第6章 数据库设计

本章要点

数据库设计的目标就是根据特定的用户需求及一定的计算机软硬件环境，设计并优化数据库的逻辑结构和物理结构，建立高效、安全的数据库，为数据库应用系统的开发和运行提供良好的平台。

数据库技术是研究如何对数据进行统一，有效地组织、管理和加工处理的计算机技术，该技术已应用于社会方方面面，大到一个国家的信息中心，小到个体私人小企业，都会利用数据库技术对数据进行有效的管理，达到提高生产效率和决策水平。目前，一个国家的数据库建设规模（指数据库的个数、种类）、数据库的信息量的大小和使用频度已成为衡量这个国家信息化程度的重要标志之一。

第6章 数据库设计

本章要点

本章详细的介绍了设计一个数据库应用系统需经历的六个阶段，即需求分析、概念设计、逻辑结构设计、物理结构设计、实施与运行维护。其中概念结构设计和逻辑结构设计是本章的重点，也是掌握本章的难点所在。

本章逻辑结构

本章逻辑结构

本章逻辑结构

6.1 数据库设计概述

6.1.1 数据库设计的任务、内容和特点

1、数据库设计的任务

数据库设计是指根据用户需求研制数据库结构并应用的过程。具体的说，数据库设计是指对于给定的应用环境，构造最优的数据库模式，建立数据库及其应用系统，使之能有效地存储数据，满足用户的信息要求和处理要求，也就是把现实世界中的数据，根据各种应用处理的要求，加以合理组织，使之能满足硬件和操作系统的特性，利用已有的DBMS来建立能够实现系统目标的数据库。

2、数据库设计的内容

6.1.1 数据库设计的任务、内容和特点

(1) 数据库的结构设计是指根据给定的应用环境，进 行数据库的模式设计或子模式的设计。它包括数据库的概念设计、逻辑设计和物理设计。

(2)数据库的行为设计是指数据库用户的行为和动作。在数据库系统中，用户的行为和动作指用户对数据库的操作，这些要通过应用程序来实现，所以数据库的行为设计就是操作数据库的应用程序的设计。即设计应用程序、事务处理等，所以行为设计是动态的，行为设计又称为动态模式设计。

3、数据库设计的特点

(1) 数据库建设是硬件、软件和干件（技术和管理的界面）的结合

6.1.1 数据库设计的任务、内容和特点

(2) 数据库设计应该与应用系统设计相结合,也就是说要把行为设计和结构设计密切结合起来是一种“反复探寻，逐步求精的过程”。

6.1.2 数据库设计方法简述

比较著名的有新奥尔良（New Orleans）法，它是目前公认的比较完整和权威的一种规范设计法，它将数据库设计分为四个阶段：需求分析（分析用户的需求）、概念设计（信息分析和定义）、逻辑设计（设计的实现）和物理设计（物理数据库设计），其后，S.B.Yao等又将数据库设计分为五个步骤。目前大多数设计方法都起源于新奥尔良法，并在设计的每个阶段采用一些辅助方法来具体实现，下面简单介绍几种比较有影响的设计方法。

1、基于E-R模型的数据库设计方法

E-R方法的基本步骤是：① 确定实体类型；② 确定实体联系；③ 画出E-R图；④ 确定属性；⑤ 将E-R图转换成某个DBMS可接受的逻辑数据模型；⑥ 设计记录格式。

6.1.2 数据库设计方法简述

2、基于3NF的数据库设计方法

基于3NF的数据库设计方法的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与属性之间的依赖关系，将它们组织一个单一的关系模式中，然后再将其投影分解，消除其中不符合3NF的约束条件，把其规范成若干个3NF关系模式的集合。

3、计算机辅助数据库设计方法

计算机辅助数据库设计主要分为，需求分析、逻辑结构设计、物理结构设计几个步骤。设计中，哪些可在计算机辅助下进行？能否实现全自动化设计呢？这是计算机辅助数据库设计需要研究的课题。

6.1.3 数据库设计的步骤

按照规范化的设计方法，以及数据库应用系统开发过程，数据库的设计过程可分为以下六个设计阶段（如图6.1）需求分析、概念结构设计、逻辑结构设计、物理结构设计、数据库的实施、数据库运行和维护。

数据库设计中，前两个阶段是面向用户的应用要求，面向具体的问题，中间两个阶段是面向数据库管理系统，最后两个阶段是面向具体的实现方法。前四个阶段可统称为“分析和设计阶段”，后面两个阶段统称为“实现和运行阶段”。按照规范化的设计方法，以及数据库应用系统开发过程，数据库的设计过程可分为以下六个设计阶段（如图6.1）需求分析、概念结构设计、逻辑结构设计、物理结构设计、数据库的实施、数据库运行和维护。

6.1.3 数据库设计的步骤

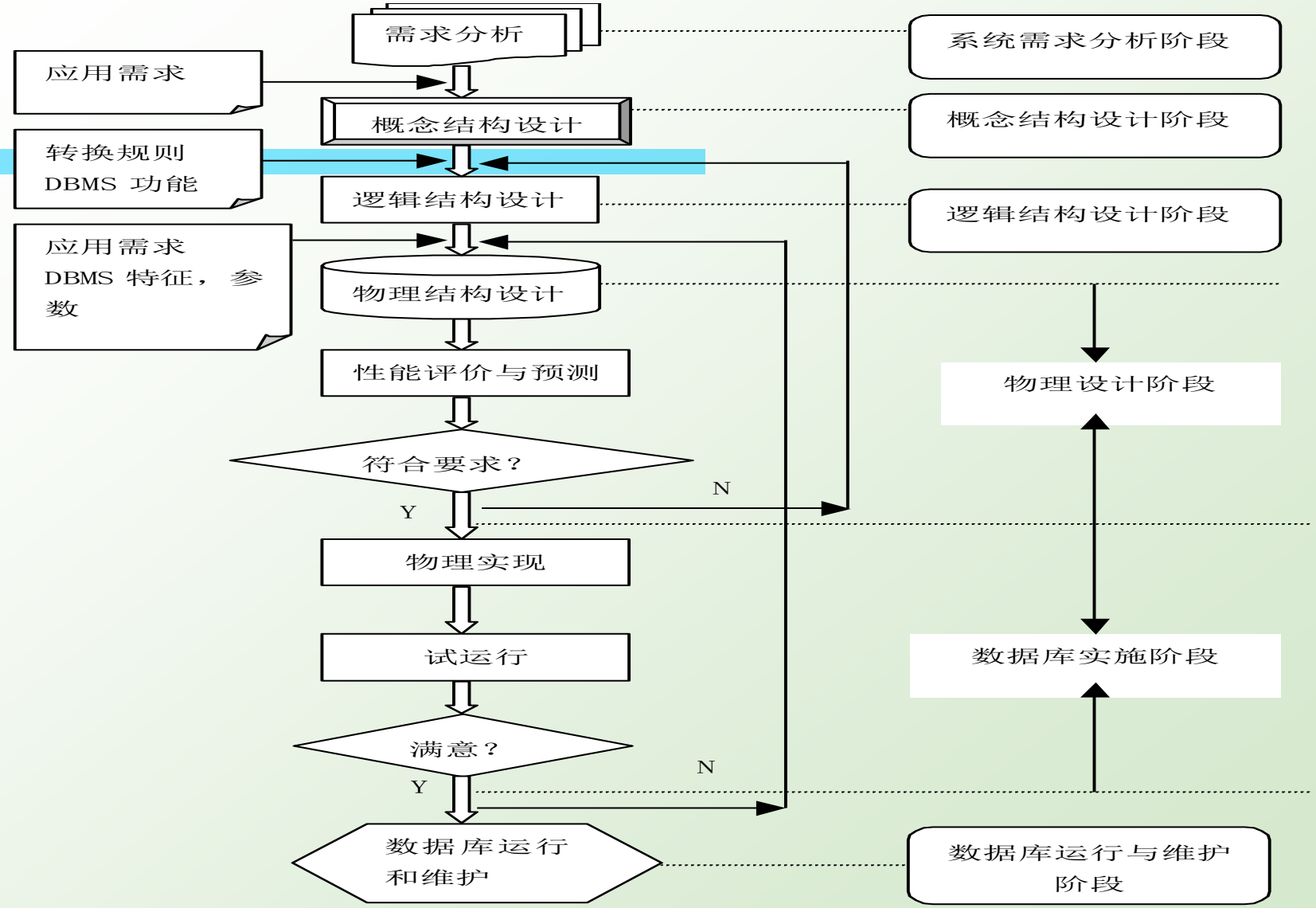


图6.1数据库设计步骤

返回本节首页

6.1.3 数据库设计的步骤

以下是数据库设计六个步骤的具体内容：

1、需求分析阶段

需求分析是指准确了解和分析用户的需求，这是最困难、最费时、最复杂的一步，但也是最重要的一步。它决定了以后各步设计的速度和质量。

2、概念结构设计阶段

概念结构设计是指对用户的需求进行综合、归纳与抽象，形成一个独立于具体DBMS的概念模型，是整个数据库设计的关键。

3、逻辑结构设计阶段

逻辑结构设计是指将概念模型转换成某个DBMS所支持的数据模型，并对其进行优化。

6.1.3 数据库设计的步骤

4、物理设计阶段

物理设计是指为逻辑数据模型选取一个最适合应用环境的物理结构（包括存储结构和存储方法）。

5、数据库实施阶段

数据库实施是指建立数据库，编制与调试应用程序，组织数据入库，并进行试运行。

6、数据库运行与维护阶段

数据库运行与维护是指对数据库系统实际正常运行使用，并时时进行评价、调整与修改。

6.1.3 数据库设计的步骤

表6.1 数据库各个设计阶段的描述

设计各阶段	设计描述	
	数据	处理
需求分析	数据字典，全系统中数据项、数据流 、数据存储的描述	数据流图、判定表（或判定树）、数据字典中处理过程的描述
概念结构设计	概念模型 数据字典	系统说明书。包括： （1）新系统要求、方案和概图 （2）反映新系统信息的数据流图
逻辑结构设计	某种数据模型 关系模型	系统结构图 模块结构图
物理设计	存储安排 存取方法选择 存取路径建立	模块设计 IPO 表
实施阶段	编写模式 装入数据 数据库试运行	程序编码 编译联结 测试
运行维护	性能测试，转储/恢复数据库 重组和重构	新旧系统转换、运行、维护（修正性、适应性、改善性维护）

6.1.3 数据库设计的步骤

可以看出，设计一个数据库不可能一蹴而就的，它往往是上述各个阶段的不断反复。以上六个阶段是从数据库应用系统设计和开发的全过程来考察数据库设计的问题。因此，它既是数据库也是应用系统的设计过程。在设计过程中，努力使数据库设计和系统其他部分的设计紧密结合，把数据和处理的需求收集，分析，抽象，设计和实现在各个阶段同时进行、相互参照、相互补充，以完善两个方面的设计。按照这个原则，数据库各个阶段的设计可用图6.2描述。

6.1.3 数据库设计的步骤

如图6.2所示:需求分析阶段，综合各个用户的应用需求；在概念设计阶段形成独立于机器特点，独立于各个DBMS产品的概念模型，在本书中就是E-R图；在逻辑设计阶段将E-R图转换成具体的数据库产品支持的数据模型，如关系模型中的关系模式；然后根据用户处理的要求、安全性完整性要求等，在基本表的基础上再建立必要的视图（可认为是外模式或子模式）；在物理设计阶段，根据DBMS特点和处理性能等的需要，进行物理设计（如存储安排、建立索引等），形成数据库内模式；实施阶段开发设计人员基于外模式，进行系统功能模块的编码与调试；设计成功的话就进入系统的运行与维护阶段。

6.1.3 数据库设计的步骤

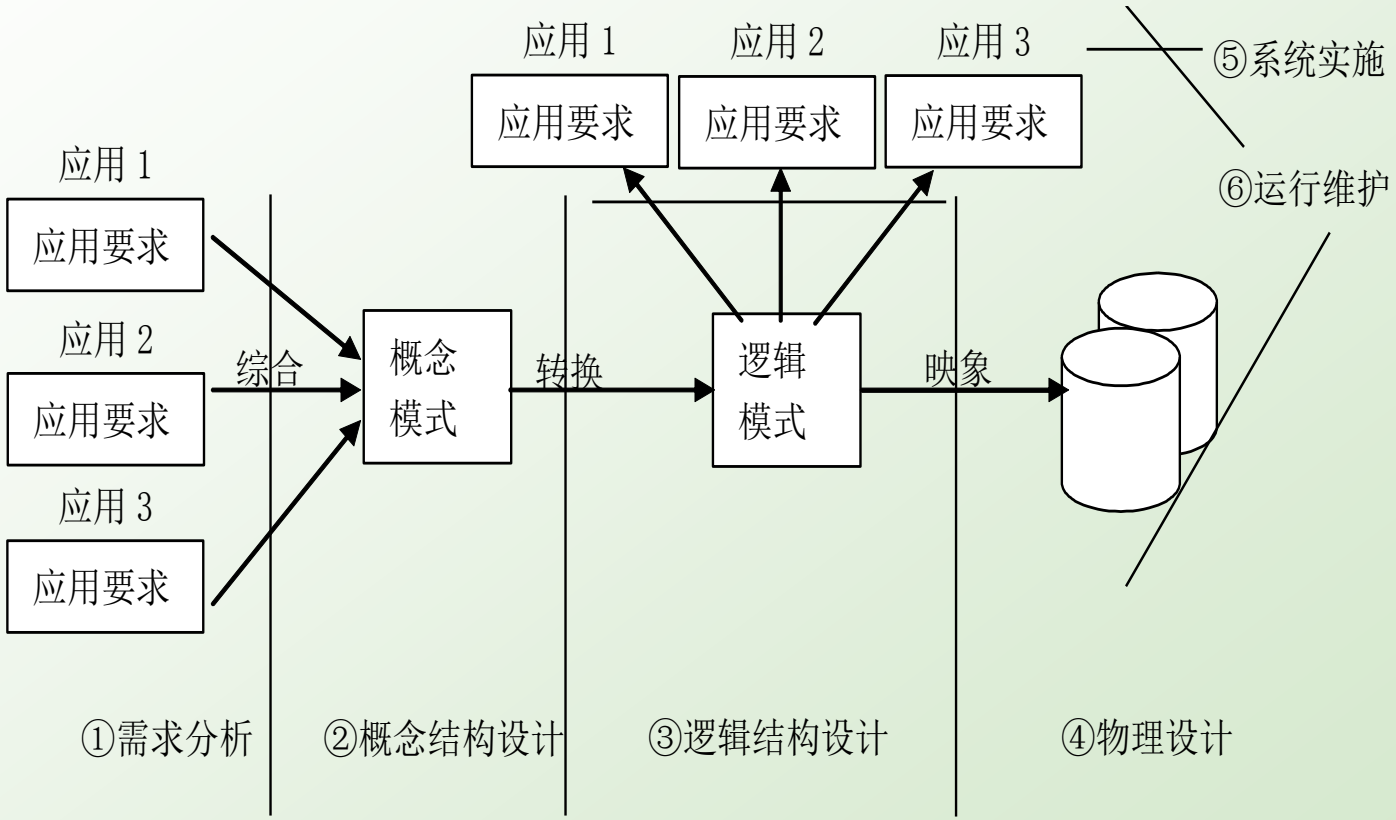


图6.2 数据库设计过程与数据库各级模式

6.2 系统需求分析

需求分析简单的说是分析用户的要求，需求分析是设计数据库的起点，需求分析的结果是否准确地反映了用户的实际需求，将直接影响到后面的各个阶段的设计，并影响到设计结果是否合理与实用。也就是说如果这一步走得不对，获取的信息或分析结果就有误，那么后面的各步设计即使再优化也只能前功尽弃。因此，必须高度重视系统的需求分析。

6.2.1系统需求分析

需求分析的任务

需求分析的任务是通过详细调查现实世界要处理的对象，通过充分对原系统的工作概况的了解，明确用户的各种需求，然后在此基础上确定新系统的功能.

数据库需求分析的任务主要包括“数据或信息”和“处理”两个方面：

（1）信息要求：指用户需要从数据库中获得信息的内容与性质。由信息要求可以导出各种数据要求。

（2）处理要求：指用户有什么处理要求（如响应时间、处理方式等），最终要实现什么处理功能。

6.2.1需求分析的任务

具体而言，需求分析阶段的任务包括以下方面：

1、调查、收集、分析用户需求，确定系统边界

具体的做法是：

(1) 调查组织机构情况。包括了解该组织的部门组成情况、各部门的职责等，为分析信息流程做准备。

(2) 调查各部门的业务活动情况，包括了解各部门输入和使用什么数据，如何加工处理这些数据？输出什么信息？输出到什么部门？输出结果的格式是什么？这是调查的重点

6.2.1需求分析的任务

（3）在熟悉业务的基础上，明确用户对新系统的各种要求，如信息要求，处理要求，完全性和完整性要求

（4）确定系统边界。即确定那些活动由计算机和将来由计算机来完成，哪些只能由人工来完成。由计算机完成的功能是新系统应该实现的功能。

2、编写系统需求分析说明书

系统需求分析说明书也称系统需求规范说明书，是系统分析阶段的最后工作，是对需求分析阶段的一个总结，编写系统需求分析说明书是一个不断反复、逐步完善的过程。系统需求分析说明书一般应包括如下内容：

6.2.1需求分析的任务

（1）系统概况，包括系统的目标、范围、背景、历史和现状等；

（2）系统的原理和技术；

（3）系统总体结构和子系统结构说明；

（4）系统总体功能和子系统功能说明；

（5）系统数据处理概述、工程项目体制和设计阶段划分；

（6）系统方案及技术、经济、实施方案可行性等。

6.2.1需求分析的任务

随系统需求分析说明书可提供以下附件：

（1）系统的软硬件支持环境的选择及规格要求（所选择的数据库管理系统、操作系统、计算机型号及其网络环境等）。

（2）组织机构图、组织之间联系图和各机构功能业务一览图。

（3）数据流程图、功能模块图和数据字典等图表。

6.2.2需求分析的方法

用于需求分析的方法有很多种，主要的方法有自顶向下和自底向上两种，其中结构化分析方法（Structured Analysis，简称SA）是一种简单实用的方法。SA方法是从最上层的系统组织入手，采用自顶向下、逐层分解的方法分析系统。

SA方法把每个系统都抽象成图6.3的形式。图6.3只是给出了最高层次抽象的系统概貌，要反映更详细的内容，可将处理功能分解为若干个子系统，每个子系统还可以继续分解，直到把系统工作过程表示清楚为止。在处理功能逐步分解的同时，它们所用的数据也逐级分解，形成有若干层次的数据流图。

6. 2. 2需求分析的方法

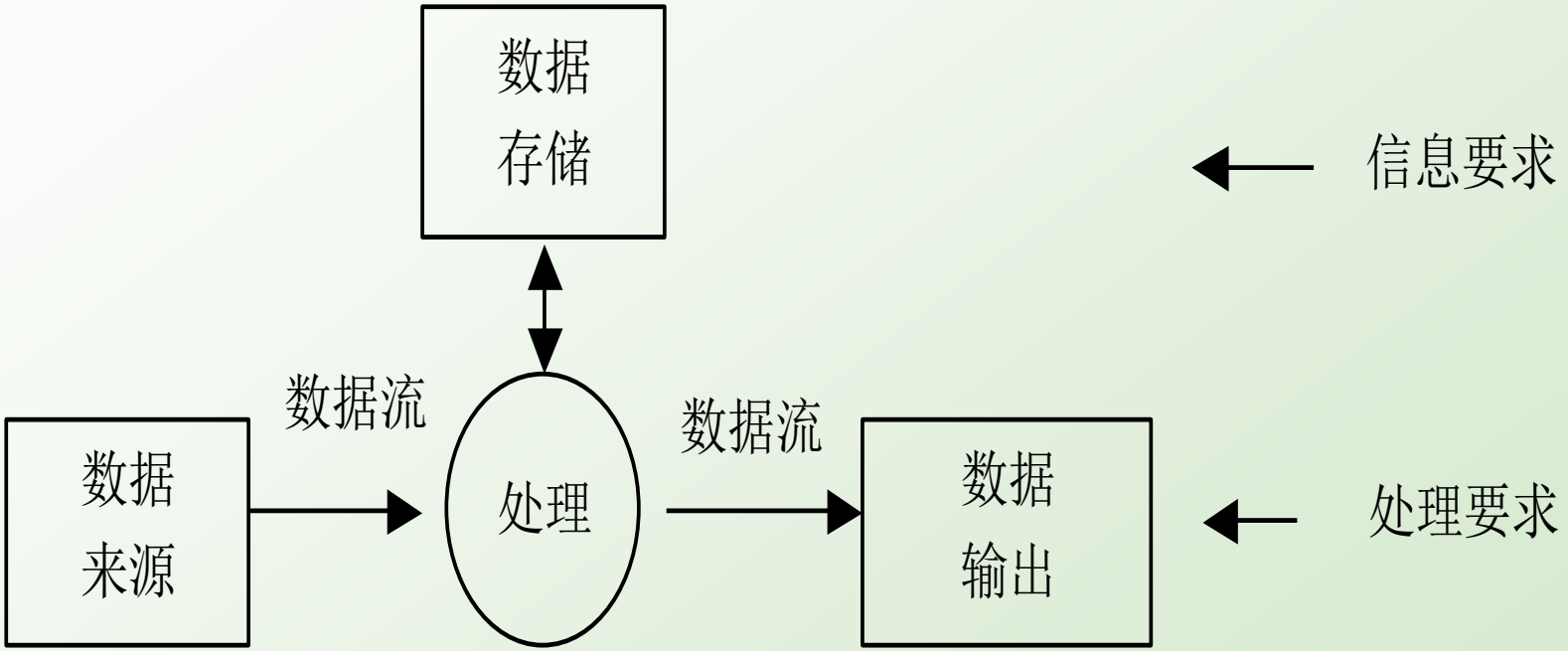


图6.3 系统最高层数据抽象图

6.2.2需求分析的方法

数据流图表达了数据和处理过程的关系。在SA方法中，处理过程的处理逻辑常常借助判定表和判定树来描述。系统中的数据则借助数据字典（DD）来描述。

下面介绍一下数据字典和数据流图

1、数据字典

数据字典是系统中各类数据描述的集合，是各类数据结构和属性的清单。它与数据流图互为解释，数据字典贯穿于数据库需求分析直到数据库运行的全过程，在不同的阶段其内容形式和用途各有区别，在需求分析阶段，它通常包含以下五个部分内容。

6.2.2需求分析的方法

（1）数据项

数据项描述＝｛数据项名，数据项含义说明，别名，数据类型，长度，取值范围，取值含义，与其他数据项的逻辑关系，数据项之间的联系｝，其中，取值范围、与其他数据项的逻辑关系定义了数据的完整性约束条件。

（2）数据结构

数据结构描述＝｛数据结构名，含义说明，组成:｛数据项或数据结构｝｝

（3）数据流

数据流描述＝｛数据流名，说明，数据流来源，数据流去向，组成:｛数据结构｝，平均流量，高峰期流量｝

6.2.2需求分析的方法

（4）数据存储

数据存储描述＝｛数据存储名，说明，编号，流入的数据流 ，流出的数据流 ， 组成:｛数据结构｝，数据量，存取方式｝

（5）处理过程

处理过程描述＝｛处理过程名，说明，输入:｛数据流｝，输出:｛数据流｝，处理:｛简要说明｝｝

6.2.2需求分析的方法

其中简要说明主要说明该处理过程的功能及处理要求：

功能：该处理过程用来做什么。

处理要求：处理频度要求（如单位时间里处理多少事务，多少数据量）；响应时间要求等。

处理要求是后面物理设计的输入及性能评价的标准。

最终形成的数据流图和数据字典为“系统需求分析说明书”的主要内容，这是下一步进行概念设计的基础。

6.2.2需求分析的方法

2、数据流图

数据流图（Data Flow Diagram，简称DFD）表达了数据与处理的关系。

数据流图中的基本元素有：

(1) 圆圈表示处理，输入数据在此进行变换产生输出数据。其中注明处理的名称。

(2) 矩形描述一个输入源点或输出汇点。其中注明源点或汇点的名称。

(3) 命名的箭头描述一个数据流。被加工的数据及其流向，流线上注明数据名称，箭头代表数据流动方向。

6.3 概念结构设计

6.3.1 概念结构设计的必要性

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，它是整个数据库设计的关键。

在进行功能数据库设计时，如果将现实世界中的客观对象直接转换为机器世界中的对象，就会感到比较复杂，注意力往往被牵扯到更多的细节限制方面，而不能集中在最重要的信息的组织结构和处理模式上，因此，通常是将现实世界中的客观对象首先抽象为不依赖任何DBMS支持的数据模型。故概念模型可以看成是现实世界到机器世界的一个过度的中间层次。

6.3.1 概念结构设计的必要性

概念模型是各种数据模型的共同基础，它比数据模型更独立于机器、更抽象。将概念结构设计从设计过程中独立出来，可以带来以下好处：

(1) 任务相对单一化，设计复杂程度大大降低，便于管理。

(2) 概念模式不受具体的DBMS的限制，也独立于存储安排和效率方面的考虑，因此，更稳定。

(3) 概念模型不含具体DBMS所附加的技术细节，更容易被用户理解，因而更能准确的反映用户的信息需求。

6.3.2 概念模型设计的特点

概念结构设计的特点有以下几点：

① 易于理解，从而可以用它和不熟悉计算机的用户交换意见，用户的积极参与是数据库的设计成功的关键。

② 能真实、充分地反映现实世界，包括事物和事物之间的联系，能满足用户对数据的处理要求。是对现实世界的一个真实模型。

③ 易于更改，当应用环境和应用要求改变时，容易对概念模型修改和扩充。

④ 易于向关系、网状、层次等各种数据模型转换。

6.3.3 概念结构的设计方法和步骤

1、概念结构的设计方法

(1) 自顶向下。 如图6.4所示

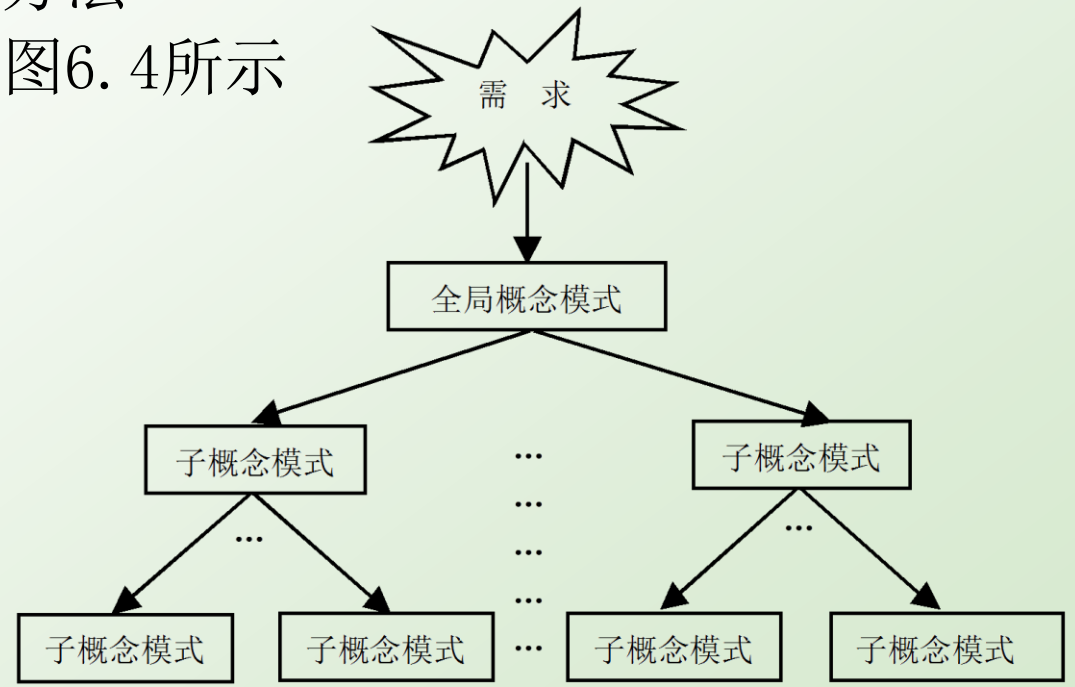


图6.4 自顶向下的设计方法

6.3.3 概念结构的设计方法和步骤

（2）自底向上。如图6.5所示

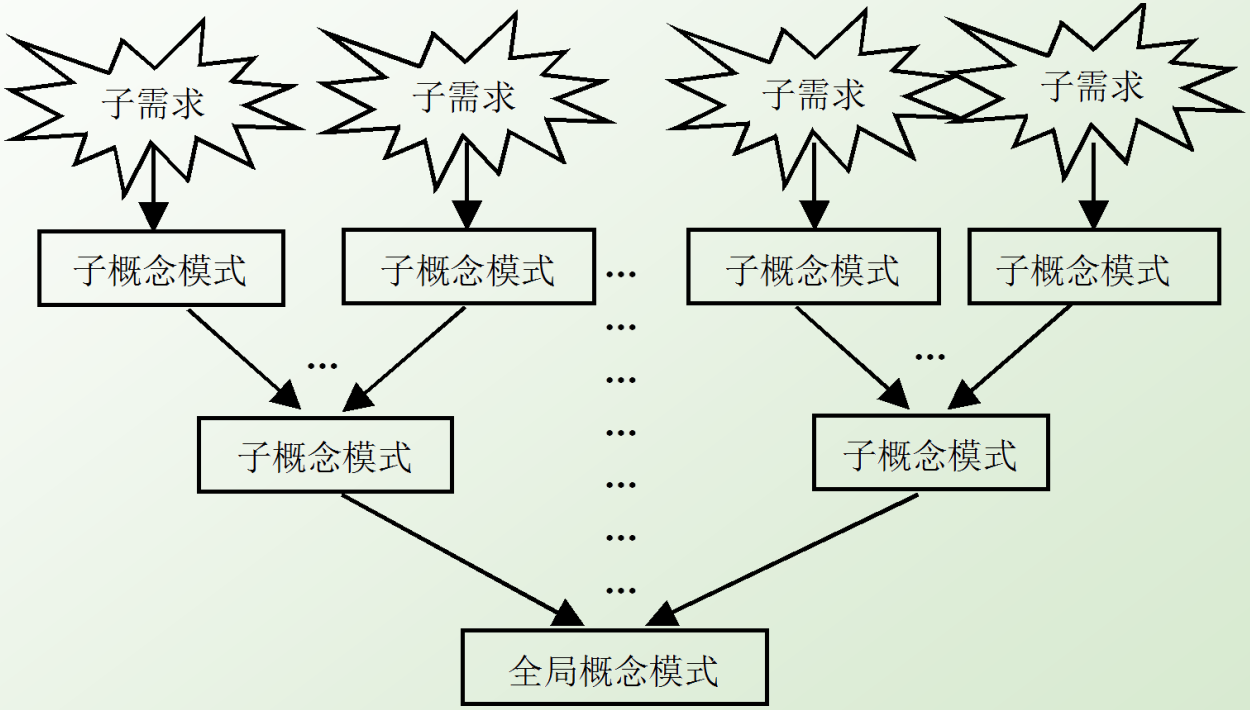


图6.5 自底向上的设计方法

6.3.3 概念结构的设计方法和步骤

（3）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。如图6.6所示。

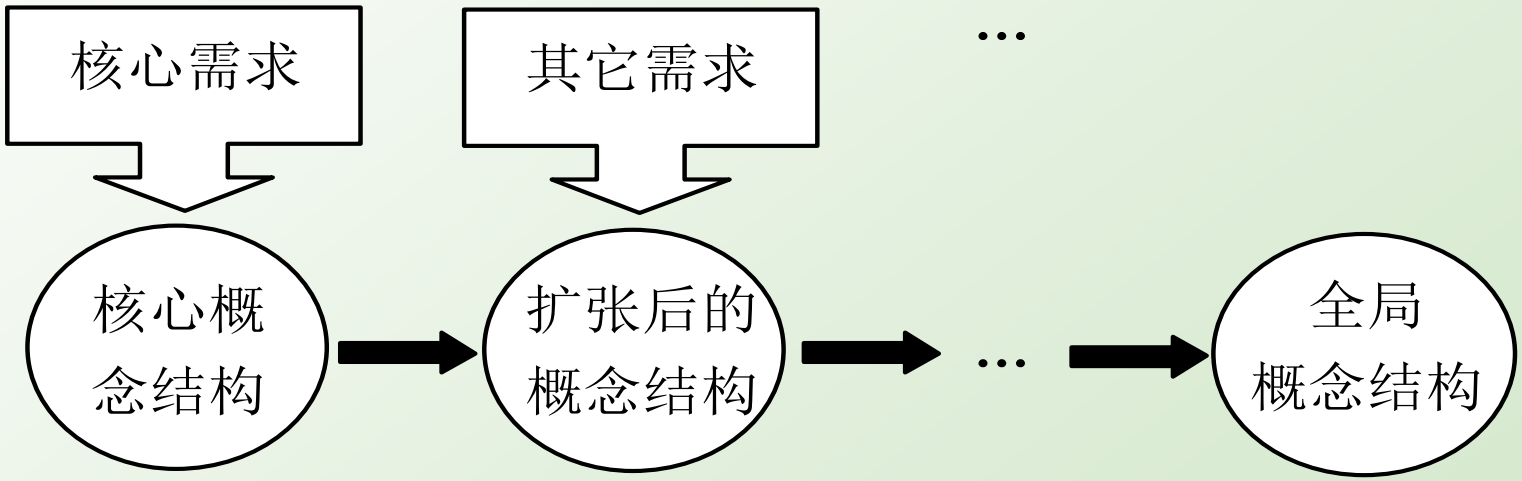


图6.6 逐步扩张的设计方法

6.3.3 概念结构的设计方法和步骤

（4）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。

其中最常用的方法是自底向上。即自顶向下地进行需求分析，再自底向上地设计概念模式结构。

2、概念结构设计的步骤

对于自底向上的设计方法来说，概念结构的步骤分为两步（如图6.7所示）

① 进行数据抽象，设计局部E-R模型

② 集成各局部E-R模型，形成全局E-R模型

6.3.3 概念结构的设计方法和步骤

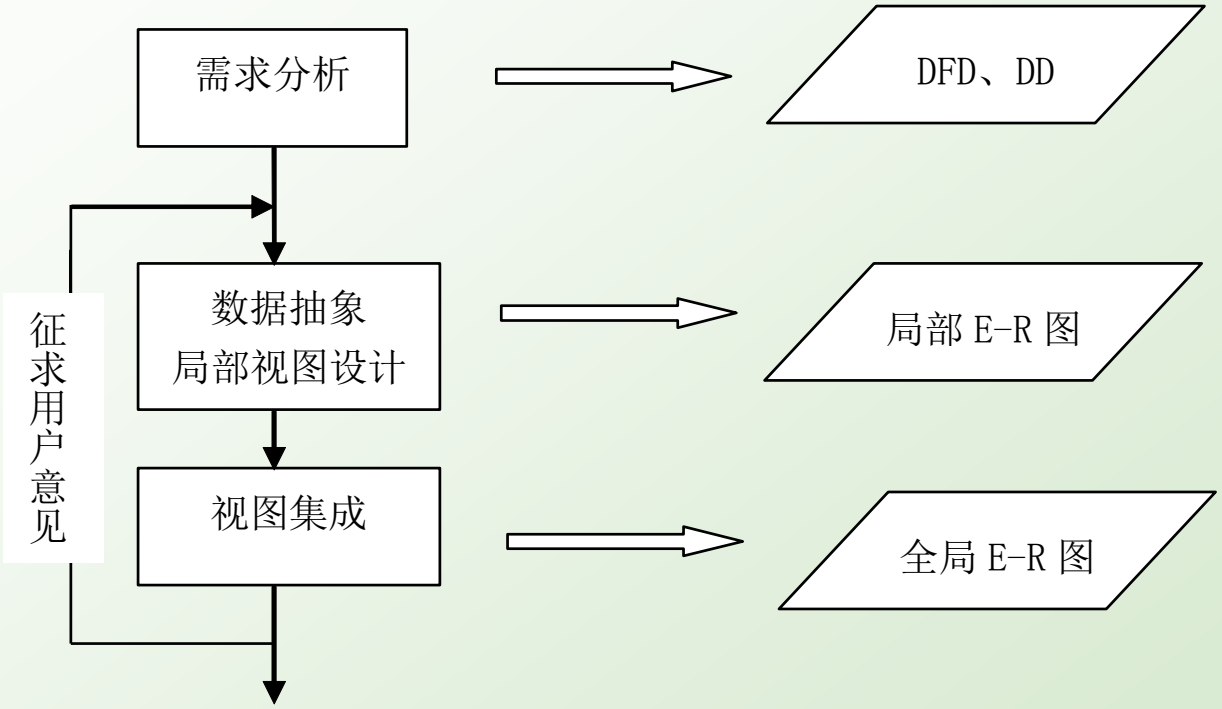


图6.7自底向上方法的设计步骤

6.3.3 概念结构的设计方法和步骤

3、数据抽象与局部E-R模型设计

(1) 数据抽象

设计局部E-R模型的关键就在于正确划分实体和属性。实体和属性在形式上并无可以明显区分的界限，通常是按照现实世界中事物的自然划分来定义实体和属性，将现实世界中的事物进行数据抽象，得到实体和属性。一般有两种数据抽象：分类和聚集。

① 分类

定义某一类概念作为现实世界中一组对象的类型，将一组具有某些共同特性和行为的对象抽象为一个实体，对象和实体之间是“is member of ”的关系。

6.3.3 概念结构的设计方法和步骤

② 聚集

定义某个类型的组成成分。将对象的类型的组成成分抽象为实体的属性。抽象了对象内部类型和成分的“is part of”的语义。

(2) 局部视图设计

选择好一个局部应用之后，就要对每个局部应用逐一设计分E-R图，也称局部E-R图. 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、标识实体的键，确定实体之间的联系及其类型（1:1，1:n，m:n）。

6.3.3 概念结构的设计方法和步骤

实际上实体和属性是相对而言的，往往要根据实际情况进行必要的调整，在调整时要遵守两条原则：

① 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。

② 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。

例如：“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。

6.3.3 概念结构的设计方法和步骤

再如：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。如图6.8

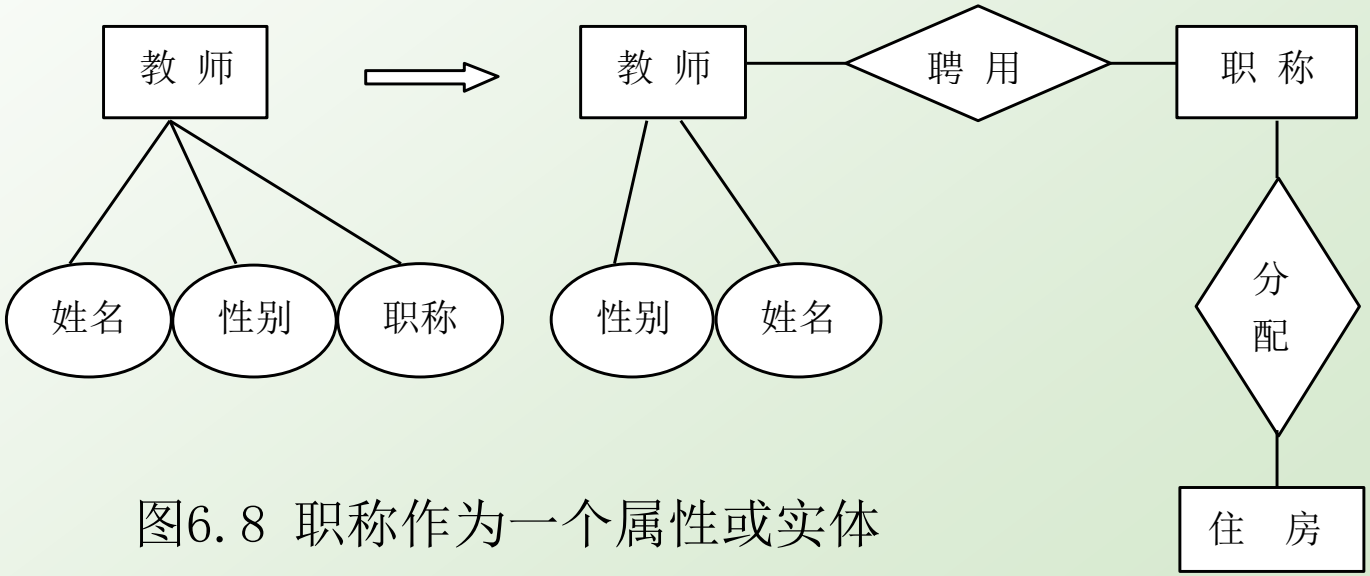


图6.8 职称作为一个属性或实体

6.3.3 概念结构的设计方法和步骤

［例1］设有如下实体：

学生：学号、单位名称、姓名、性别、年龄、选修课程名

课程：编号、课程名、开课单位、任课教师号

教师：教师号、姓名、性别、职称、讲授课程编号

单位：单位名称、电话、教师号、教师姓名

上述实体中存在如下联系：

（1）一个学生可选修多门课程，一门课程可为多个学生选修

（2）一个教师可讲授多门课程，一门课程可为多个教师讲授

（3）一个系可有多个教师，一个教师只能属于一个系

6.3.3 概念结构的设计方法和步骤

根据上述约定，可以得到学生选课局部E-R图和教师授课局部E-R图。分别如图6.12。

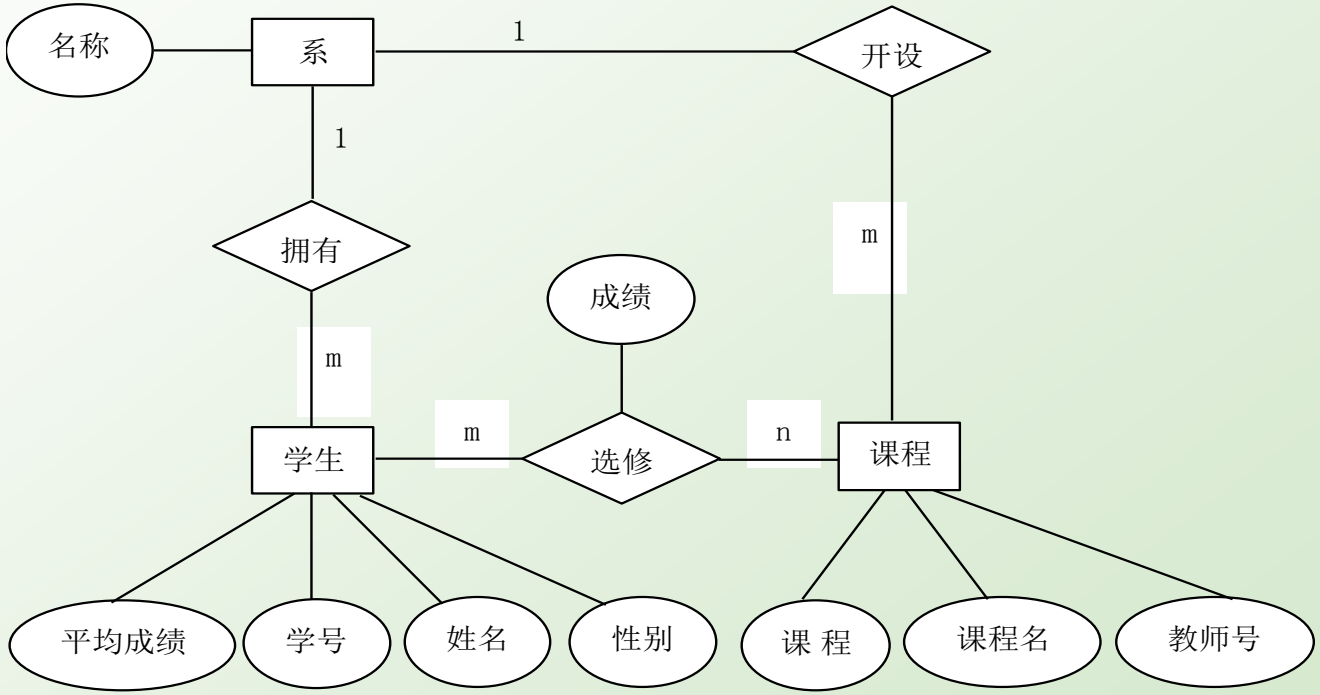


图6.12 学生选课局部E-R图

6.3.3 概念结构的设计方法和步骤

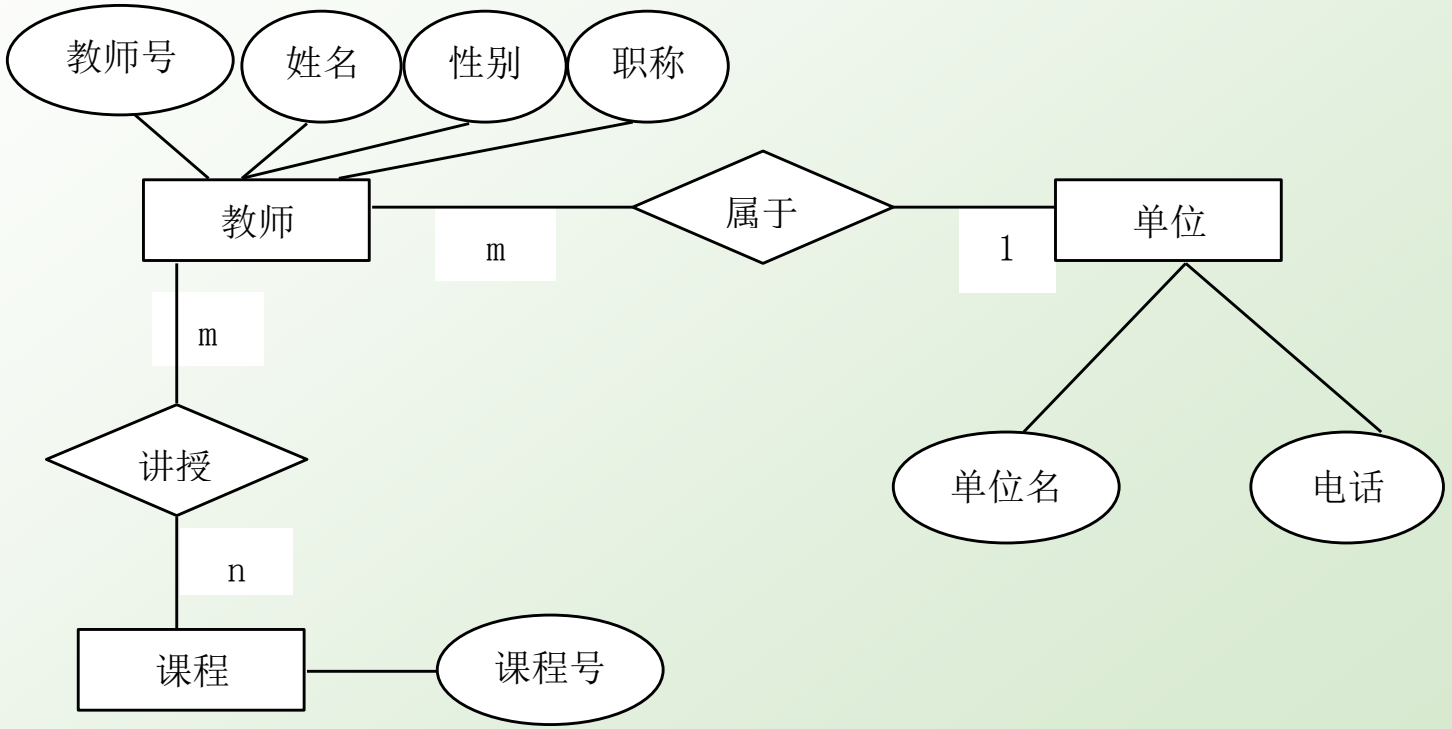


图6.13 教师任课局部E-R图

6.3.3 概念结构的设计方法和步骤

4、全局E-R模型设计

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，集成为一个整体的概念数据结构即全局E-R图。也就是视图的集成，视图的集成有两种方式：

(1)一次集成法：一次集成多个分E-R图，通常用于局部视图比较简单时。如图6.14所示。

(2)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，如图6.15所示。

6.3.3 概念结构的设计方法和步骤

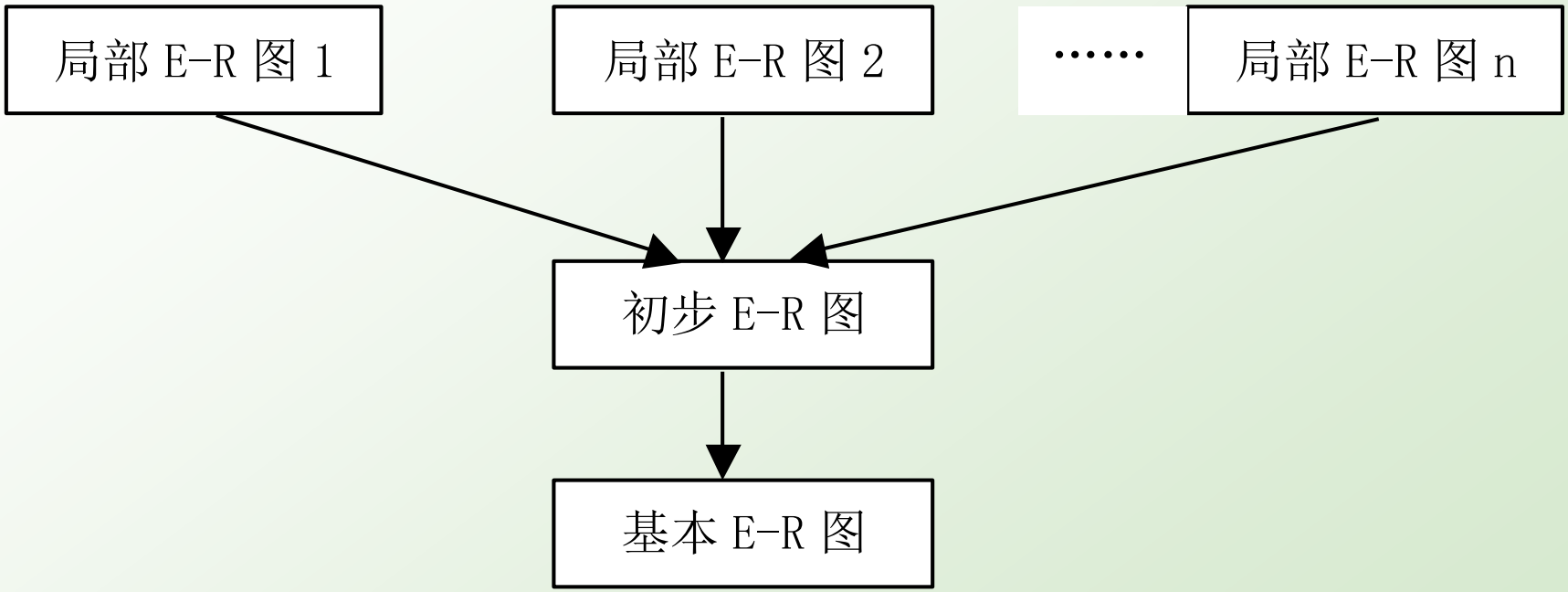


图6.14 一次集成法

6.3.3 概念结构的设计方法和步骤

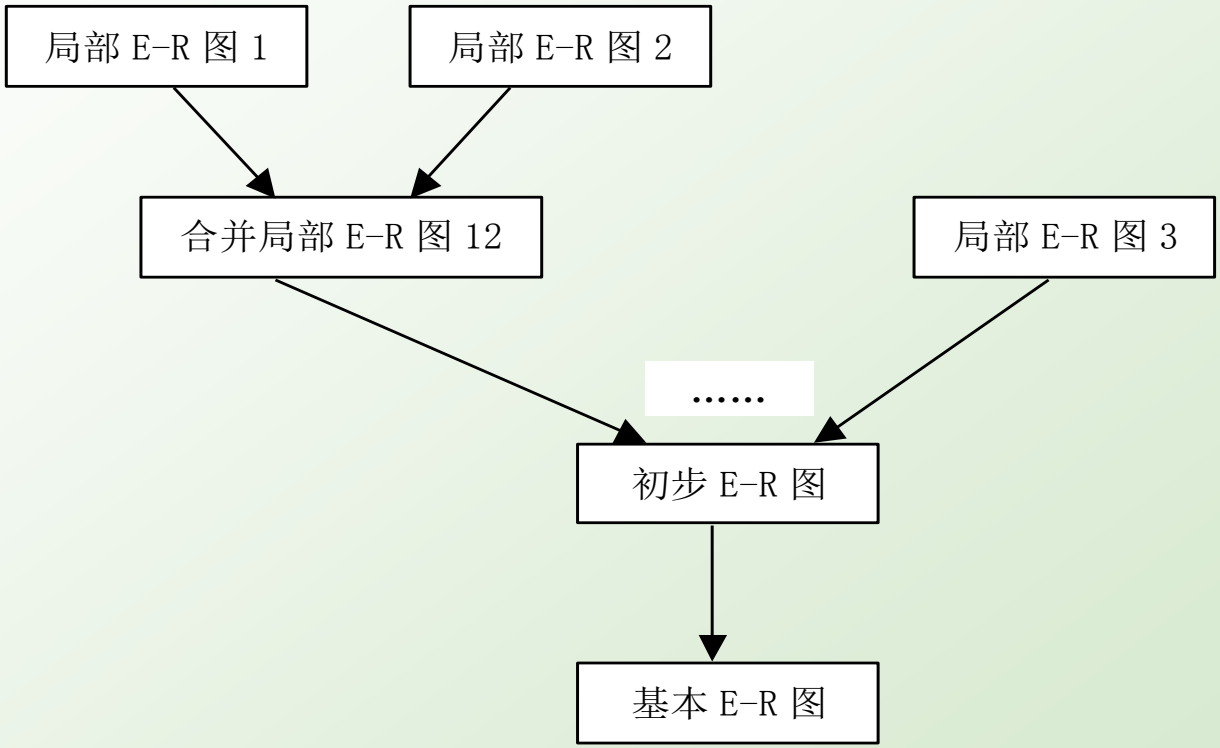


图6.15 逐步逐步累积式

6.3.3 概念结构的设计方法和步骤

由上图可知，不管用哪种方法，集成局部E-R图的都分为两个步骤，如图6.16所示。

(1)合并：解决各个局部E-R图之间的冲突，将各个局部E-R图合并起来生成初步E-R图。

(2)修改与重构：消除不必要的冗余，生成基本E-R图。

6.3.3 概念结构的设计方法和步骤

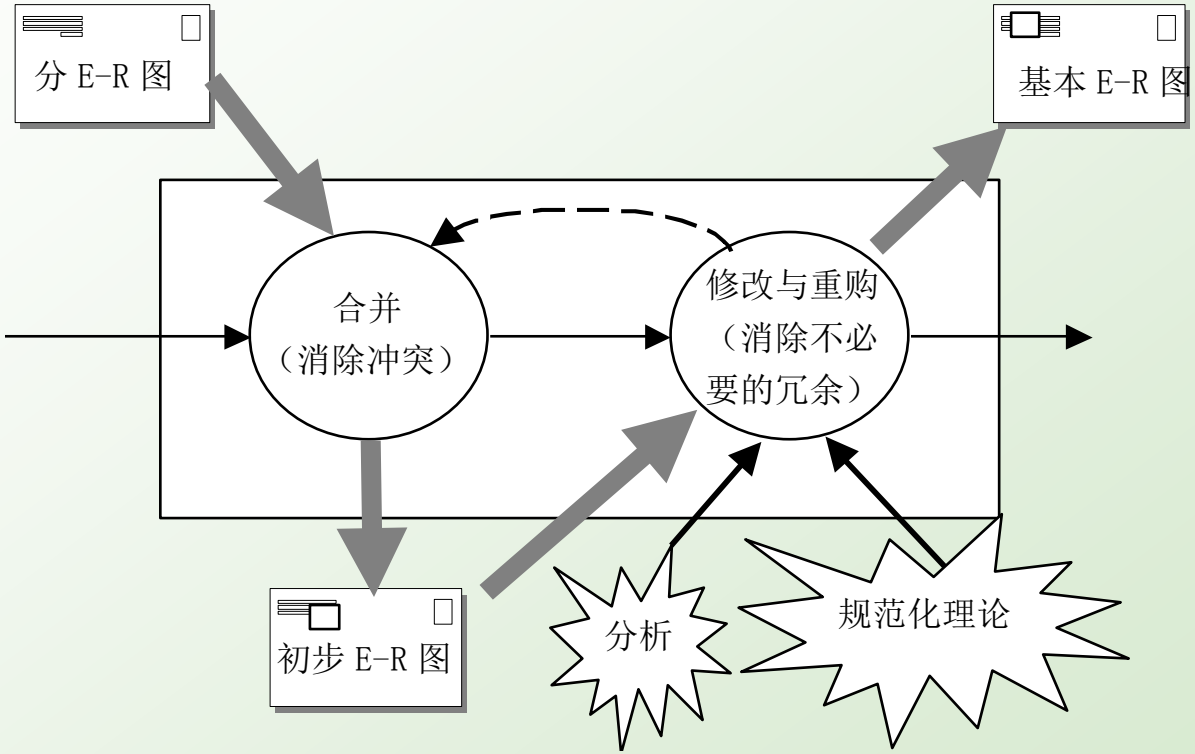


图6.16 视图的集成

6.3.3 概念结构的设计方法和步骤

（1）合并分E-R图，生成初步E-R图

合并分E-R图时并不能简单地将各个分E-R图画到一起，而是必须着力消除各个分E-R图中不一致的地方，以形成一个能为全系统中所有用户共同理解和接受的统一概念模型。合理消除各分E-R图的冲突，合并分E-R图的主要工作与关键所在。

E-R图中的冲突有三种：属性冲突，命名冲突结构冲突。

① 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。

属性取值单位冲突。

6.3.3 概念结构的设计方法和步骤

② 命名冲突

命名不一致可能发生在实体名、属性名或联系名之间，其中属性的命名冲突更为常见。一般表现为同名异义或异名同义。

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。

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。

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。解决命名冲突的方法是通常用讨论、协商等行政手段加以解决。

6.3.3 概念结构的设计方法和步骤

③ 结构冲突(有三类结构冲突)

同一对象在不同应用中具有不同的抽象

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。

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。

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。

实体之间的联系在不同局部视图中呈现不同的类型

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。

6.3.3 概念结构的设计方法和步骤

（2）消除不必要的冗余，设计基本E-R图

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。

采用分析的方法来消除数据冗余，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。

前面图6.12和图6.13在形成初步E-R图后，以及消除冗余联系，便可得到基本的E-R模型，如下图6.17所示。

6.3.3 概念结构的设计方法和步骤

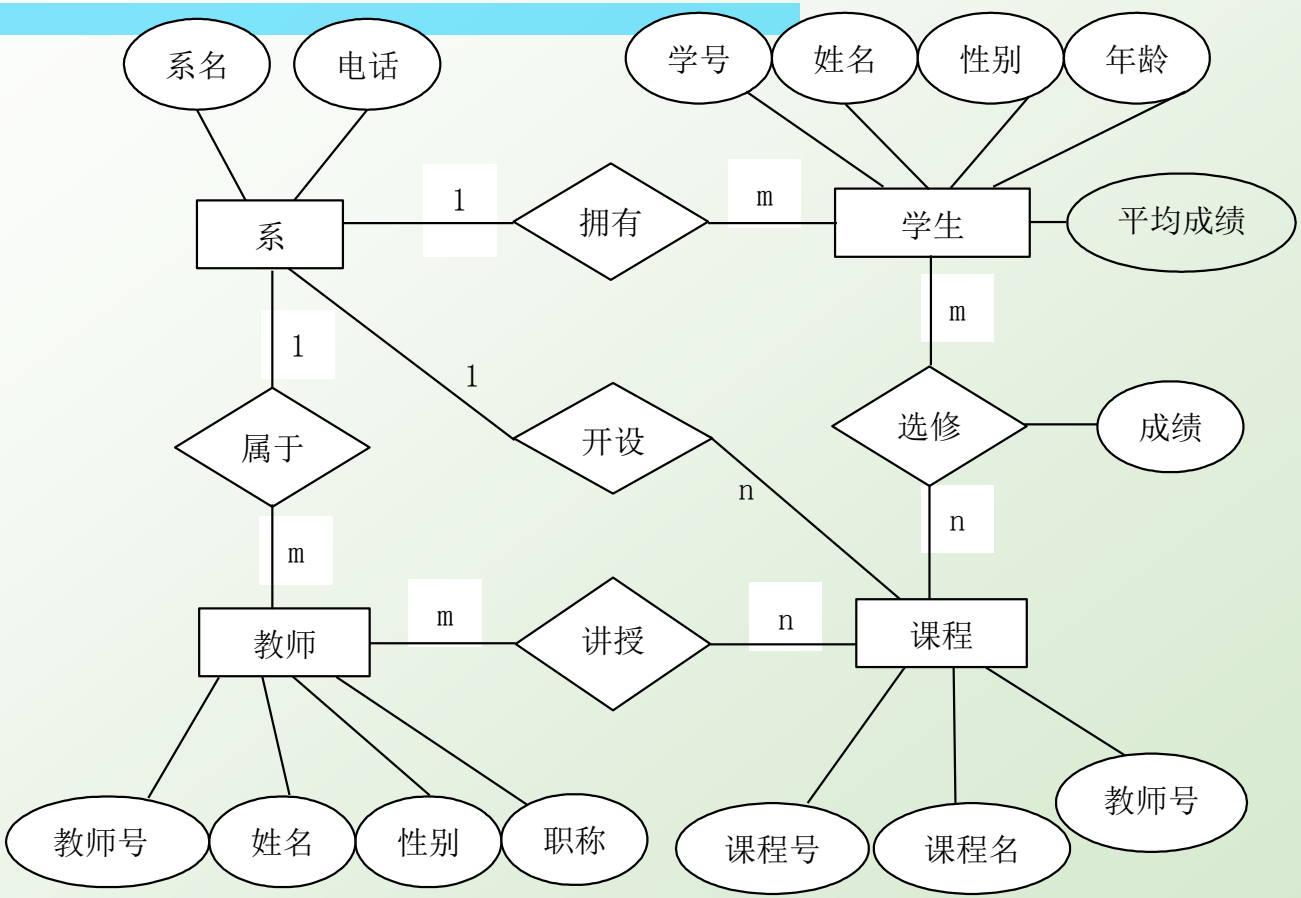


图6.17 初步的全局E-R图

6.3.3 概念结构的设计方法和步骤

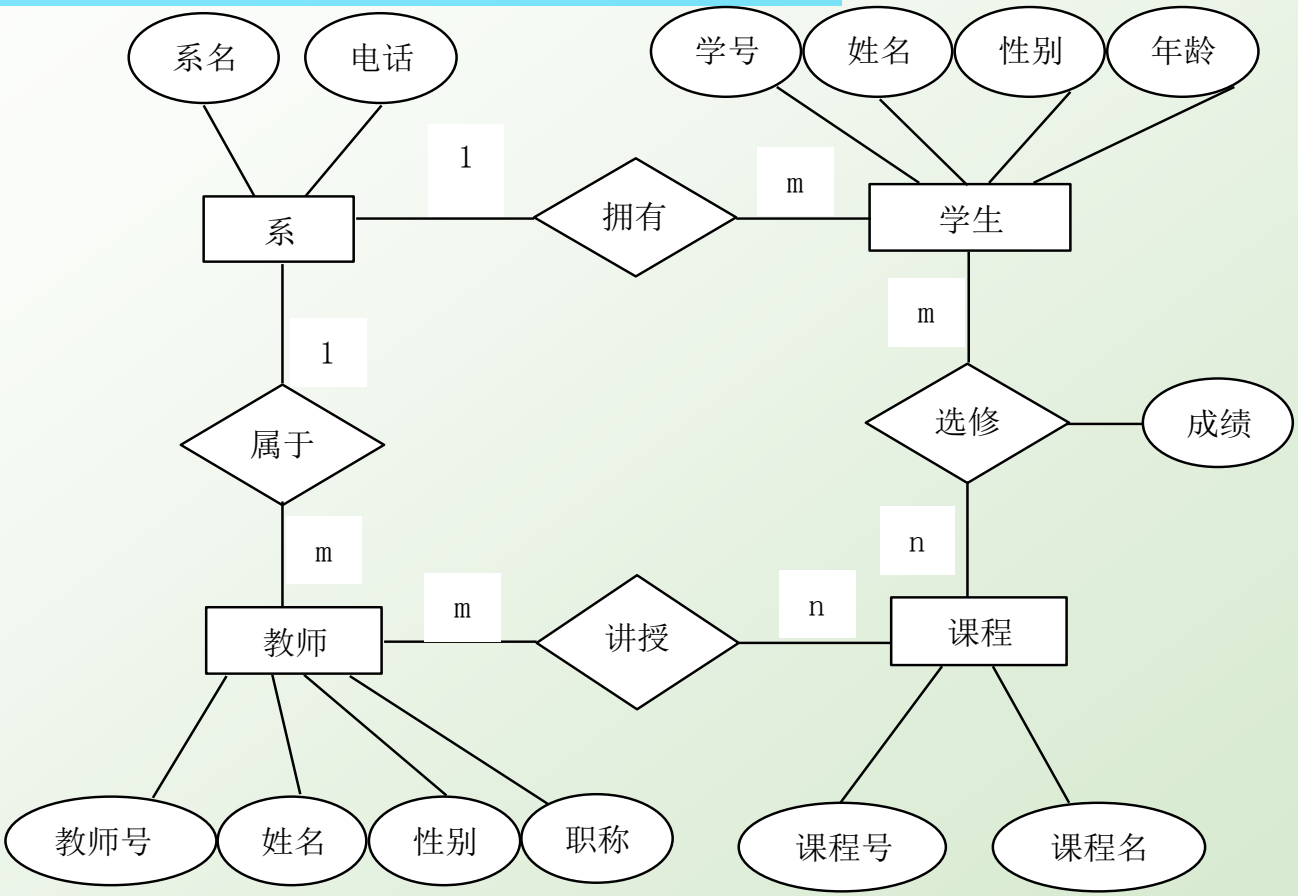


图6.18 优化后的基本E-R图

6.4 逻辑结构设计

6.4.1 逻辑结构设计的任务和步骤

概念结构是各种数据模型的共同基础。为了能够用某一DBMS实现用户需求，还必须将概念结构进一步转化为相应的数据模型，这正是数据库逻辑结构设计所要完成的任务。

一般的逻辑结构设计分为以下三个步骤（如图6.16）

- 将概念结构转化为一般的关系、网状、层次模型。
- 将转化来的关系、网状、层次模型向特定DBMS支持下的数据模型转换。
- 对数据模型进行优化。

6.4.1 逻辑结构设计的任务和步骤

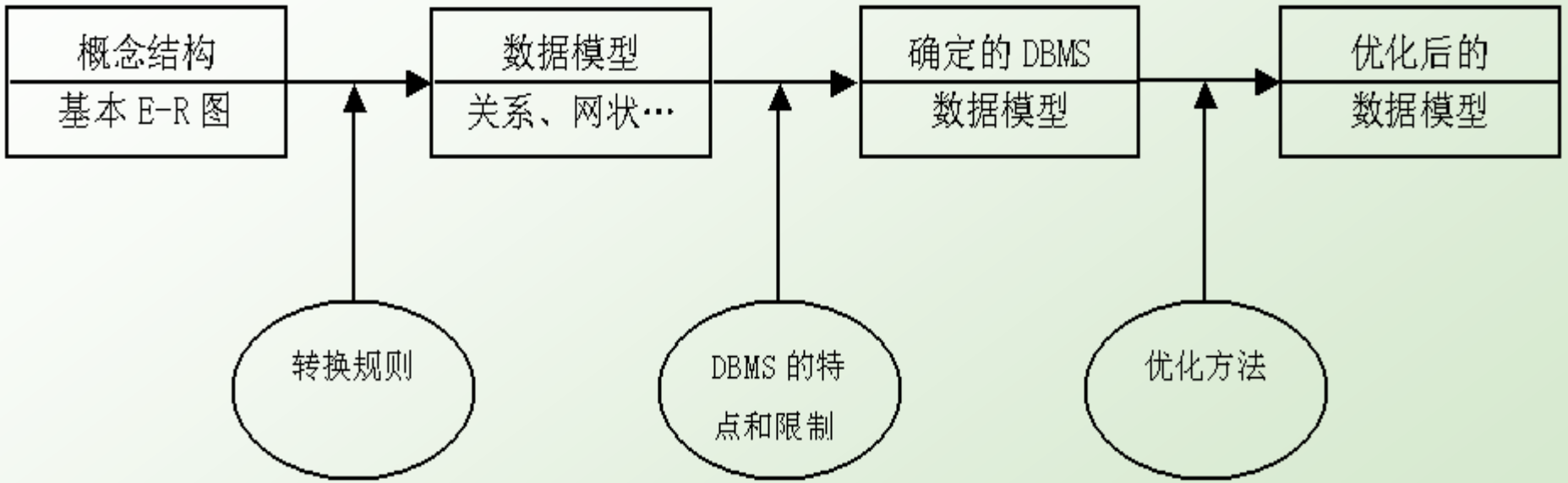


图 6.19 逻辑结构设计三步骤

6.4.2 初始化关系模式设计

1、转换原则

（1）一个实体转换为一个关系模式。

关系的属性：实体的属性

关系的键：实体的键

（2）一个m:n联系转换为一个关系模式。

关系的属性：与该联系相连的各实体的键以及联系本身的属性。

关系的键：各实体键的组合。

（3）一个1:n联系可以转换为一个关系模式

6.4.2 初始化关系模式设计

关系的属性：与该联系相连的各实体的码以及联系本身的属性

关系的码：n端实体的键

说明：一个1:n联系也可以与n端对应的关系模式合并，这时需要把1端关系模式的码和联系本身的属性都加入到n端对应的关系模式中。

（4）一个1:1联系可以转换为一个独立的关系模式。

关系的属性：与该联系相连的各实体的键以及联系本身的属性

关系的候选码：每个实体的码均是该关系的候选码

说明：一个1:1联系也可以与任意一端对应的关系模式合并，这时需要把任一端关系模式的码及联系本身的属性都加入到另一端对应的关系模式中。

（5）三个或三个以上实体间的一个多元联系转换为一个关系模式。

6.4.2 初始化关系模式设计

关系的属性：与该多元联系相连的各实体的键以及联系本身的属性。

关系的码：各实体键的组合。

2、具体做法

（1）把一个实体转换为一个关系。先分析该实体的属性，从中确定主键，然后再将其转换为关系模式

6.4.2 初始化关系模式设计

[例2] 以6.15图为例将四个实体分别转换为关系模式（带下划线的为主键）：

学生（学号，姓名，性别，年龄）
课程（课程号，课程名）
教师（教师号，姓名，性别，职称）
系（系名，电话）

6.4.2 初始化关系模式设计

(2) 把每个联系转换成关系模式：

[例3] 把6.18图中的四个联系也转换成关系模式：

属于（教师号，系名）

讲授（教师号，课程号）

选修（学号，课程号，成绩）

拥有（系名，学号）

(3) 三个或三个以上的实体间的一个多元联系在转换为一个关系模式时，与该多元联系相连的各实体的主键及联系本身的属性均转换成为关系的属性，转换后所有得到的关系的主键为个实体键的组合。

6.4.2 初始化关系模式设计

[例4] 图6.20表示供应商、项目和零件三个实体之间的多对多联系，如果已知三个实体的主键分别为“供应商号”，“项目号”与“零件号”，则它们之间的联系“供应”转换为关系模式：供应（供应号，项目号，零件号，数量）。

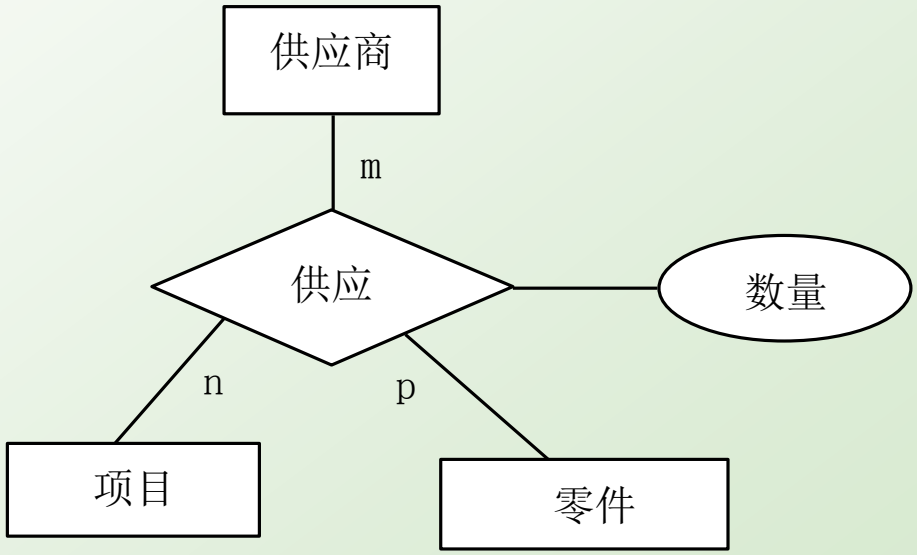


图6.20 多个实体之间的联系

6.4.3 关系模式的规范化

关系模型的优化通常是以规范化理论为基础。方法为：

1、确定数据依赖，按需求分析阶段所得到的语义，分别写出每个关系模式内部各属性之间的数据依赖以及不同关系模式属性之间数据依赖。

2、对于各个关系模式之间的数据依赖进行极小化处理，消除冗余的联系。

3、按照数据依赖的理论对关系模式逐一进行分析，考查是否存在部分函数依赖、传递函数依赖、多值依赖等，确定各关系模式分别属于第几范式。

6.4.3 关系模式的规范化

4、按照需求分析阶段得到的各种应用对数据处理的要求，分析对于这样的应用环境这些模式是否合适，确定是否要对它们进行合并或分解。

5、按照需求分析阶段得到的各种应用对数据处理的要求，对关系模式进行必要的分解或合并，以提高数据操作的效率和存储空间的利用率。

6.4.4 关系模式的评价与改进

1、模式的评价

对模式的评价包括设计质量的评价和性能评价两个方面

2、数据模式的改进

（1）分解

关系模式的分解一般分为水平分解和垂直分解两种。

（2）合并

具有相同主键的关系模式，且对这些关系模式的处理主要是查询操作，而且经常是多关系的查询，那么可对这些关系模式按照组合频率进行合并。

6.5 数据库物理设计

数据库物理设计的任务是为上一阶段得到的数据库逻辑模式，即数据库的逻辑结构选择合适的应用环境的物理结构，既确定有效地实现逻辑结构模式的数据库存储模式，确定在物理设备上所采用的存储结构和存取方法，然后对该存储模式进行性能评价、修改设计，经过多次反复，最后得到一个性能较好的存储模式。

6.5.1 确定物理结构

数据库物理设计内容包括记录存储结构的设计，存储路径的设计，记录集簇的设计。

1、记录存储结构的设计

记录存储结构的设计就是设计存储记录的结构形式，它涉及不定长数据项的表示。

2、关系模式的存取方法选择

DBMS常用存取方法有：索引方法，目前主要是B+树索引方法，聚簇（Cluster）方法，HASH方法。

6.5.1 确定物理结构

(1) 索引方法

索引存取方法的主要内容：对哪些属性列建立索引，对哪些属性列建立组合索引，对哪些索引要设计为唯一索

(2) 聚簇

为了提高某个属性（或属性组）的查询速度，把这个或这些属性（称为聚簇码）上具有相同值的元组集中存放在连续的物理块称为聚簇。聚簇的用途：大大提高按聚簇属性进行查询的效率，

6.5.1 确定物理结构

（3）Hash方法

当一个关系满足下列两个条件时，可以选择HASH存取方法：

① 该关系的属性主要出现在等值连接条件中或主要出现在相等比较选择条件中。

② 该关系的大小可预知且关系的大小不变或该关系的大小动态改变但所选用的DBMS提供了动态HASH存取方法。

6.5.2 评价物理结构

和前面几个设计阶段一样，在确定了数据库的物理结构之后，要进行评价，重点是时间和空间的效率。如果评价结果满足设计要求，则可进行数据库实施，实际上，往往需要经过反复测试才能优化物理设计。

6.6 数据库实施

数据库实施是指根据逻辑设计和物理设计的结果，在计算机上建立起实际的数据库结构，装入数据，进行测试和试运行的过程。数据库实施的工作内容包括：用DDL定义数据库结构，组织数据入库，编制与调试应用程序，数据库试运行。

6.6.1 建立实际数据库结构

确定了数据库的逻辑结构与物理结构后，就可以用所选用的DBMS提供的数据定义语言（DDL）来严格描述数据库结构。

6.6.2 装入数据

数据装载方法有人工方法与计算机辅助数据入库方法两种。

1、人工方法

适用于小型系统其步骤如下：

① 筛选数据。需要装入数据库中的数据通常都分散在各个部门的数据文件或原始凭证中，所以首先必须把需要入库的数据筛选出来。

② 转换数据格式。筛选出来的需要入库的数据，其格式往往不符合数据库要求，还需要进行转换。这种转换有时可能很复杂。

6.6.2 装入数据

③ 输入数据。将转换好的数据输入计算机中。

④ 校验数据。检查输入的数据是否有误。

2、计算机辅助数据入库：

适用于中大型系统其步骤如下：

① 筛选数据。

② 输入数据。由录入员将原始数据直接输入计算机中。数据输入子系统应提供输入界面。

6.6.2 装入数据

③ 校验数据。数据输入子系统采用多种检验技术检查输入数据的正确性。

④ 转换数据。数据输入子系统根据数据库系统的要求，从录入的数据中抽取有用成分，对其进行分类，然后转换数据格式。抽取、分类和转换数据是数据输入子系统的主要工作，也是数据输入子系统的复杂性所在。

⑤ 综合数据。数据输入子系统对转换好的数据根据系统的要求进一步综合成最终数据。

6.6.3 编制与调试应用程序

数据库应用程序的设计应该与数据库设计并行进行。在数据库实施阶段，当数据库结构建立好后，就可以开始编制与调试数据库的应用程序。调试应用程序时由于数据入库尚未完成，可先使用模拟数据。

6.6.4 数据库试运行

数据库试运行也称为联合调试，其主要工作包括：

1、功能测试：实际运行应用程序，执行对数据库的各种操作，测试应用程序的各种功能。

2、性能测试：测量系统的性能指标，分析是否符合设计目标。

6.6.4 数据库试运行

重新设计物理结构甚至逻辑结构，会导致数据重新入库。由于数据入库工作量实在太大，所以可以采用分期输入数据的方法：

- 先输入小批量数据供先期联合调试使用。
- 待试运行基本合格后再输入大批量数据。
- 逐步增加数据量，逐步完成运行评价。

6.6.5 整理文档

在程序的编制和试运行中，应将发现的问题和解决方法记录下来，将它们整理存档为资料，供以后正式运行和改进时参考，全部的调试工作完成之后，应该编写应用系统的技术说明书，在系统正式运行时给用户，完整的资料是应用系统的重要组成部分。

6.7 数据库运行和维护

数据库试运行结果符合设计目标后，数据库就可以真正投入运行了。数据库投入运行标着开发任务的基本完成和维护工作的开始，对数据库设计进行评价、调整、修改等维护工作是一个长期的任务，也是设计工作的继续和提高。

对数据库经常性的维护工作主要是由DBA完成的，包括：三个方面的内容即数据库的转储和恢复，数据库的安全性、完整性控制，数据库性能的监督、分析和改进。

6.7.1 数据库的安全性、完整性

DBA必须根据用户的实际需要授予不同的操作权限，在数据库运行过程中，由于应用环境的变化，对安全性的要求也会发生变化，DBA需要根据实际情况修改原有的安全性控制。由于应用环境的变化，数据库的完整性约束条件也会变化，也需要DBA不断修正，以满足用户要求。

6.7.2 监视并改善数据库性能

在数据库运行过程中， DBA必须监督系统运行，对监测数据进行分析，找出改进系统性能的方法。

- 利用监测工具获取系统运行过程中一系列性能参数的值。
- 通过仔细分析这些数据，判断当前系统是否处于最佳运行状态。
- 如果不是，则需要通过调整某些参数来进一步改进数据库性能。

6.7.3 数据库的重组织和重构造

数据库的重组织，并不改变原设计的逻辑和物理结构，而数据库的重构造则不同，它是指部分修改数据库的模式和内模式。

由于数据库应用环境发生变化，增加了新的应用或新的实体，取消了某些旧的应用，有的实体与实体间的联系也发生了变化等，使原有的数据库设计不能满足新的需要，必需要调整数据库的模式和内模式。当然数据库的重构也是有限的，只能做部分修改。

6.8 UML简介*

随着面向对象程序设计语言及其技术、面向对象数据库管理系统等的不断发展与广泛深入应用，面向对象的分析与设计建模语言UML的学习与应用已成为一种必然趋势，本节将简要介绍UML语言。

6.8 UML简介*

- 6.8.1 概述

- 1、UML的概念和特点

- UML（Unified Modeling Language，统一建模语言）是一种为面向对象开发系统的产品进行说明、可视化和编制文档的一种标准语言，是非专利的第三代建模和规约语言。UML使用面向对象设计的观念，但独立于任何具体的程序设计语言。

- UML作为一种统一的软件建模语言具有广泛地建模能力。UML是在消化、吸收、提炼至今存在的所有软件建模语言的基础上提出的，集百家之所长，是软件建模语言的集大成者。UML还突破了软件的限制，广泛吸收了其他领域的建模方法，并根据建模的一般原理，结合了软件的特点，因此具有坚实的理论基础和广泛性。UML不仅可以用于软件建模，还可以用于其他领域的建模工作。

6.8 UML简介*

- UML立足于对事物的实体、性质、关系、结构、状态和动态变化过程的全程描述和反映。UML可以从不同角度描述人们所观察到的软件视图，也可以描述在不同开发阶段中的软件的形态。UML可以建立需求模型、逻辑模型、设计模型和实现模型等，但UML在建立领域模型方面存在不足，需要进行补充。
- 作为一种建模语言，UML有严格地语法和语义规范。UML建立在元模型理论基础上，包括4层元模型结构，分别是基元模型、元模型、模型和用户对象。4层结构层层抽象，下一层是上一层的实例。UML中的所有概念和要素均有严格的语义规范。

6.8 UML简介*

- 2、UML的出现与流行
- 公认的面向对象建模语言出现于20世纪70年代中期。从1989-1994年，其数量从不到10种增加到50多种。面向对象的分析与设计（OOA&D）方法的发展在20世纪80年代末至90年代中出现了一个高潮，UML是这个高潮的产物。它不仅统一了BOOCH、Rumnaugh和Jacobson的表示方法，而且对其作了进一步的发展，并最终统一为大众所接受的标准建模语言。1996年底，UML已稳占面向对象技术市场的85%，成为可视化建模语言事实上的工业标准。1997年11月17日，OMG(对象管理组)采纳UML1.1作为基于面向对象技术的标准建模语言。
- UML是一种可视化的建模语言，能够让系统构造者用标准的、易于理解的方式建立起能够表达他们设计思想的系统蓝图，并且提供一种机制，以便于不同的人之间有效地共享和交流设计成果。
- UML代表了面向对象方法的软件开发技术的发展方向，具有巨大的市场前景。

6.8 UML简介*

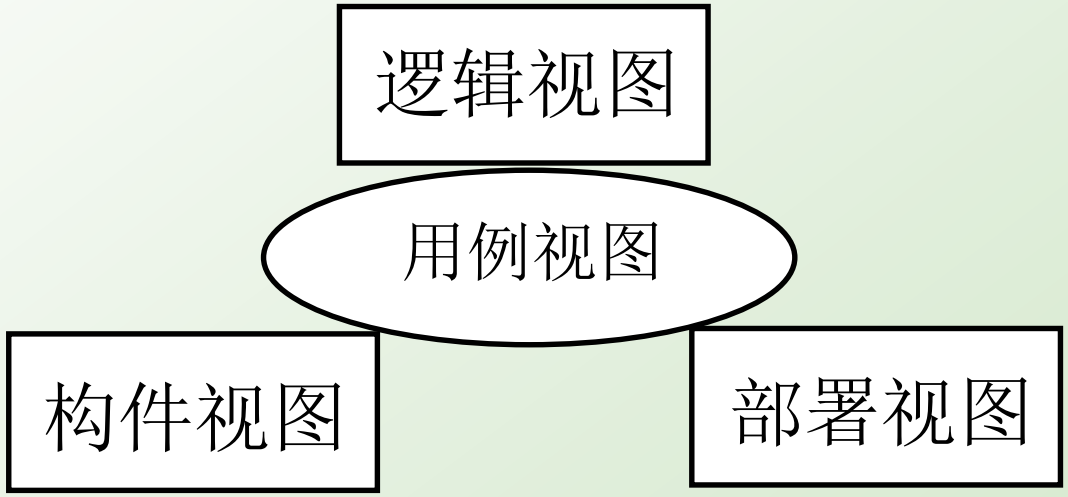
- 2、UML的出现与流行
- 公认的面向对象建模语言出现于20世纪70年代中期。从1989-1994年，其数量从不到10种增加到50多种。面向对象的分析与设计（OOA&D）方法的发展在20世纪80年代末至90年代中出现了一个高潮，UML是这个高潮的产物。它不仅统一了BOOCH、Rumnaugh和Jacobson的表示方法，而且对其作了进一步的发展，并最终统一为大众所接受的标准建模语言。1996年底，UML已稳占面向对象技术市场的85%，成为可视化建模语言事实上的工业标准。1997年11月17日，OMG(对象管理组)采纳UML1.1作为基于面向对象技术的标准建模语言。
- UML是一种可视化的建模语言，能够让系统构造者用标准的、易于理解的方式建立起能够表达他们设计思想的系统蓝图，并且提供一种机制，以便于不同的人之间有效地共享和交流设计成果。
- UML代表了面向对象方法的软件开发技术的发展方向，具有巨大的市场前景。

6.8 UML简介*

- 6.8.2 UML的构成
- 作为一种对客观系统的建模语言，UML提供了描述事物实体、性质、结构、功能、行为、状态和关系的建模元素，并通过一组图来描述由建模元素所构成的多种模型。
- 1、UML视图
- UML提供4种视图来展示软件在开发过程的不同阶段的模型，这4种视图作为4个视角，从不同侧面展现软件，使人们对软件有一个全面地把握。如图6.25所示，4种视图分别是用例视图、逻辑视图、构件视图和部署视图。

6.8 UML简介*

- 图6.25 UML视图



6.8 UML简介*

- 用例视图：

是向用户和开发人员展现的视图，主要展现软件能够外部提供的功能，所以，用例视图也被称为功能视图。用例视图用于描述软件和信息系统的需求，对需求进行建模。用例视图包括包图、用例图、类图、活动图和状态图等。其中包图用来对需求结构进行建模。用例图是用例视图中最重要的一种图，用来描述系统功能。在用例视图中有时也需要用类图来描述业务对象的关系，用活动图描述一些事务的处理流程，用状态图描述复杂业务对象的状态及其变化。

6.8 UML简介*

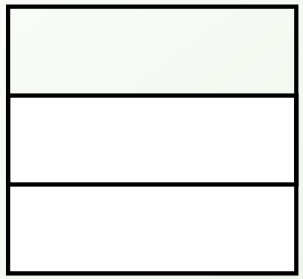
- 逻辑视图：描述软件和信息系统逻辑结构和逻辑组成，是对系统的分析和设计的建模。逻辑视图中包括包图、类图、顺序图、活动图和状态图等。包图用来建立软件的体系结构；类图描述系统的各种实体类、界面类和控制类的组成、关系和结构；顺序图用来描述为实现用例中规定的功能、系统相关构成要素之间的动态消息联系；活动图可以用来描述类中操作的算法流程；状态图用来描述类的状态。
- 构件视图：是对软件的实现建模。描述软件的构件以及构件之间的相关关系。
- 部署视图：描述软件和信息系统硬件的物理配置和结点布局。

6.8 UML简介*

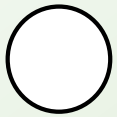
- 2、UML模型元素
- UML模型元素是模型的基本要素，每个模型元素有其确定的含义，称为模型元素的语义。模型元素有名字和表示模型元素的符号，并且应该遵循确定的建模规则。同一个模型元素可以出现在不同的图中。
- UML中的模型元素可以分为结构类、行为类、分组类和注释类4种类型。
- （1）结构类模型元素：用来描述软件模型中的静态要素，UML共定义了7种结构类模型元素，如图6.26所示。

6.8 UML简介*

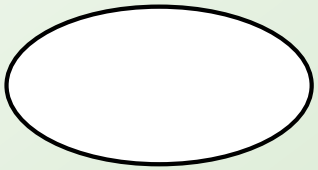
- 图6.26 UML的结构类模型元素



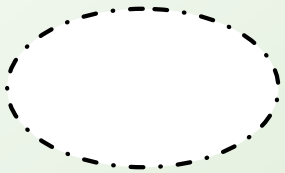
类和主动类



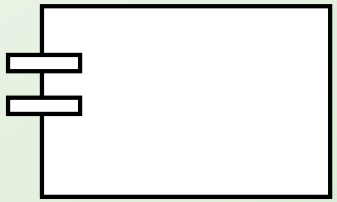
接口



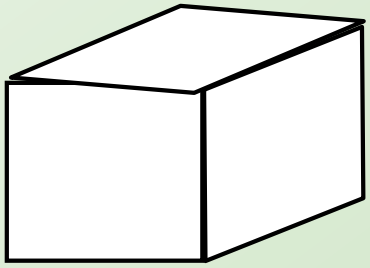
用例



协作



构件



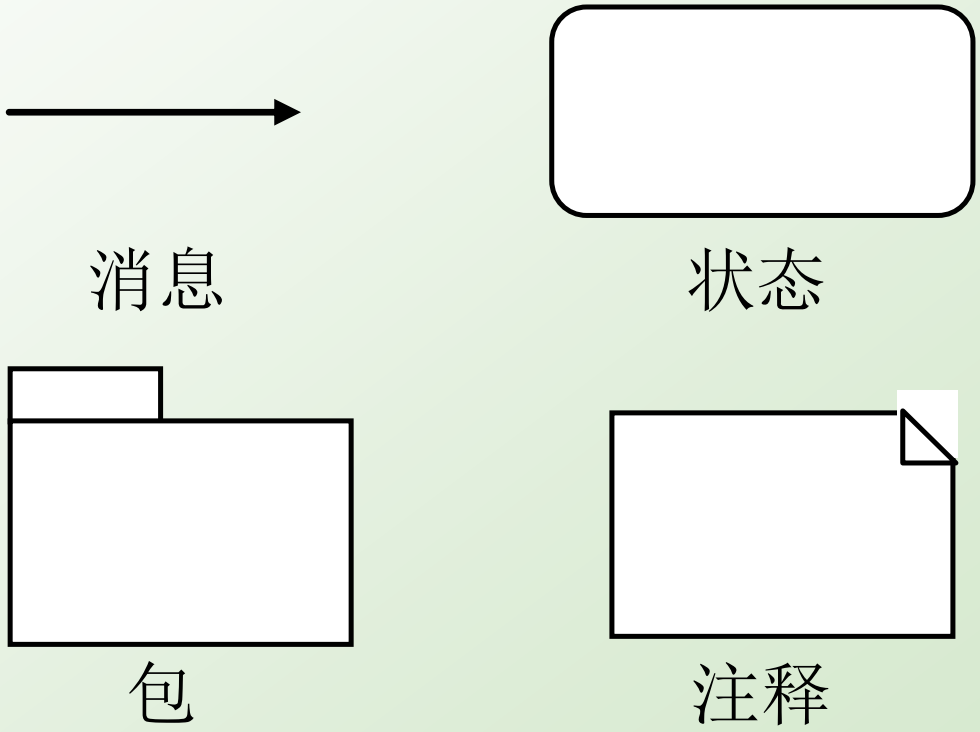
结点

6.8 UML简介*

- （2）行为类模型元素。如图6.27所示，行为类模型元素用来描述动态行为，UML提供了交互和状态机两个行为类模型元素。
- ①交互：描述对象之间交互的消息，通过消息来传递一个交互信息。
- ②状态机：一个对象在其生命周期中所经历的状态序列，状态机涉及状态、转移和事件。
- （3）分组类模型。UML中分组模型元素是包，通过包来把若干个模型元素组织成为一个模型。
- （4）注释类模型元素。用来说明和标注其他模型元素。注释类模型元素只有一个，用折角的矩形框表示。

6.8 UML简介*

- 图6.27 UML的行为类模型元素

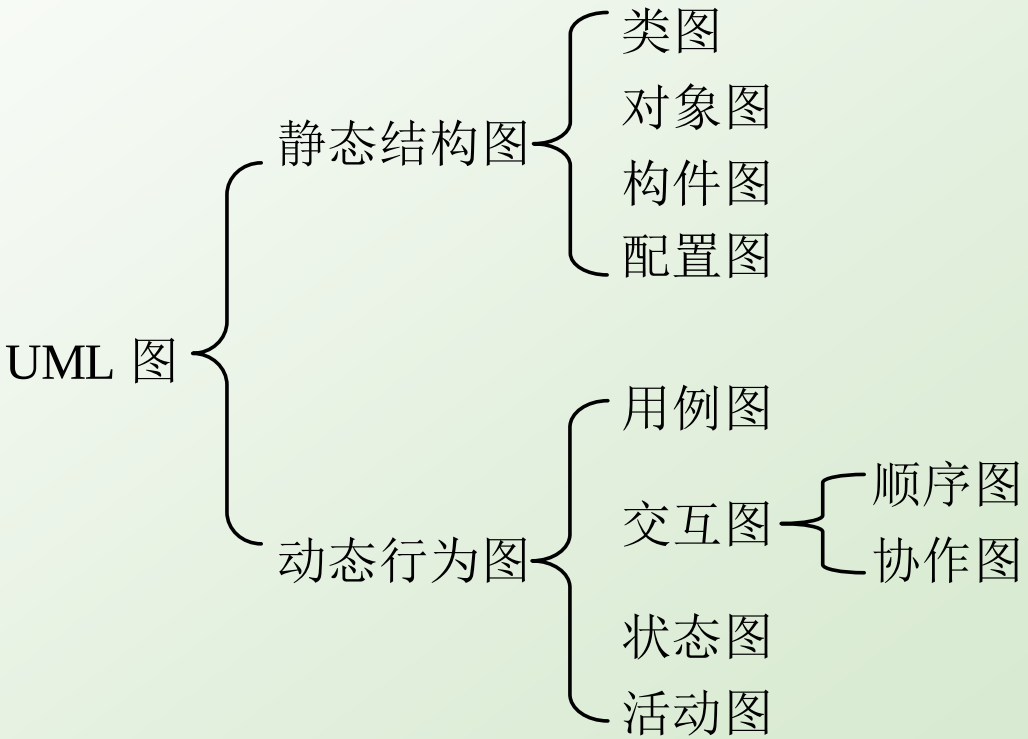


6.8 UML简介*

- 3、UML模型元素之间的关系
- UML定义了模型元素之间的关系共包括关联关系、泛化关系、依赖关系和实现关系，这几种关系的具体含义将在下面相应章节中介绍。
- 4、图
- 如图6.28所示是一组模型元素的图形表示，用来描述一个具有确定含义的子模型。一个信息系统的模型应该包括多种图，UML共定义了10种图：用例图、类图、对象图、顺序图、协作图、状态图、活动图、构件图、配置图、交互图和包图等。

6.8 UML简介*

- 图6.28 UML定义的图



6.8 UML简介*

- 6.8.3 UML的系统开发思路及开发阶段
- 1、开发思路
- 从应用的角度看，当采用面向对象技术设计系统时，第一步是描述需求；第二步是根据需求建立系统的静态模型，以构造系统的结构；第三是描述系统的行为。其中在第一步与第二步中所建立的模型都是静态的，包括用例图、类图（包括包）、对象图、组件图和配置图五个图形，是标准建模语言UML的静态建模机制。其中第三步中所建立的模型或者可以执行，或者表示执行时的时序状态或交互关系。它包括状态图、活动图、顺序图和合作图四个图形，是标准建模语言UML的动态建模机制。因此，标准建模语言UML的主要内容也可以归纳为静态建模机制和动态建模机制两大类。

6.8 UML简介*

- 在需求分析阶段，可以用用例来捕获用户需求。通过用例建模，描述对系统感兴趣的外部角色及其对系统（用例）的功能要求。分析阶段主要关心问题域的主要概念（如抽象、类和对象）和机制，需要识别这些以及它们相互间的关系，并用UML类图来描述。为实现用例，类之间需要协作，这可以用UML动态模型来描述。在系统分析阶段，只对问题域的对象（现实世界的概念）建模，而不考虑定义软件系统中技术细节的类（如处理用户接口、数据库、通信和并行性等问题的类）。这些技术细节将在系统设计阶段引入，因此设计阶段为编码阶段提供更详细的规格说明。

6.8 UML简介*

- 2、开发阶段
- 因此，基于UML的系统开发可分为5个阶段：需求分析、系统分析、系统设计、编码和测试阶段，具体说明如下：
- （1）需求分析阶段：UML语言利用用例来捕获客户的需求。通过用例模型，就可以使那些对系统感兴趣的外部参与者与他们要求系统具备的功能（即用例）一起被建模。外部参与者和用例之间是通过关系建模的，并且相互之间存在通信关联，或者被分解为更具体的层次结构。参与者和用例是由UML的用例图来描述的。每一个用例都是用文本进行描述的，它确定了客户的需求，即在不考虑功能如何实现的情况下客户所企盼的系统功能。需求分析不仅是软件系统的需求，业务过程同样也需要。
- （2）系统分析阶段：分析阶段关注的是出现在问题域中的主要抽象（类和对象）和机制。被建模的类以及类之间的关系在UML的类图中被明确指定和描述。为了实现用例，各类之间需要相互协作。这种协作是由UML中的动态模型描述的。在分析阶段，只有在问题域（现实世界的概念）中的类才被建模，这里的类并不是那些在软件系统中定义了细节和解决方案的技术类。此阶段的类有用户界面类、数据库类、通信类和并发类等。

6.8 UML简介*

- （3）系统设计阶段：在设计阶段，分析阶段的结果被扩展为一个技术解决方案。新类被加入进来，以提供以下一些技术基础结构：用户界面、处理对象存储的数据库、与其他系统的通信、与系统中各种设备的接口等。在分析阶段获得的问题域中的类被“嵌入”到此技术基础结构中，这样就能够同时改变问题域和基础结构。设计阶段将为随后的构建阶段产生详细的规格说明。
- （4）编码阶段：在编码阶段（或称为构建阶段），设计阶段的类被转换为使用面向对象编程语言编制（不推荐使用过程语言）的实际代码。这一任务可能比较困难，也可能比较容易，主要取决于所使用的编程语言本身的能力。用UML创建分析模型和设计模型时，最好避免试图将模型转换为代码。在开发的早期阶段，模型是帮助理解和搭建系统结构的一种手段。这样，如果在早期阶段就考虑代码，势必达不到预期的目标，即创建简单的和正确的模型。所以，编码是一个单独的阶段，也就是只有到了编码阶段，模型才被转换为代码。

6.8 UML简介*

- （5）测试阶段：通常，一个系统需要经过单元测试、集成测试、系统测试和接受性测试。单元测试是对单个类或一组类的测试，一般情况下由编程者自己完成。集成测试是集成组件和类，以校验它们是否是像指定的那样进行合作。系统测试将系统看做一个“黑盒子”，检验系统是否具有最终用户所期望的功能。接受性测试与系统测试相似，它是由客户实施的，以验证系统是否满足客户的需求。不同的测试团队使用不同的UML图作为他们工作的基础；单元测试团队使用类图和类规格说明；集成测试团队一般使用组件图和协作图；系统测试团队则利用用例图检验最初在这些图中定义的系统行为。

6.8 UML简介*

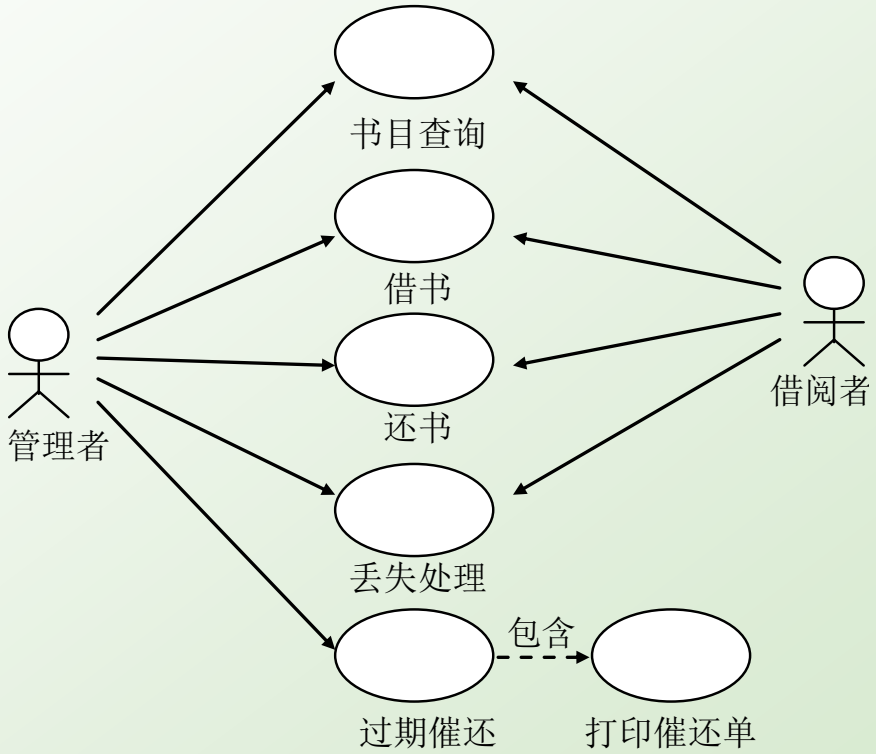
- 6.8.4 用例图
- 1、用例的概念
- 用例（Use Case）是外部使用者与系统之间，为达到确定目的所进行的一次交互活动。使用者（也被称为参与者或活动者）向系统提供某些交互要求，系统向使用者反馈所要的结果。在信息系统模型中，用例被用在需求分析中，用来描述系统的功能。

6.8 UML简介*

- 2、用例图
- （1）用例图：用例图（Use Case Diagram）用来描述软件系统向一组参与者提供的一组相关的功能。在一个用例图中，有一个或多个参与者与一个或多个用例相互关联。一个系统的全部用例图构成该系统的功能需求模型的需求。图2-14是图书借阅管理的用例图。在用例图中，小人表示与系统进行交互的参与者。椭圆表示用例，椭圆中或在椭圆下面填写用例名。参与者与用例之间的连线表示参与者与系统之间的交互关系。用例之间存在着泛化、包含和扩展关系。

6.8 UML简介*

- 图6.31 图书借阅用例图

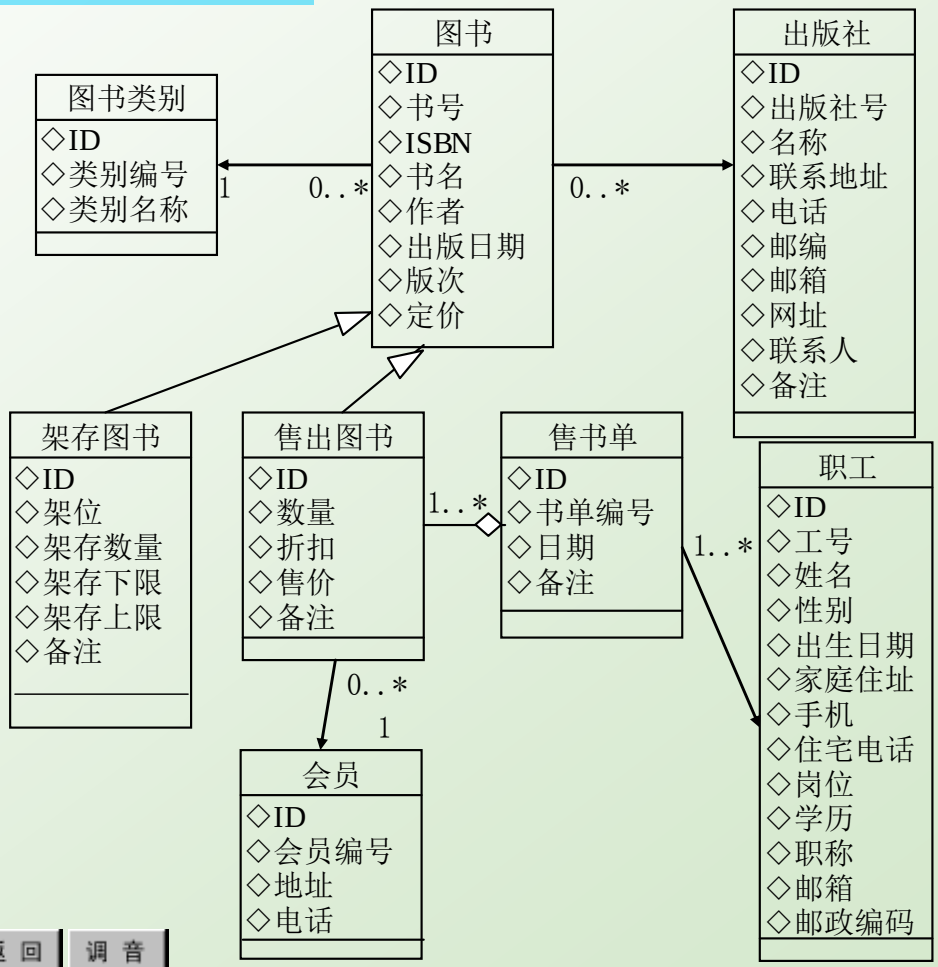


6.8 UML简介*

- 6.8.5 类图
- 1、概述
- 类图（Class Diagram）用来描述系统中的一些要素的静态结构。一个类图由一组类以及它们之间的关系构成。类描述事物以及事物的静态和动态性质，类的关系反映事物之间的联系，主要有关联关系、泛化关系、依赖关系和实现关系等。图6.36是一个书店销售管理类图的例子。
- 2、对象和类
- （1）对象的概念.：对象是系统中用来描述客观事物的一个实体，它是构成系统的一个基本单位，一个对象由一组属性和操作组成。对象既可以描述一个客观中存在的事物，也可以表示一个抽象概念。对象的属性描述它所反映的客观事物的静态性质，对象的操作描述客观事物的动态性质。对象是系统中一个独立实体。例如，一张桌子，一辆汽车，一个学生都是对象。

6.8 UML简介*

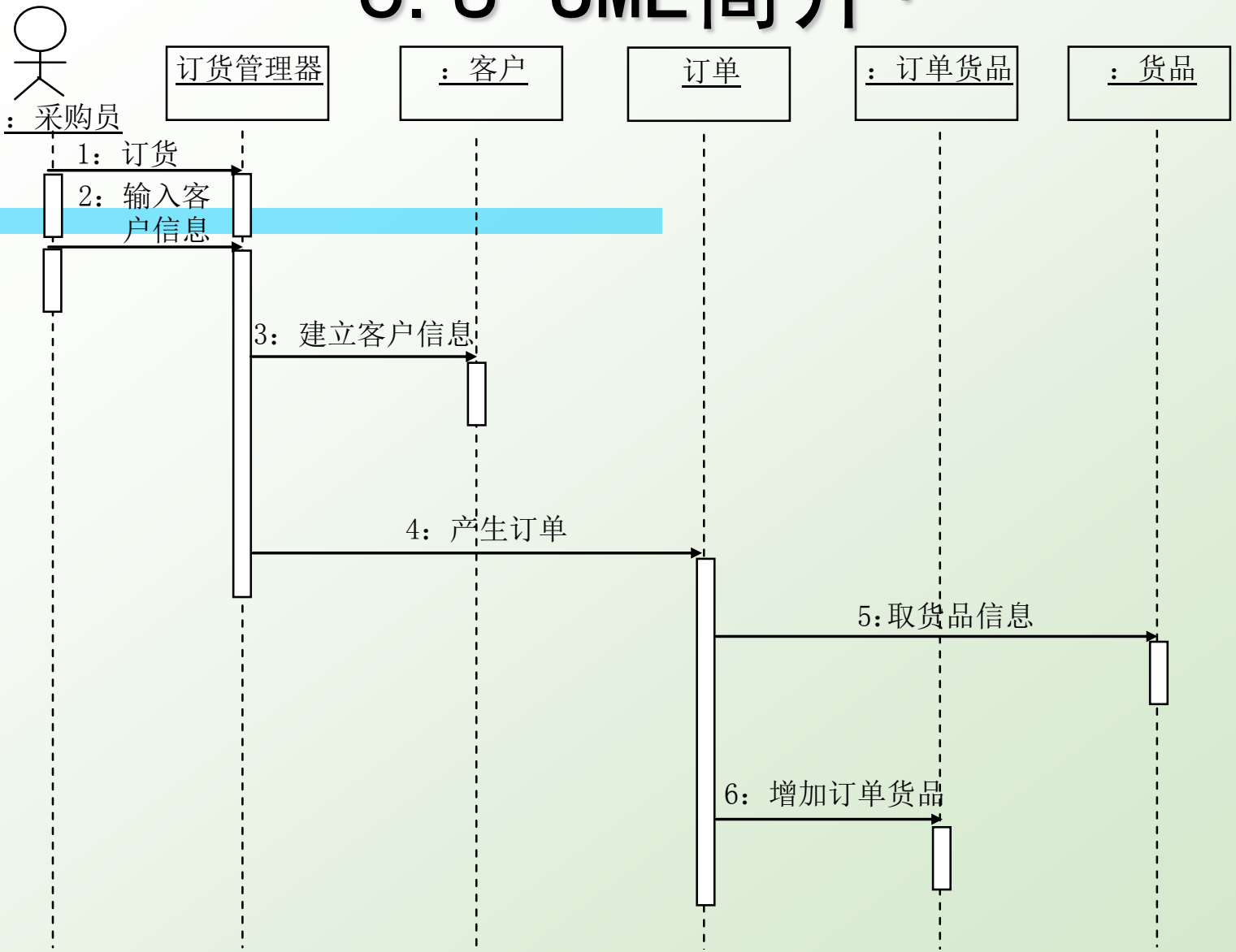
- 图6.36 书店销售管理类图



6.8 UML简介*

- 6.8.6 交互图
- 1、概述
- 交互图描述一组对象合作完成一项工作时，相互之间传递消息的交互过程。交互图反映在系统运行过程中，对象之间的动态联系活动，以帮助人们理解和把握内部各对象之间的动态协作关系。
- 交互图分为顺序图和协作图两种形式。顺序图反映各对象之间的消息传送顺序，重点描述对象之间交互的时序关系。协作图反映为完成一件工作所参与的对象，以及对象之间的消息联系。
- 2、顺序图
- 顺序图反映对象之间消息传送的时序关系。顺序图右对象、对象生命线、对象激活期和对象之间传输的消息等图形要素构成，图6.50是一个描述订货管理工作的顺序图。
- 在顺序图中，参与交互活动的对象用矩形框表示。在框中标注对象名，也可以采用匿名对象。对象的生命线表示对象对象的存活期，在对象下面用一条虚线表示。在对象生命线上的窄条为对象的激活期，表示对象在生存期内处在激活状态。消息是对象之间的通信信息，用带箭头的线段表示一个对象传送给另一个对象的消息。在消息上要注明消息名。虚线表示消息的返回。

6.8 UML简介*



首 页 | 上一页 | 下一页 | 尾 页 | 结 束 | 返 回 | 调 音

图6.50 订货管理的顺序图

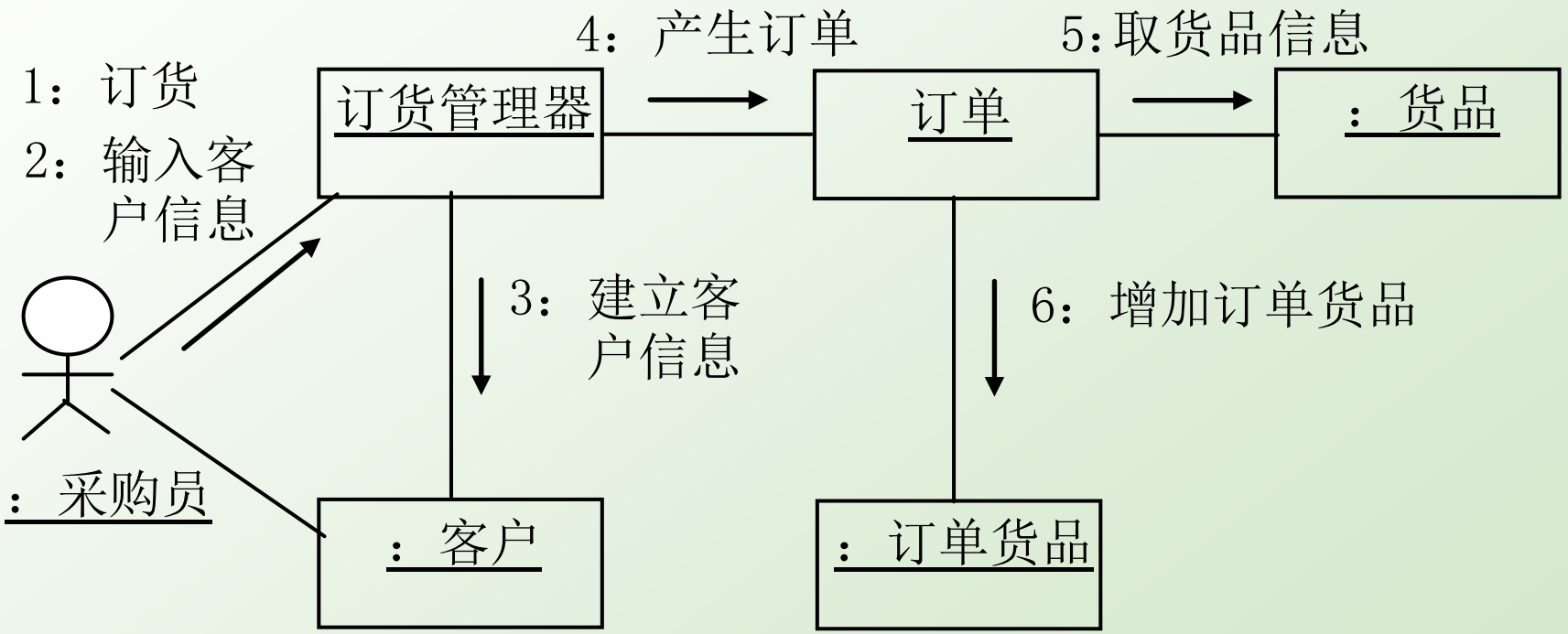
返回本节首页

6.8 UML简介*

- 3、协作图
- 协作图描述一组对象之间消息交互关系的协作结构。在协作图中，对象作为结点。存在消息交互关系的对象之间直线连接，并在直线上标注交互的消息名。图6.51是订货管理的协作图。
- 协作图与顺序图是等价的，包含的信息量相同，其差别是描述得角度不同。顺序图侧重于反映对象之间的消息交互时序，协作图重点描述对象之间的消息交互结构。

6.8 UML简介*

- 图6.51 订货管理的协作图

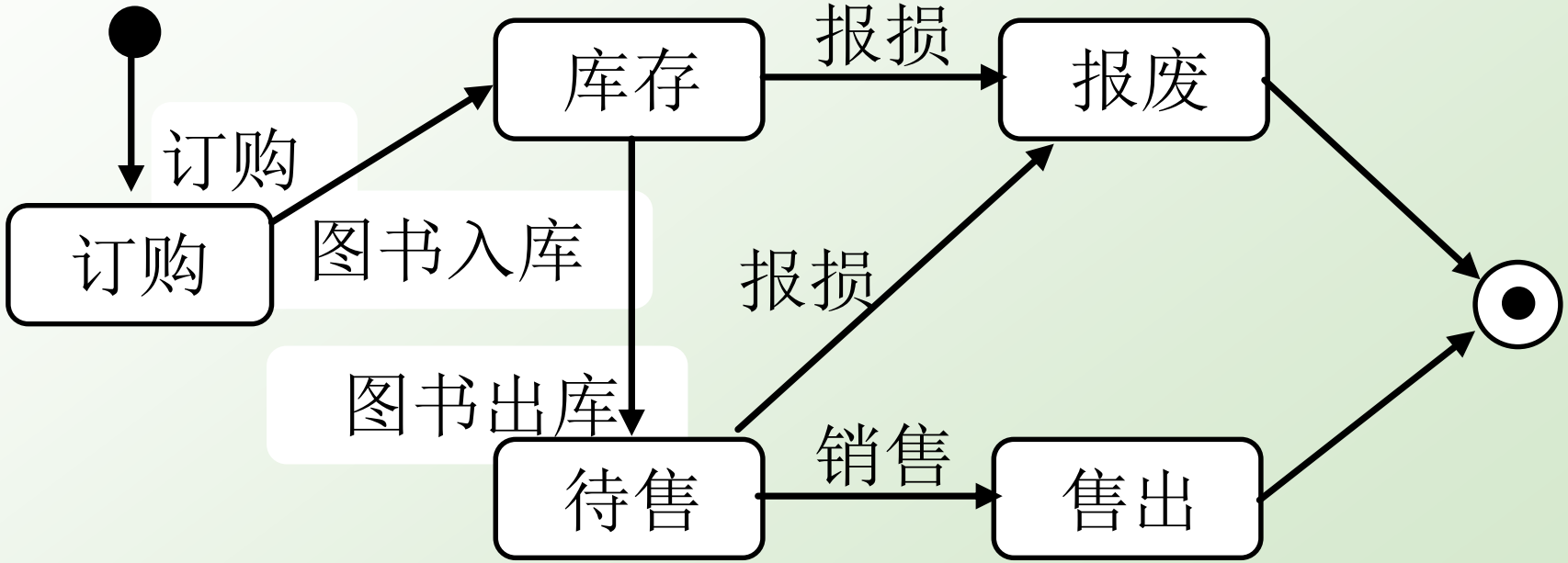


6.8 UML简介*

- 6.8.7 状态图
- 状态图描述事物对象在其生存周期中所具有的各种状态，以及因事件激励状态的变化和相互关系。图6.52是书店图书的状态图。书店的图书要经过订购、库存、待销售、售出或报废等状态。
- 状态图中的结点是事物所处的状态。实心圆表示初始状态，带圆圈的实心圆表示结束状态。一副状态图中一般有一个初始态。箭头表示状态的切换，在箭头上标注状态切换的激发条件。例如，图6.52通过图书入库这个激发条件把图书从订购状态转换为库存状态。

6.8 UML简介*

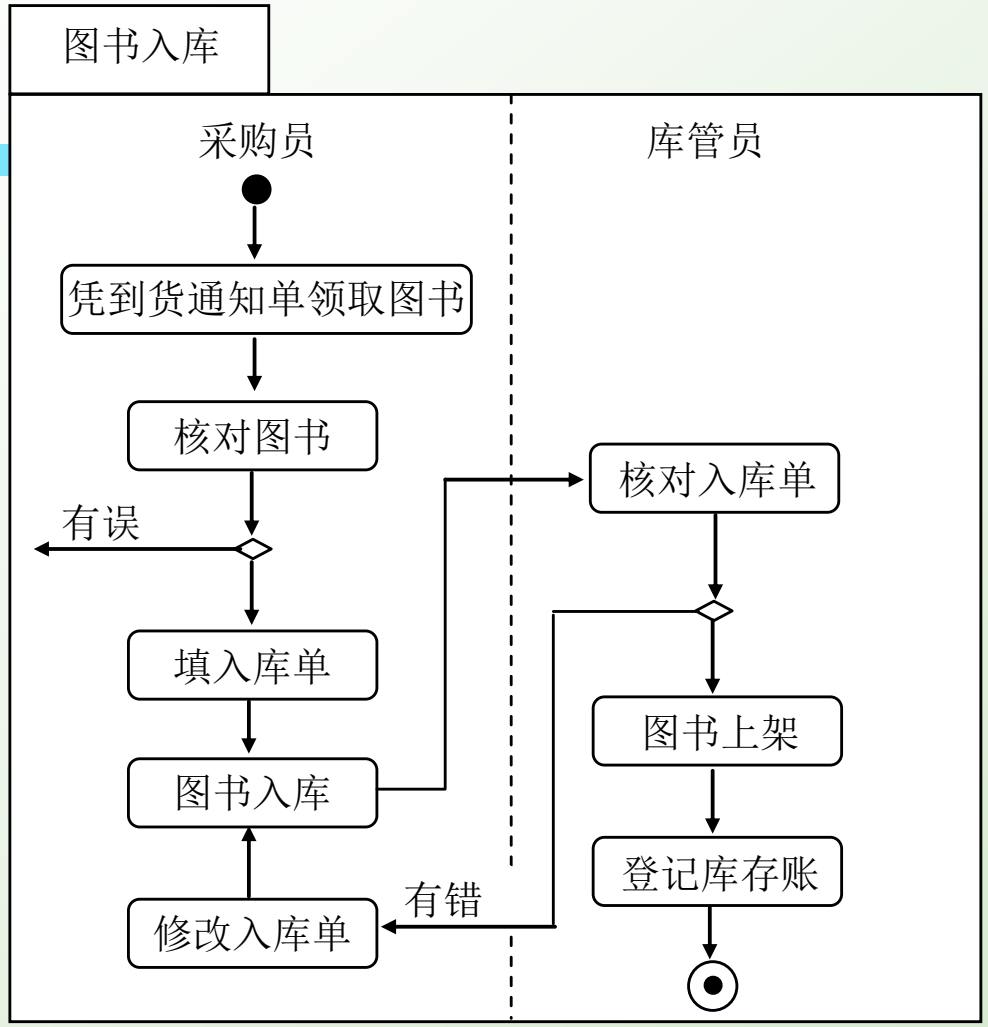
- 图6.52 书店图书状态



6.8 UML简介*

- 6.8.8 活动图
- 活动图用来描述事物发展变化的过程。活动图可以描述业务流程、工作流程、类中的操作流程等。图6.53是反映书店图书入库业务流程的活动图。采购员凭到货通知单到车站或邮局领取图书，并把到货的图书与图书订单进行核对，检查是否存在偏差。如果有偏差，则与运输部门或邮局进行联系；如果没有偏差，则领回图书。采购员领回图书之后，填写图书入库单，然后持入库单到书库入库。库管员把图书与入库单进行核对，如果发现有误，则请采购员修改入库0单。如果核对无误，库管员登记入库账，并把入库图书收库，入库过程结束。
- 在活动图中，要表示出活动的开始和结束。圆形框表示活动，菱形框表示检查；用虚线隔开的两个部分称为泳道，表示两个实体所进行的活动，如图6.53。

6.8 UML简介*



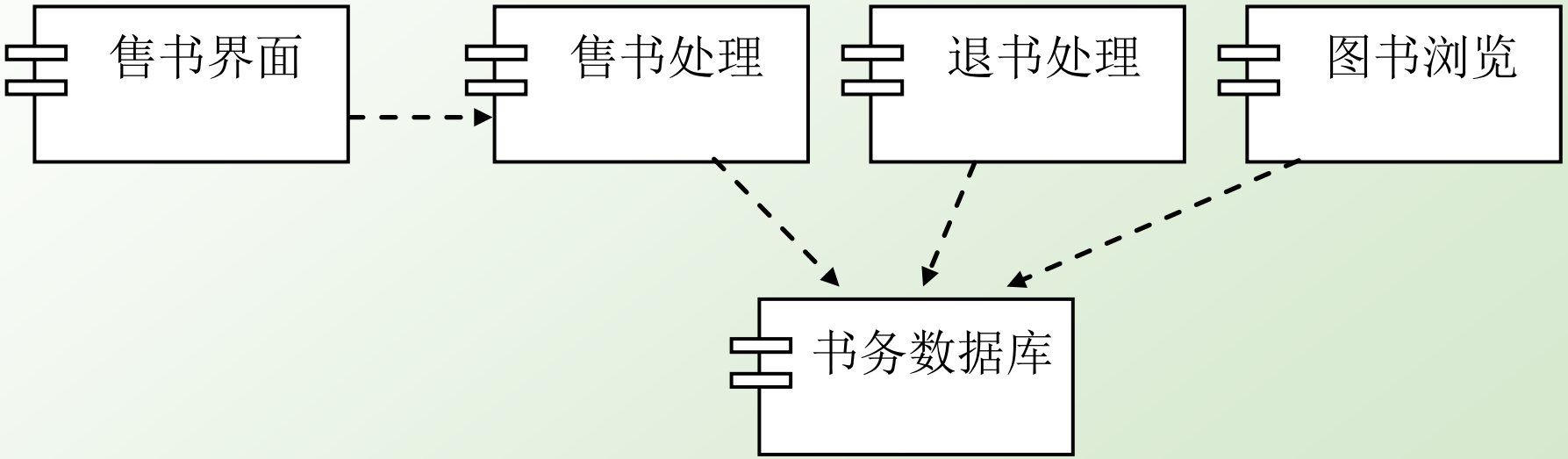
- 图6.53 图书入库的活动图

6.8 UML简介*

- 6.8.9 构件图
- 构件可以是一段源程序代码、一个文本文件、一个二进制文件或一个可执行文件。基于构件开发的软件系统，由多个软件构件按照确定的关系构成软件系统。构件图用来描述构成软件系统的软构件以及它们之间的相互依赖关系。
- 在构件图中，带两个小棒的矩形框表示构件。构件之间的带箭头的虚线表示构件之间的依赖关系。图6.54是一个构件图的例子。该图包括售书界面、售书处理、退出处理、图书浏览和书务数据库5个构件。

6.8 UML简介*

- 图6.54 构件图



6.8 UML简介*

- 6.8.10 配置图
- 配置图反映系统的物理结点的分布，以及各结点中构建的构成情况。图6.55是图书销售管理的配置图。

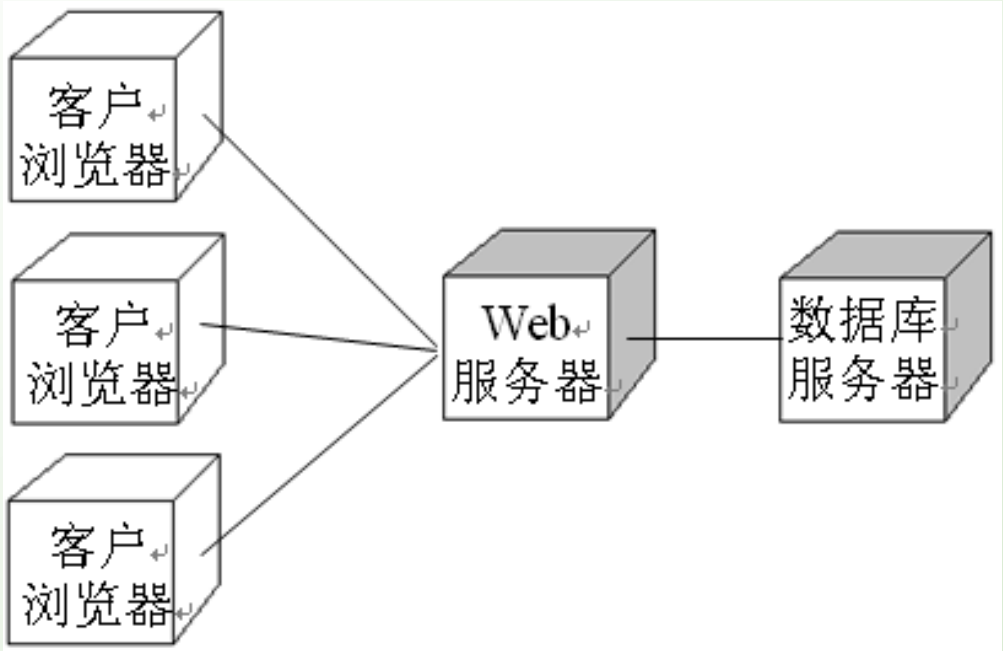


图6.55 配置图

6.8 UML简介*

- 6.8.11 包图
- 1、包的含义
- 包是模型的组织单位。一个复杂的系统模型需要分解成为多个部分，每一部分用包来表示。包是UML一种模型元素，可以用来表示模型、子模型、系统和子系统等系统模型单位。包表示为图6.56的形式。如果包中的内容被隐藏，包名就注在包的中间。如果包表示的内容要展现，包名写在上面的小框中。
- 2、包间的关系
- 包与包之间可以存在依赖关系。图6.57描述教学管理系统中三个子系统的包之间的依赖关系。

6.8 UML简介*

- 图6.56 包

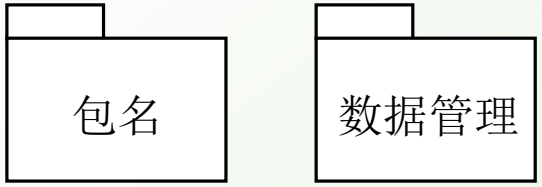
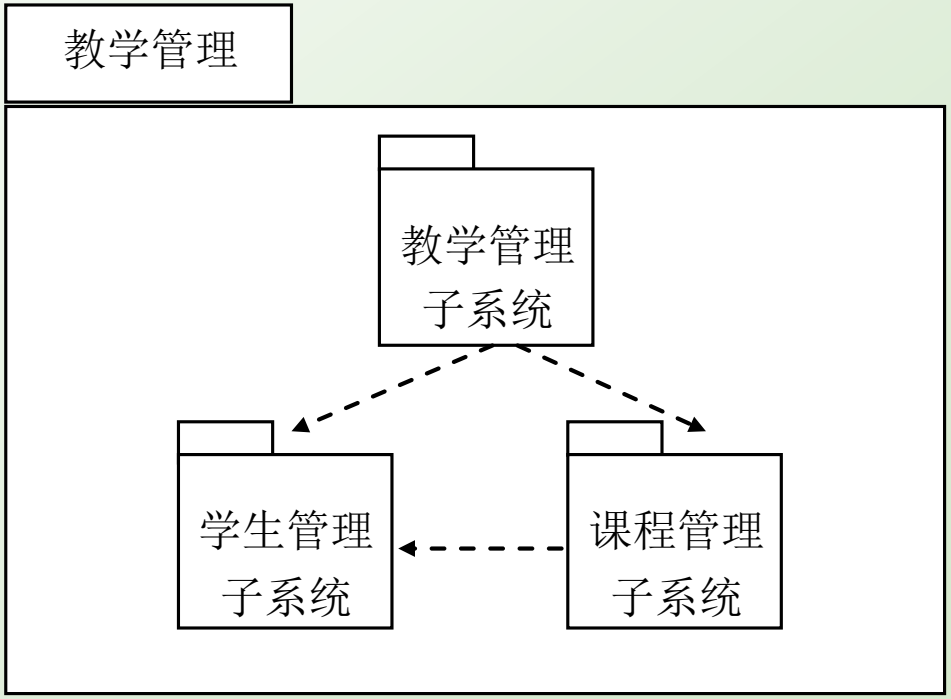


图6.57 包的依赖



6.8 小结

数据库设计这一章主要讨论数据库设计的方法和步骤，本章介绍了数据库设计的六个阶段：系统需求分析、概念结构设计、逻辑结构设计、物理设计、数据库及应用系统的实施、数据库及应用系统运行与维护。其中重点的是概念结构设计和逻辑结构设计，这也是数据库设计过程中最重要的两个环节。本章还简要介绍了UML的基本知识。

学习本章，要努力掌握书中讨论的基本方法和开发设计步骤，特别要能在实际的应用系统开发中运用这些思想，设计符合应用要求的数据库应用系统。

习　题

一、选择题

1、下列对数据库应用系统设计的说法中正确的是( )。

A．必须先完成数据库的设计，才能开始对数据处理的设计

B．应用系统用户不必参与设计过程

C．应用程序员可以不必参与数据库的概念结构设计　　D．以上都不对

2、在需求分析阶段，常用( )描述用户单位的业务流程。

A．数据流图　　B．E-R图　　C．程序流图　　D．判定表

3、下列对E-R图设计的说法中错误的是( )。

A．设计局部E-R图中，能作为属性处理的客观事物应尽量作为属性处理

B．局部E-R图中的属性均应为原子属性，即不能再细分为子属性的组合

C．对局部E-R图集成时既可以一次实现全部集成，也可以两两集成，逐步进行

D．集成后所得的E-R图中可能存在冗余数据和冗余联系，应予以全部清除

习　题

4、下列属于逻辑结构设计阶段任务的是( )。

A．生成数据字典　　B．集成局部E-R图

C．将E-R图转换为一组关系模式　　D．确定数据存取方法

5、将一个一对多联系型转换为一个独立关系模式时，应取( )为关键字。

A．一端实体型的关键属性　　B．多端实体型的关键属性

C．两个实体型的关键属性的组合　　D．联系型的全体属性

6、将一个M对N（M>N）的联系型转换成关系模式时，应( )。

A．转换为一个独立的关系模式　　B．与M端的实体型所对应的关系模式合并

C．与N端的实体型所对应的关系模式合并　　D．以上都可以

习　题

7、在从E-R图到关系模式的转化过程中，下列说法错误的是( )。

A．一个一对一的联系型可以转换为一个独立的关系模式

B．一个涉及三个以上实体的多元联系也可以转换为一个独立的关系模式

C．对关系模型优化时有些模式可能要进一步分解，有些模式可能要合并

D．关系模式的规范化程度越高，查询的效率就越高

8、对数据库的物理设计优劣评价的重点是( )。

A．时空效率 B．动态和静态性能 C．用户界面的友好性 D．成本和效益

9、下列不属于数据库物理结构设计阶段任务的是( )。

A．确定选用的DBMS　　B．确定数据的存放位置

C．确定数据的存取方法　　D．初步确定系统配置

10、确定数据的存储结构和存取方法时，下列策略中( )不利于提高查询效率。

A．使用索引　　B．建立聚簇　　C．将表和索引存储在同一磁盘上

D．将存取频率高的数据与存取频率低的数据存储在不同磁盘上

习　题

二、填空题

1、在设计分E-R图时，由于各个子系统分别面向不同的应用，所以各个分E-R图之间难免存在冲突，这些冲突主要包括_____、_____和_____三类。

2、数据字典中的_____是不可再分的数据单位。

3、若在两个局部E-R图中都有实体“零件”的“重量”属性，而所用重量单位分别为公斤和克，则称这两个E-R图存在_____冲突。

4、设有E-R图如图6.59，其中实体“学生”的关键属性是“学号”，实体“课程”的关键属性是“课程编码”，设将其中联系“选修”转换为关系模式R，则R的关键字应为属性集_____。

习　题

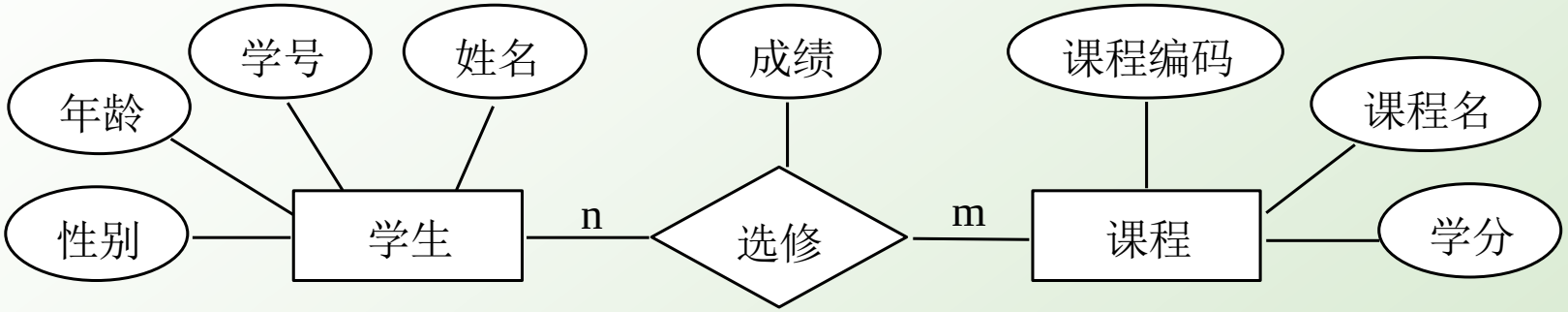


图6.59 E-R图

5、确定数据库的物理结构主要包括三方面内容，即：_____、_____和_____。

6、将关系R中在属性A上具有相同值的元组集中存放在连续的物理块上，称为对关系R基于属性A进行_____。

7、数据库设计的重要特点之一要把_____设计和_____设计密切结合起来，并以_____为核心而展开。

习　题

8、数据库设计一般分为如下六个阶段：需求分析、_____、_____、数据库物理设计、数据库实施、数据库运行与维护。

9、概念设计的结果是得到一个与_____无关的模型。

10、在数据库设计中，_____是系统各类数据的描述的集合。

三、简答题

1、数据库设计分为哪几个了阶段？每个阶段的主要工作是什么？

2、在数据库设计中，需求分析阶段的设计目标是什么？调查的内容主要包括哪几个方面？

3、数据库设计的特点是什么？

4、什么是数据库的概念结构？试述概念结构设计的步骤

习　题

5、什么是E-R图？构成E-R图的基本要素是什么？

6、用E-R图表示概念模式有什么好处？

7、局部E-R图的集成主要解决什么问题？

8、一个图书馆理系统中有如下 信息：

图书：书号、书名、数量、位置

借书人：借书证号、姓名、单位

出版社：出版社名、邮编、地址、电话、E-mail

其中约定：任何人可以借多种书，任何一种书可以被多个人借，借书和还书时，要登记相应的借书日期和还书日期；一个出版社可以出版多种书籍，同一本书仅为一个出版社所出版，出版社名具有惟一性。

根据以上情况，完成如下设计：

（1）设计系统的E-R图；

（2）将E-R图转换为关系模式；

（3）指出转换后的每个关系模式的关系键。

习　题

9、有如下运动队和运动会两个方面的实体：

（1）运动队方面

运动队：队名、教练姓名、队员姓名。

队员：队名、队员姓名、性别、项名。

其中，一个运动队有多个队员，一个队员仅属于一个运动队，一个队有一个教练。

（2）运动会方面

运动队：队编号、队名、教练姓名。

项目：项目名、参加运动队编号、队员姓名、性别、比赛场地。

其中，一个项目可由多个队参加，一个运动员可参加多个项目，一个项目一个比赛场地。

请完成如下设计：

① 分别设计运动队和运动会两个局部E-R图。

② 将它们合并为一个全局E-R图

③ 合并时存在什么冲突，你是如何解决这些冲突的？