
  - 建立数据库
  - 编制与调试应用程序
  - 组织数据入库
  - 进行试运行

# 数据库设计的基本步骤（续）

## 6.数据库运行和维护阶段

- 数据库应用系统经过试运行后即可投入正式运行。
- 在数据库系统运行过程中必须不断地对其进行评价、调整与修改。

# 数据库设计的基本步骤（续）

设计一个完善的数据库应用系统往往是上述六个阶段的不断反复。

# 第七章 数据库设计

7.1 数据库设计概述

7.2 需求分析

7.3 概念结构设计

7.4 逻辑结构设计

7.5 数据库的物理设计

7.6 数据库实施

7.7 数据库运行与维护

# 需求分析（续）

- **需求分析就是分析用户的需要与要求**
  - 需求分析是设计数据库的起点
  - 需求分析的结果是否准确反映了用户的实际要求，将直接影响到后面各个阶段的设计，并影响到设计结果是否合理和实用

# 一、需求分析的任务

- 通过详细调查现实世界要处理的对象（组织、部门、企业等），充分了解原系统（手工系统或计算机系统）工作概况，明确用户的各种需求
- 在此基础上确定新系统的功能。新系统必须充分考虑今后可能的扩充和改变，不能仅仅按当前应用需求来设计数据库

# 二、需求分析的重点

- 需求分析的重点是调查、收集与分析用户在数据管理中的信息要求、处理要求、安全性与完整性要求。

- 信息要求
  - 用户需要从数据库中获得信息的内容与性质
  - 由用户的信息要求可以导出数据要求，即在数据库中需要存储哪些数据

# 需求分析的重点（续）

- **处理要求**
  - 对处理功能的要求
  - 对处理的响应时间的要求
  - 对处理方式的要求(批处理 / 联机处理)
- **新系统的功能必须能够满足用户的信息要求、处理要求、安全性与完整性要求。**

# 三、需求分析的难点

## 确定用户最终需求的难点

- 用户缺少计算机知识，开始时无法确定计算机究竟能为自己做什么，不能做什么，因此无法一下子准确地表达自己的需求，他们所提出的需求往往不断地变化。
- 设计人员缺少用户的专业知识，不易理解用户的真正需求，甚至误解用户的需求。
- 新的硬件、软件技术的出现也会使用户需求发生变化。

# 需求分析的难点(续)

- **解决方法**
  - 设计人员必须采用有效的方法，与用户不断深入地进行交流，才能逐步得以确定用户的实际需求

# 7.3 概念结构设计

7.3.1 概念结构设计概述

7.3.2 概念结构设计的方法与步骤

7.3.3 数据抽象与局部视图设计

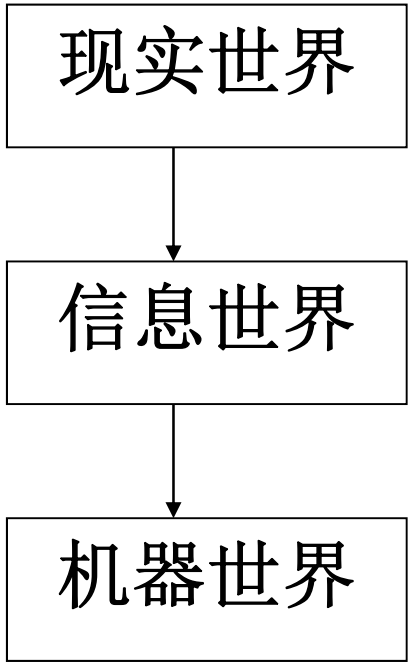
7.3.4 视图的集成

# 7.3.1 概念结构

## 什么是概念结构设计

- 需求分析阶段描述的用户应用需求是现实世界的具体需求
- 将需求分析得到的用户需求抽象为信息结构即概念模型的过程就是概念结构设计

# 概念结构（续）




需求分析

概念结构设计

# 概念结构（续）

- 描述概念模型的工具
  - E-R模型

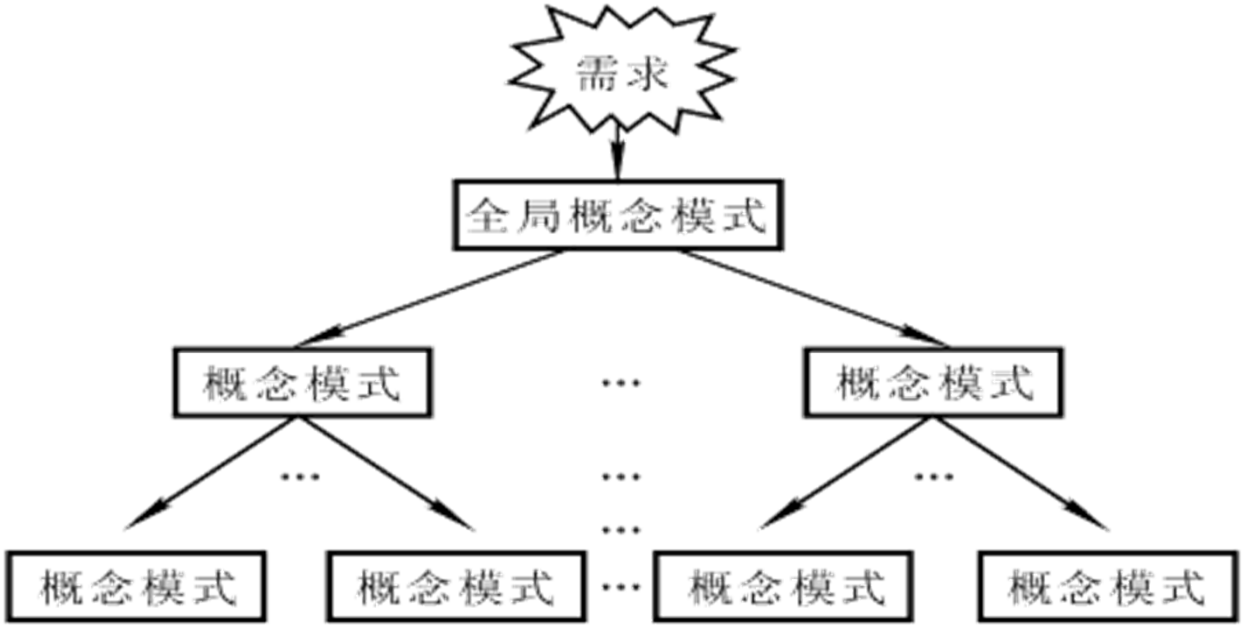
# 7.3.2 概念结构设计的方法与步骤

- 设计概念结构的四类方法
  - 自顶向下
    - 首先定义全局概念结构的框架，然后逐步细化
  - 自底向上
    - 首先定义各局部应用的概念结构，然后将它们集成起来，得到全局概念结构

# 概念结构设计的方法与步骤（续）

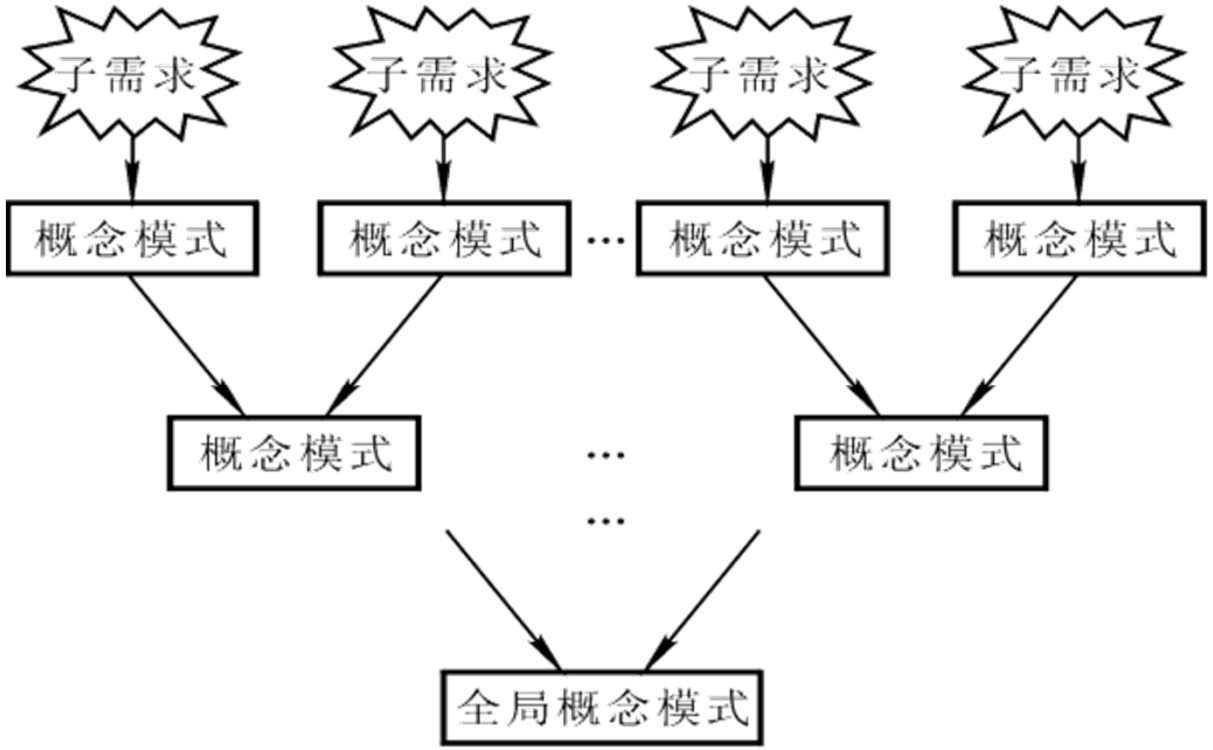
- 逐步扩张 
  - 首先定义最重要的核心概念结构，然后向外扩充，以滚雪球的方式逐步生成其他概念结构，直至总体概念结构
- 混合策略
  - 将自顶向下和自底向上相结合，用自顶向下策略设计一个全局概念结构的框架，以它为骨架集成由自底向上策略中设计的各局部概念结构。

# 概念结构设计的方法与步骤（续）



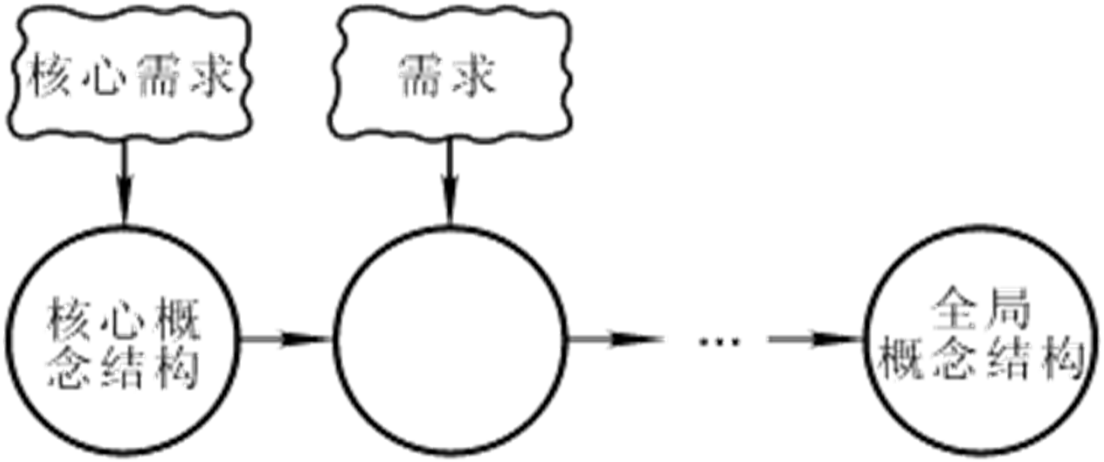

自顶向下策略

# 概念结构设计的方法与步骤（续）




自底向上策略

# 概念结构设计的方法与步骤（续）




逐步扩张

# 概念结构设计的方法与步骤（续）

- 常用策略
  - 自顶向下地进行需求分析
  - 自底向上地设计概念结构

# 7.3 概念结构设计

7.3.1 概念结构设计概述

7.3.2 概念结构设计的方法与步骤

7.3.3 数据抽象与局部视图设计

7.3.4 视图的集成

# 一、数据抽象

- 概念结构是对现实世界的一种抽象
  - 从实际的人、物、事和概念中抽取所关心的共同特性，忽略非本质的细节
  - 把这些特性用各种概念精确地加以描述
  - 这些概念组成了某种模型

# 数据抽象（续）

## 常用抽象

### 1. 分类（Classification）

- 定义某一类概念作为现实世界中一组对象的类型
- 这些对象具有某些共同的特性和行为
- 它抽象了对象值和型之间的“**is member of**”的语义
- 在**E-R**模型中，实体型就是这种抽象

# 数据抽象（续）

## 2. 聚集（Aggregation）

- 定义某一类型的组成成分
- 它抽象了对象内部类型和成分之间“**is part of**”的语义
- 在**E-R**模型中若干属性的聚集组成了实体型，就是这种抽象

# 数据抽象（续）

## 3. 概括（Generalization）

- 定义类型之间的一种子集联系
- 它抽象了类型之间的“**is subset of**”的语义
- 概括有一个很重要的性质：继承性。子类继承超类上定义的所有抽象。

# 数据抽象（续）

## 数据抽象的用途

- 对需求分析阶段收集到的数据进行分类、组织（聚集），形成
  - **实体**
  - **实体的属性，标识实体的码**
  - **确定实体之间的联系类型(1:1，1:n，m:n)**

# 二、局部视图设计

**设计分E-R图的步骤:**

**1.选择局部应用**

**2.逐一设计分E-R图**

# 1. 选择局部应用

- 需求分析阶段，已用多层数据流图和数据字典描述了整个系统。
- 设计分E-R图首先需要根据系统的具体情况，在多层的数据流图中选择一个适当层次的数据流图，让这组图中每一部分对应一个局部应用，然后以这一层次的数据流图为出发点，设计分E-R图。

# 选择局部应用（续）

- **通常以中层数据流图作为设计分E-R图的依据。**

  **原因：**

  - 高层数据流图只能反映系统的概貌
  - 中层数据流图能较好地反映系统中各局部应用的子系统组成
  - 低层数据流图过细

# 选择局部应用（续）

例：由于学籍管理、课程管理等都不太复杂，因此可以从它们入手设计学生管理子系统的分E-R图。

# 2. 逐一设计分E-R图

- 任务
  - 标定局部应用中的实体、属性、码，实体间的联系
    - 将各局部应用涉及的数据分别从数据字典中抽取出来，参照数据流图，标定各局部应用中的实体、实体的属性、标识实体的码，确定实体之间的联系及其类型（1:1，1:n，m:n）

# 逐一设计分E-R图（续）

## 如何抽象实体和属性

- 实体：现实世界中一组具有某些共同特性和行为的对象就可以抽象为一个实体。对象和实体之间是“**is member of"**的关系。

**例：在学校环境中，可把张三、李四等对象抽象为学生实体。**

# 逐一设计分E-R图（续）

- 属性：对象类型的组成成分可以抽象为实体的属性。

  组成成分与对象类型之间是“**is part of"**的关系。

**例：学号、姓名、专业、年级等可以抽象为学生实体的属性。其中学号为标识学生实体的码。**

# 逐一设计分E-R图（续）

## 如何区分实体和属性

- 实体与属性是相对而言的。同一事物，在一种应用环境中作为“属性”，在另一种应用环境中就必须作为“实体”。

例：学校中的系，在某种应用环境中，它只是作为“学生”实体的一个属性，表明一个学生属于哪个系；而在另一种环境中，由于需要考虑一个系的系主任、教师人数、学生人数、办公地点等，这时它就需要作为实体了。

# 逐一设计分E-R图（续）

- 一般原则
  - 属性不能再具有需要描述的性质。即属性必须是不可分的数据项，不能再由另一些属性组成。
  - 属性不能与其他实体具有联系。联系只发生在实体之间。
- 符合上述两条特性的事物一般作为属性对待。
- 为了简化E-R图的处置，现实世界中的事物凡能够作为属性对待的，应尽量作为属性。

# 逐一设计分E-R图（续）

- 举例

例1：“学生”由学号、姓名等属性进一步描述，根据准则 1 ，“学生”只能作为实体，不能作为属性。

例2：职称通常作为教师实体的属性，但在涉及住房分配时，由于分房与职称有关，也就是说职称与住房实体之间有联系，根据准则 2 ，这时把职称作为实体来处理会更合适些。

# 逐一设计分E-R图（续）

## 设计分E-R图的步骤

- （1）以数据字典为出发点定义E-R图。
  - 数据字典中的“数据结构”、“数据流”和“数据存储”等已是若干属性的有意义的聚合
- （2）按上面给出的准则进行必要的调整。

# 逐一设计分E-R图（续）

**例：学籍管理局部应用中主要涉及的实体包括学生、宿舍、档案材料、班级、班主任。**

**实体之间的联系：**

- 由于一个宿舍可以住多个学生，而一个学生只能住在某一个宿舍中，因此宿舍与学生之间是**1:n**的联系。
- 由于一个班级往往有若干名学生，而一个学生只能属于一个班级，因此班级与学生之间也是**1:n**的联系。

# 逐一设计分E-R图（续）

- 由于班主任同时还要教课，因此班主任与学生之间存在指导联系，一个班主任要教多名学生，而一个学生只对应一个班主任，因此班主任与学生之间也是**1:n**的联系。

- 而学生和他自己的档案材料之间，班级与班主任之间都是**1:1**的联系。

# 逐一设计分E-R图（续）

**接下来需要进一步斟酌该E-R图，做适当调整。**

- 在一般情况下，性别通常作为学生实体的属性，但在本局部应用中，由于宿舍分配与学生性别有关，根据准则 2，应该把性别作为实体对待。

# 逐一设计分E-R图（续）

该E-R图中省略了各个实体的属性描述：

学生：｛学号，姓名，出生日期，性别｝

性别：｛性别｝

档案材料：｛档案号，……｝

班级：｛班级号，学生人数｝

班主任：｛职工号，姓名，性别，是否为优秀班主任｝

宿舍：｛宿舍编号，地址，人数｝

其中有下划线的属性为实体的码。

# 7.3 概念结构设计

7.3.1 概念结构

7.3.2 概念结构设计的方法与步骤

7.3.3 数据抽象与局部视图设计

7.3.4 视图的集成

# 7.3.4 视图的集成

- 各个局部视图即分E-R图建立好后，还需要对它们进行合并，集成为一个整体的数据概念结构即总E-R图。

# 视图的集成（续）

- 视图集成的两种方式
  - 一次集成
    - 一次集成多个分E-R图
    - 通常用于局部视图比较简单时
  - 逐步累积式
    - 首先集成两个局部视图（通常是比较关键的两个局部视图）
    - 以后每次将一个新的局部视图集成进来

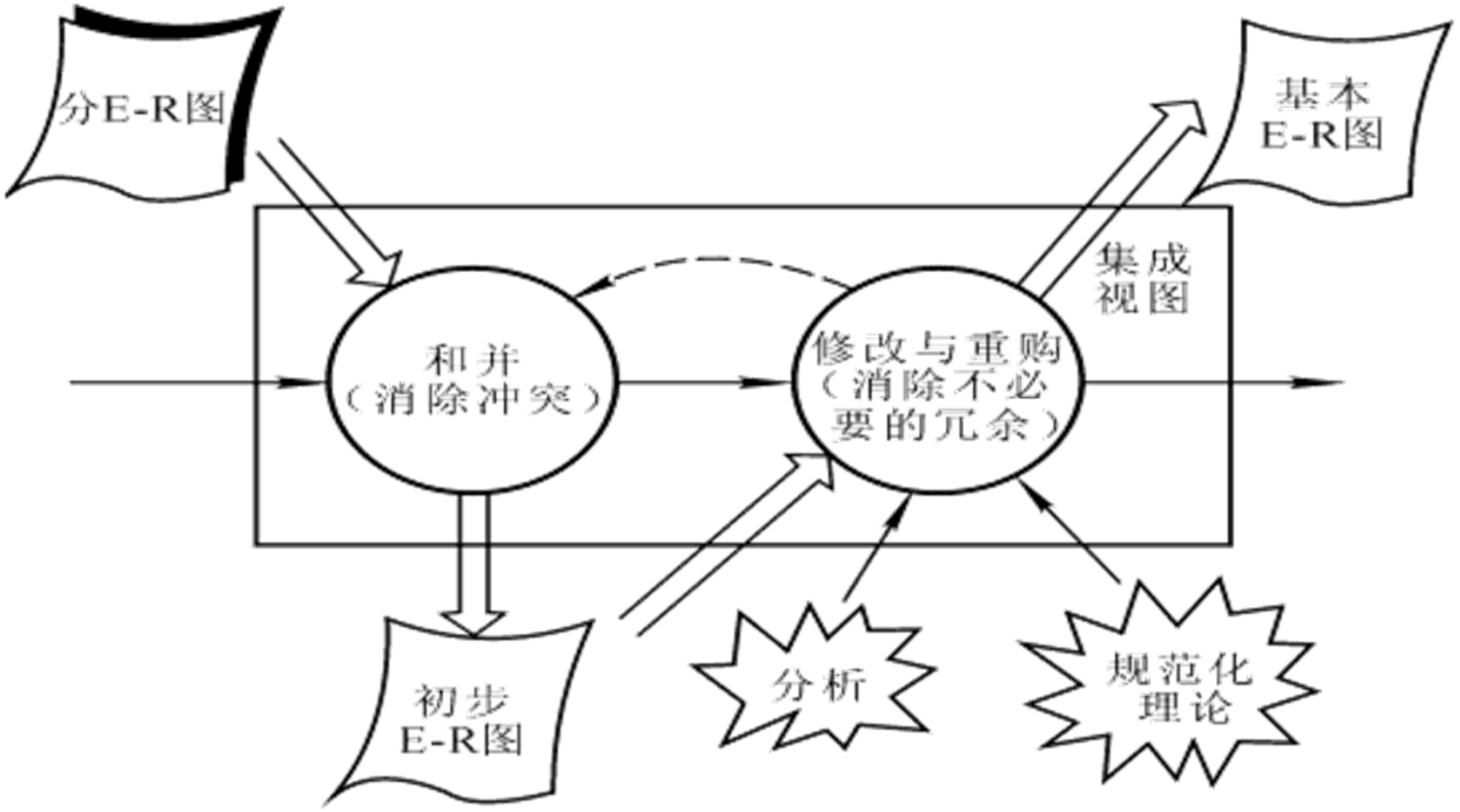
# 视图的集成（续）

- **集成局部E-R图的步骤**

  **1.** 合并

  **2.** 修改与重构

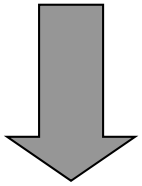
# 视图的集成（续）



# 一、合并分E-R图，生成初步E-R图

- **各分E－R图存在冲突**
  - 各个局部应用所面向的问题不同

    由不同的设计人员进行设计



    各个分**E-R**图之间必定会存在许多不一致的地方
  - 合并分**E-R**图的主要工作与关键所在：合理消除各分**E-R**图的冲突

# 合并分E-R图，生成初步E-R图（续）

- **冲突的种类**
  - 属性冲突
  - 命名冲突
  - 结构冲突

# 1. 属性冲突

- 两类属性冲突
  - 属性域冲突：属性值的类型、取值范围或取值集合不同。

**例1，** 由于学号是数字，因此某些部门（即局部应用）将学号定义为整数形式，而由于学号不用参与运算，因此另一些部门（即局部应用）将学号定义为字符型形式。

**例2，** 某些部门（即局部应用）以出生日期形式表示学生的年龄，而另一些部门（即局部应用）用整数形式表示学生的年龄。

# 属性冲突（续）

- 属性取值单位冲突。

例：学生的身高，有的以米为单位，有的以厘米为单位，有的以尺为单位。

# 属性冲突（续）

- **属性冲突的解决方法**
  - 通常用讨论、协商等行政手段加以解决

# 2. 命名冲突

- 两类命名冲突
  - 同名异义：不同意义的对象在不同的局部应用中具有相同的名字

    例，局部应用A中将教室称为房间

    局部应用B中将学生宿舍称为房间
  - 异名同义（一义多名）：同一意义的对象在不同的局部应用中具有不同的名字

    例，有的部门把教科书称为课本

    有的部门则把教科书称为教材

# 命名冲突（续）

- 命名冲突可能发生在属性级、实体级、联系级上。其中属性的命名冲突更为常见。

- 命名冲突的解决方法
  - 通过讨论、协商等行政手段加以解决

# 3. 结构冲突

## 三类结构冲突

- 同一对象在不同应用中具有不同的抽象

例，“课程”在某一局部应用中被当作实体

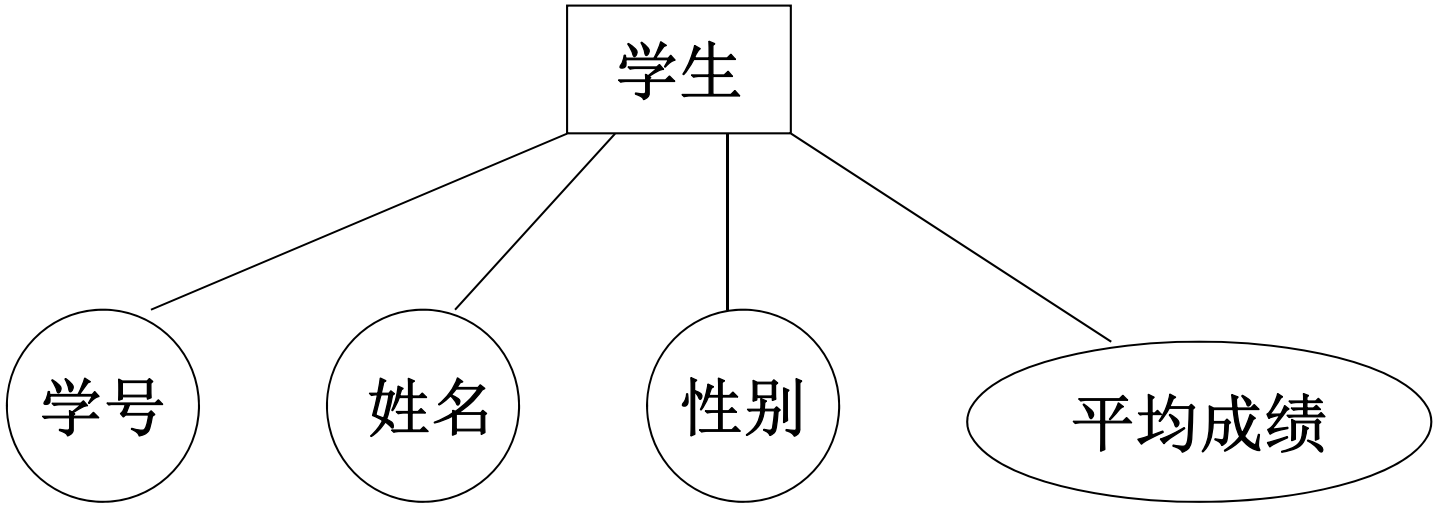
在另一局部应用中则被当作属性

- **解决方法：通常是把属性变换为实体或把实体变换为属性，使同一对象具有相同的抽象。变换时要遵循两个准则。**

# 结构冲突（续）

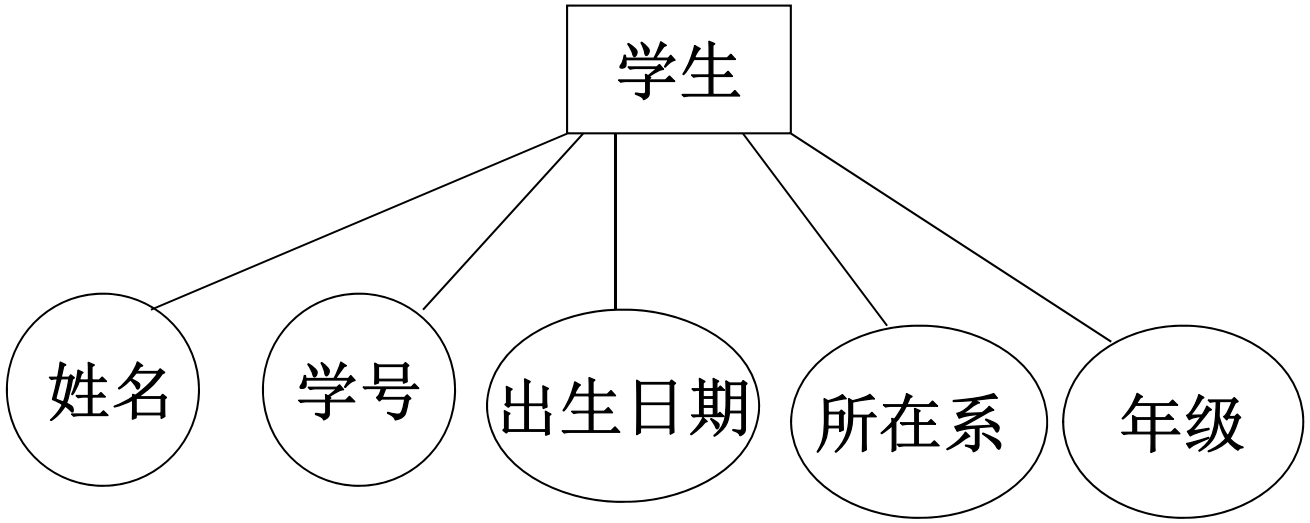
- 同一实体在不同局部视图中所包含的属性不完全相同，或者属性的排列次序不完全相同。
  - 产生原因：不同的局部应用关心的是该实体的不同侧面。
  - 解决方法：使该实体的属性取各分E-R图中属性的并集，再适当设计属性的次序。

# 结构冲突（续）



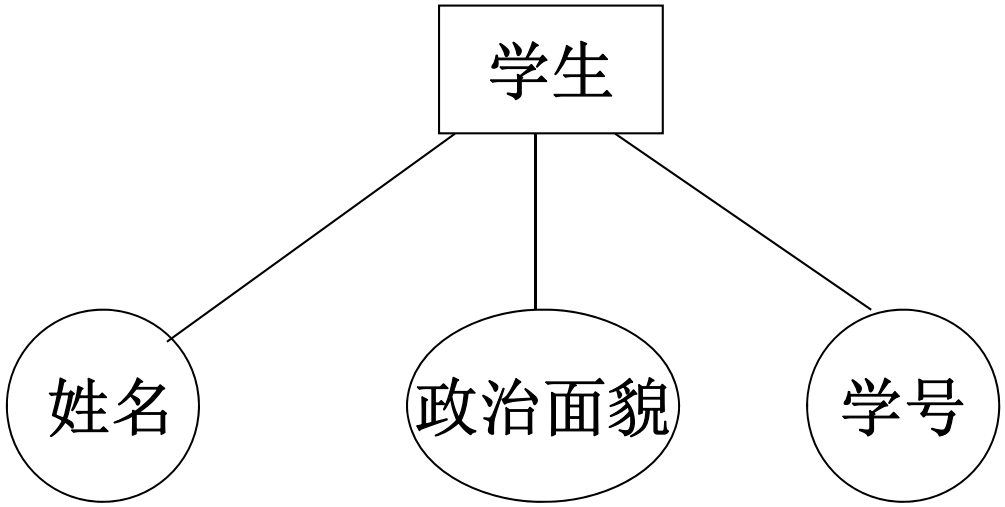

**(a)在局部应用A中**

# 结构冲突（续）



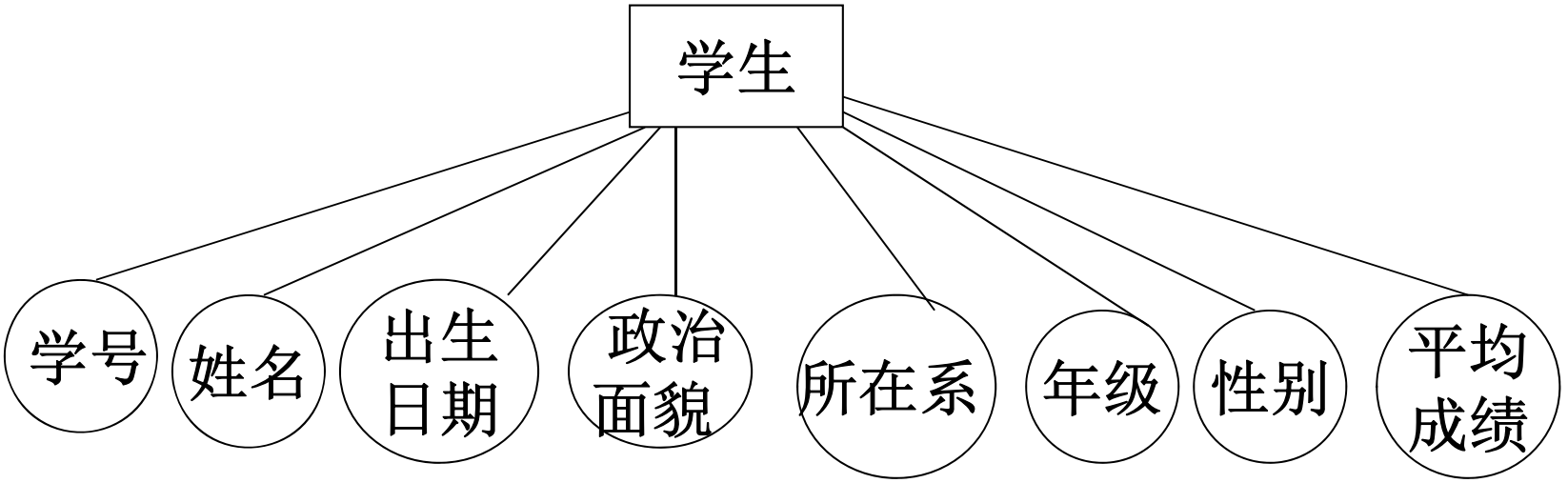

**(b)在局部应用B中**

# 结构冲突（续）




**(c)在局部应用C中**

# 结构冲突（续）




**(d)合并后**

# 结构冲突（续）

- 实体之间的联系在不同局部视图中呈现不同的类型

例1， 实体E1与E2在局部应用A中是多对多联系，而在局部应用B中是一对多联系

例2， 在局部应用X中E1与E2发生联系，而在局部应用Y中E1、E2、E3三者之间有联系。

解决方法：根据应用语义对实体联系的类型进行综合或调整。

# 合并分E-R图，生成初步E-R图实例

例：生成学校管理系统的初步E-R图

以合并学籍管理局部视图，课程管理局部视图为例

这两个分E-R图存在着多方面的冲突：

# 合并分E-R图，生成初步E-R图实例

（1） 班主任实际上也属于教师，也就是说学籍管理中的班主任实体与课程管理中的教师实体在一定程度上属于异名同义，可以将学籍管理中的班主任实体与课程管理中的教师实体统一称为教师，统一后教师实体的属性构成为：

教师：｛职工号，姓名，性别，职称，

是否为优秀班主任｝

# 合并分E-R图，生成初步E-R图实例（续）

(2) 将班主任改为教师后，教师与学生之间的联系在两个局部视图中呈现两种不同的类型，一种是学籍管理中教师与学生之间的指导联系，一种是课程管理中教师与学生之间的教学联系，由于指导联系实际上可以包含在教学联系之中，因此可以将这两种联系综合为教学联系。

# 合并分E-R图，生成初步E-R图实例（续）

(3) 性别在两个局部应用中具有不同的抽象，它在学籍管理中为实体，在课程管理中为属性，按照前面提到的两个原则，在合并后的E-R图中性别只能作为实体，否则它无法与宿舍实体发生联系。

# 合并分E-R图，生成初步E-R图实例（续）

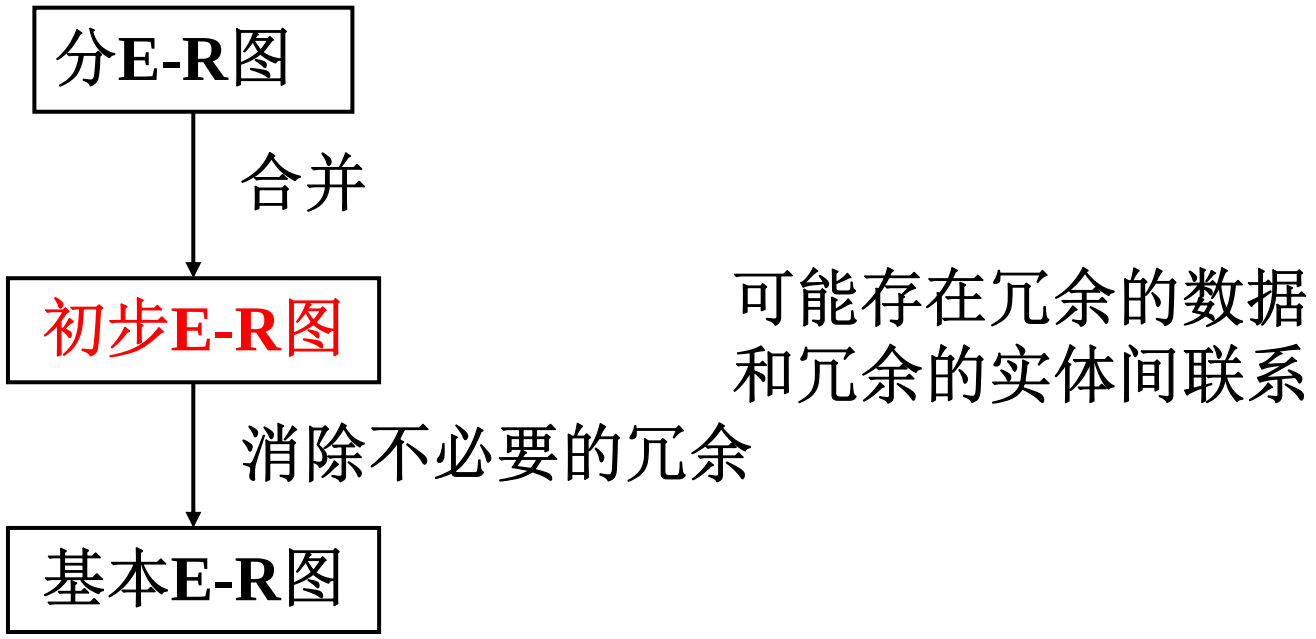
(4) 在两个局部E-R图中，学生实体属性组成及次序都存在差异，应将所有属性综合，并重新调整次序。假设调整结果为：

学生：｛学号，姓名，出生日期，年龄，所在系，年级，平均成绩｝

# 二、修改与重构

## 基本任务

- 消除不必要的冗余，设计生成基本E-R图



# 冗余

- 冗余的数据是指可由基本数据导出的数据，

  冗余的联系是指可由其他联系导出的联系。

- 冗余数据和冗余联系容易破坏数据库的完整性，给数据库维护增加困难

- 并不是所有的冗余数据与冗余联系都必须加以消除，有时为了提高某些应用的效率，不得不以冗余信息作为代价。

# 冗余（续）

- 设计数据库概念结构时，哪些冗余信息必须消除，哪些冗余信息允许存在，需要根据用户的整体需求来确定。

- 消除不必要的冗余后的初步E-R图称为基本E-R图。

# 消除冗余的方法

- 分析方法
  - 以数据字典和数据流图为依据，根据数据字典中关于数据项之间逻辑关系的说明来消除冗余。

# 消除冗余的方法(续)

例，教师工资单中包括该教师的基本工资、各种补贴、应扣除的房租水电费以及实发工资。

由于实发工资可以由前面各项推算出来，因此可以去掉，在需要查询实发工资时根据基本工资、各种补贴、应扣除的房租水电费数据临时生成。

# 消除冗余，设计生成基本E-R图实例

例学生管理子系统中存在着冗余数据和冗余联系：

(1) 学生实体中的年龄属性可以由出生日期推算出来，属于冗余数据，应该去掉。这样不仅可以节省存储空间，而且当某个学生的出生日期有误，进行修改后，无须相应修改年龄，减少了产生数据不一致的机会。

学生：｛学号，姓名，出生日期，所在系，

年级，平均成绩｝

# 消除冗余，设计生成基本E-R图实例（续）

**(2) 学生实体中的平均成绩可以从选修联系中的成绩属性中推算出来**

- 由于应用中需要经常查询某个学生的平均成绩，每次都进行这种计算效率就会太低，因此为提高效率，保留该冗余数据
- 定义一个触发器来保证学生的平均成绩等于该学生各科成绩的平均值。
- 任何一科成绩修改后，或该学生学了新的科目并有成绩后，就触发该触发器去修改该学生的平均成绩属性值。

# 消除冗余的方法（续）

- 一种更好的方法是把冗余数据定义在视图中

# 消除冗余的方法（续）

- **规范化理论**
  - 函数依赖的概念提供了消除冗余联系的形式化工具

# 消除冗余的方法（续）

- 方法

1. 确定分E-R图实体之间的数据依赖$F_L$。实体之间一对一、一对多、多对多的联系可以用实体码之间的函数依赖来表示。

例：

班级和学生之间一对多的联系：

学号→班级号

学生和课程之间多对多的联系：

（学号，课程号）→成绩

# 消除冗余的方法（续）

**2.** 求$F_L$的最小覆盖$G_L$，差集为

$$D = F_L - G_L。$$

逐一考察$D$中的函数依赖，确定是否是冗余的联系，若是，就把它去掉。

# 消除冗余的方法（续）

- 由于规范化理论受到泛关系假设的限制，应注意下面两个问题：

1.冗余的联系一定在D中，而D中的联系不一定是冗余的；

2.当实体之间存在多种联系时要将实体之间的联系在形式上加以区分。

**例** 部门和职工之间两种联系表示为：

职工号→部门号（职工属于某部门）

部门号→负责人.职工号，负责人.职工号→部门号（职工领导某部门）

# 验证整体概念结构

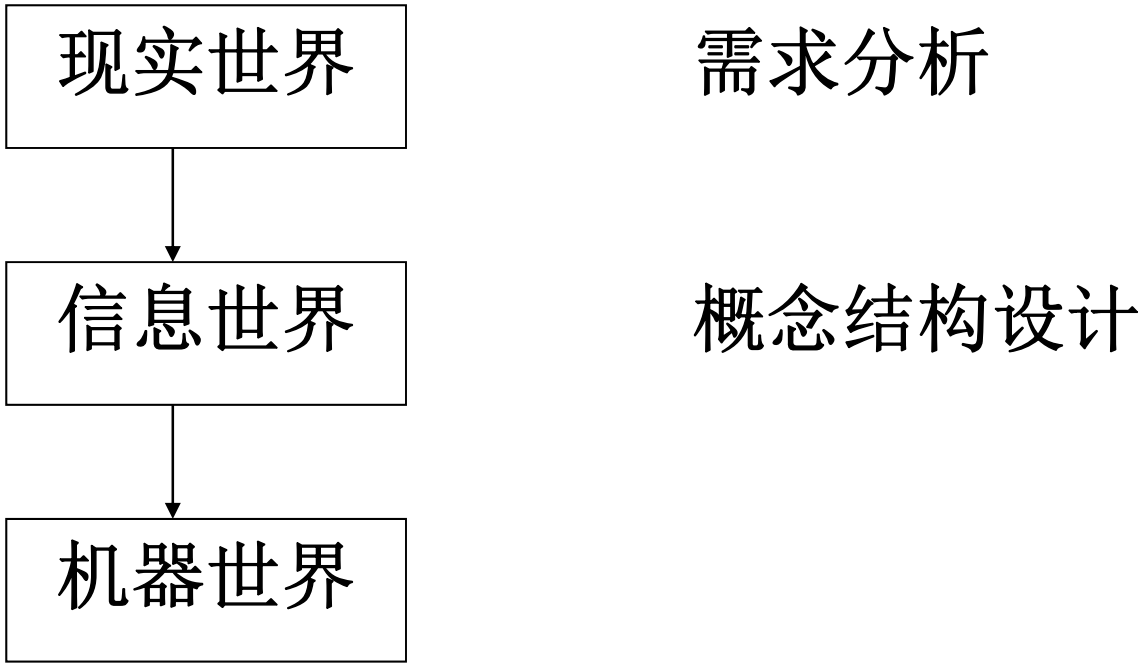
- 视图集成后形成一个整体的数据库概念结构，对该整体概念结构还必须进行进一步验证，确保它能够满足下列条件：
  - 整体概念结构内部必须具有一致性，不存在互相矛盾的表达。
  - 整体概念结构能准确地反映原来的每个视图结构，包括属性、实体及实体间的联系。
  - 整体概念结构能满足需要分析阶段所确定的所有要求。

# 验证整体概念结构（续）

- 整体概念结构最终还应该提交给用户，征求用户和有关人员的意见，进行评审、修改和优化，然后把它确定下来，作为数据库的概念结构，作为进一步设计数据库的依据。

# 概念结构设计小结

- 什么是概念结构设计



# 概念结构设计小结

- **概念结构设计的步骤**
  - 抽象数据并设计局部视图
  - 集成局部视图，得到全局概念结构
  - 验证整体概念结构

# 概念结构设计小结

- **数据抽象**
  - 分类
  - 聚集

# 概念结构设计小结

- 设计局部视图
  - 1. 选择局部应用
  - 2. 逐一设计分E-R图
    - 标定局部应用中的实体、属性、码，实体间的联系
    - 用E-R图描述出来

# 概念结构设计小结

## 集成局部视图

- 1.合并分E-R图，生成初步E-R图
  - 消除冲突
    - 属性冲突
    - 命名冲突
    - 结构冲突
- 2. 修改与重构
  - 消除不必要的冗余，设计生成基本E-R图
    - 分析方法
    - 规范化理论