第2章　数据库应用系统生命周期

第1章给出了数据库系统（DBS）和数据库应用系统（DBAS）的定义。二者之间的差别在于DBS主要提供应用数据的组织、存储、维护、访问等数据管理功能，而DBAS则不仅为用户提供数据管理功能，还根据具体应用领域业务处理逻辑，通过数据库应用程序，实现了更为复杂的业务数据处理功能。

数据库应用系统设计与开发是指根据具体应用领域数据管理和处理需求，设计应用数据在数据库中的组织和存储方式，即设计数据库模式，并且根据应用领域业务处理逻辑设计数据库应用软件。依据这些设计结果，在数据库中正确组织和存储应用数据，开发数据库应用程序，选定合适的硬件平台和操作系统、DBMS等系统软件，按照一定的体系结构将各部分有机组合起来，构成实际可运行的数据库应用系统。

作为面向数据处理的复杂软件系统，数据库应用系统的设计开发既要遵循数据库系统三级模式结构所规定的数据库设计范型，也要符合软件工程所定义的复杂软件系统开发基本原则。为此，本章提出了数据库应用系统生命周期模型，该模型规定了数据库应用系统设计、开发和运行维护各阶段的主要目标、工作内容和采用的关键技术。

本章给出了一个商场经营管理系统实例，作为本教材后续各章节讨论的各种DBAS设计与开发技术的应用案例。

2.1 软件工程与软件开发方法

2.1.1 软件工程概述

20 世纪 60 年代末至 70 年代初，大型复杂软件系统系统开发过程中的软件危机问题日益明显，严重影响到软件开发的质量、进度和成本，具体体现在：

（1）软件开发的成本和进度难以准确估计，延迟交付甚至取消项目的现象屡见不鲜。

（2）软件存在错误多、性能达不到要求、不可靠、安全性差等严重质量问题。

（3）软件成本占计算机系统整个成本的比例越来越大，最高可达 80%。

（4）软件维护更新十分困难，无法适应不断变化的用户需求和应用环境。

造成软件危机的原因有：

（1）用户对软件需求的描述不精确，可能有遗漏、二义性、错误；甚至在软件开发过程中，用户还不断提出修改软件功能、界面、支撑环境等方面的要求。

（2）软件开发人员对用户需求的理解与用户的本来愿望有差异，这种差异导致开发出来的软件产品与用户要求不一致。

（3）多人参与的大型软件项目开发中，管理人员与开发人员间、开发人员之间的信息交流不及时、不准确，有时还会产生误解。

（4）软件项目开发人员不能有效地、独立自主地处理大型软件的全部关系和各个分支，因此容易产生疏漏和错误

（5）个人或小组开发小型软件非常有效的编程技术和过程，在开发大型、复杂系统时难以发挥同样的作用。

（6）缺乏有力的方法学和工具方面的支持，过分地依靠程序设计人员在软件开发过程中的

技巧和创造性，加剧了软件产品的个性化。

为了提高软件质量、加快软件开发进度、降低开发费用，必须采用现代工程的概念、原理、技术和方法进行软件的开发、管理、维护和更新，由此诞生了计算机科学技术的一个新分支—软件工程。迄今为止，软件工程的研究与应用已经取得很大成就，它在软件开发方法、工具、管理等方面的应用大大缓解了软件危机造成的被动局面。

软件工程是指导计算机软件开发和维护的工程科学，它采用工程化的概念、原理、技术和方法，以及正确的项目管理技术，来开发和维护软件；它将系统性、规范化、定量化方法应用于软件的开发、操纵和维护，也就是将工程化（Engineering）应用于软件生产。

软件工程的目标是在给定成本、进度的前提下，开发出满足用户需求并具有下述特征的软件产品：可修改性、有效性、可靠性、可理解性、可维护性、可重用性、可适应性、可移植性、可追踪性和可互操作性。这些软件特征有助于提高软件产品的质量和开发效率，减少维护的困难。

软件生命周期是指软件产品从考虑其概念开始，到该产品不再使用的整个时期。一般包括概念阶段、需求阶段、设计阶段、实现阶段、测试阶段、安装部署及交付阶段、运行阶段和维护阶段。这些阶段可以重复，执行时也可以有迭代。

软件开发生命周期是指软件产品从考虑其概念开始，到该产品交付使用的整个时期，包括概念阶段、需求阶段、设计阶段、实现阶段、测试阶段、安装部署及交付阶段。

大型复杂软件系统的开发需要采用工程化的方法，需要有效的项目管理技术。**软件项目管理**是为了能使软件开发按预定的质量、进度和成本进行，而对成本、质量、进度、人员、风险等进行分析和有效管理的一系列活动。软件项目管理的主要任务是：制定项目实施计划；建立项目环境并组建项目开发团队，对人员进行分工组织；按照计划的进度及成本管理、质量管理和风险管理的要求，进行软件开发，最终完成软件项目规定的各项任务。软件项目管理由项目启动、项目规划、项目实施、项目收尾等活动组成。软件项目管理的主要内容包括成本管理、质量管理、风险管理和软件配置管理。

在软件工程活动中，软件开发人员和管理人员按照软件工程方法和原则，借助于计算机和软件工具，开发、维护、管理软件产品的过程称为**计算机辅助软件工程**（Computer-Aided Software Engineering, CASE），这一过程需要有效的 CASE 工具支持。典型的 CASE 工具有事务系统规划工具、项目管理工具、分析和设计工具、支撑工具、程序设计工具、测试工具、原型建造工具和维护工具等。

2.1.2 常用开发模型

软件工程以关注软件质量为特征，由方法、工具和过程三部分组成。其中，**软件（开发）过程**是指将用户的软件要求转变成软件产品的过程。

软件过程模型也称为软件开发模型，是对软件过程的一种抽象表示，表示了软件过程的整体框架和软件开发活动各阶段间的关系。下面介绍几种常见的软件开发模型。

1．瀑布模型

瀑布模型也称为软件生命周期模型，是由 Winston Royce 在 1970 年提出的经典的软件开发过程模型，在软件工程中占有重要地位，提供了软件开发的基本框架。

瀑布模型将软件过程划分为定义、开发、维护三个阶段。这三个阶段包括了计划、需求分析、软件设计、软件编码实现、软件测试、运行维护等一系列基本活动。在标准的瀑布模型中，这些活动以自上而下、相互衔接的固定次序，如同瀑布流水逐级而下。各项活动严格按照线性的方式进行，当前活动接受上一项活动的工作结果，实施并完成本活动的工作内容。当前活动的工作结果需要进行验证，如果验证通过，则该结果作为下一项活动的输人，继续进行下一项活动，否则返回进行修改。瀑布模型强调文档的作用，并要求每个阶段都要仔细

验证。

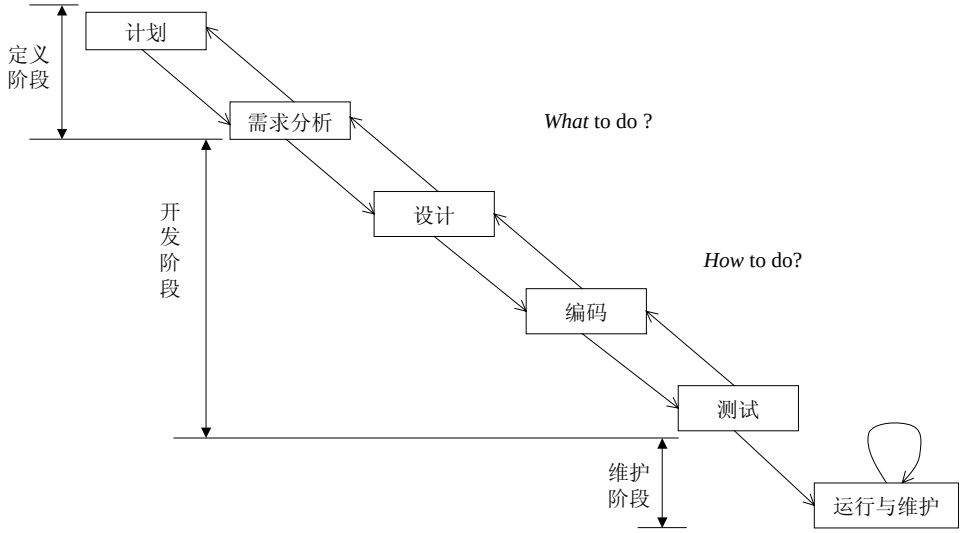

图 2.1 瀑布模型

瀑布模型是一种线性的过程，适用于在开发的早期阶段软件需求被完整确定的情况。显然这种要求过于理想化，难以适应现代软件的开发要求，其暴露出的主要问题在于：

（1）各个阶段的划分完全固定，阶段之间产生大量的文档，增加了工作量。

（2）由于开发模型是线性的，用户只有等到整个过程的末期才能见到开发成果，中间提出的变更要求很难得到响应，因此增加了软件开发风险。

（3）早期的错误可能后向传播扩散，等到开发后期才能发现，带来严重后果。

2．快速原型模型

快速原型方法可以有效降低开发初期由于软件需求不明确带来的开发风险。快速原型模型将软件开发分为如下步骤：

（1）快速构建一个可以运行的软件原型（Prototype），实现客户或未来用户与系统的交互，原型是对待开发软件系统的一种近似或模拟。

（2）由用户或客户对该原型进行评价，并进一步细化待开发软件的需求。经过逐步调整原型使其满足客户的要求之后，开发人员可以将客户的真正需求确定下来。

（3）在原型系统基础上，开发满足客户需求的软件产品。

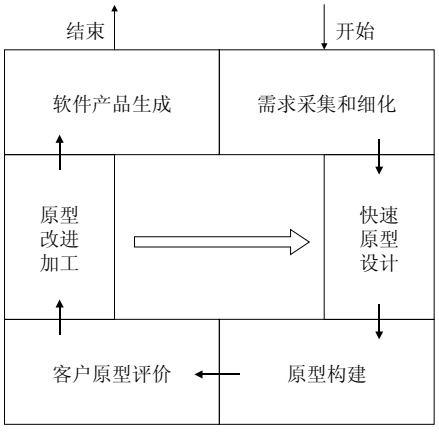

图 2.2 快速原型模型

快速原型的关键在于尽可能“快速”地构建原型，一旦确定了用户的真正需求，所构建的原型将被丢弃。因此，原型系统的内部结构并不重要，重要的是必须迅速建立原型，并迅速调整修改原型，以反映用户的需求。

软件原型是通过快速构造并不断修改而形成的，所选用的开发技术和工具不一定符合主流的发展，快速建立起来的系统结构加上连续的修改可能会导致软件产品质量低下。

3．增量模型

增量模型的目标是为了适应用户逐步细化需求的形成过程，以减少软件开发过程中的返

工。在增量模型中，软件被作为一系列的增量构件来设计、实现、集成和测试。

增量模型在各个阶段并不交付一个可运行的完整产品，而是交付满足用户需求的一个子集的可运行产品。第一个增量一般是实现基本需求的核心产品，核心产品交付用户使用后，经过评价形成下一个增量的开发计划，其中包括对核心产品的修改和一些新功能的发布。这个过程在每个增量发布后不断重复，直到产生最终的完善产品。

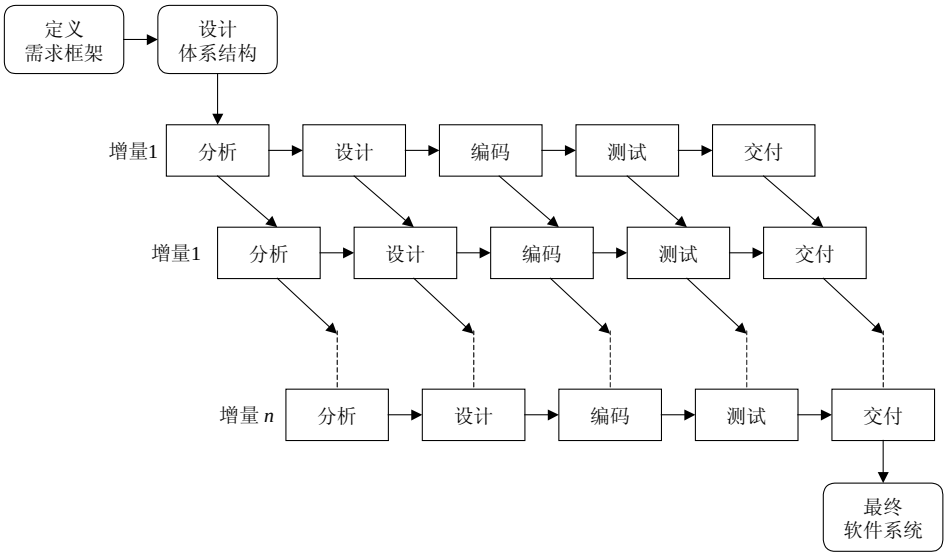

图 2.3 增量模型

增量模型将整个产品分解成若干个构件进行逐步开发和交付，因此软件开发可以较好地适应需求的变化，用户可以不断地看到所开发软件的可运行中间版本，从而降低了开发风险。但是，增量模型也存在以下缺陷：

（1）由于各个构件是逐渐并入已有的软件体系结构中，新加入的构件不能破坏已构造好的系统部分，因此软件系统需要具备开放体系结构。

（2）在开发过程中，需求的变化是不可避免的。增量模型的灵活性可以使其很好地适应需求的变化，但也很容易退化为边做边改的开发方式，从而失去对软件过程的整体控制。

4．螺旋模型

螺旋模型（Spiral model）是由 Barry Boehm 于 1988 年提出的。它将曝布模型与快速原型模型结合起来，强调了其它模型所忽视的风险分析，特别适合于大型复杂的软件系统。

螺旋模型的结构如图 2.4 所示，它是由需求定义、风险分析、工程实现、评审四个环节组成的迭代模型。软件开发过程每经过一个迭代周期，螺旋线就增加一周，软件开发又前进一个层次，系统又生成一个新版本，而软件开发的时间和成本又有了新的投入。通常情况下软件开发过程沿螺旋线路径持续，最后得到一个用户满意的软件版本。

理论上，迭代过程可以无休止地进行下去，是一个无限过程，但在实践中必须通过控制迭代次数，尽可能减少每次迭代的工作量，降低软件开发的时间和成本。

在螺旋模型每一个迭代周期中，首先确定该阶段目标、完成这些目标的方案及约束条件；其次从风险角度分析方案的开发策略，努力排除各种潜在的风险，在需求不明确的情况下可能需要构建原型系统；如果某些风险不能排除，该方案可能立即终止，否则继续启动下一步的软件开发和验证工作；最后，评价该阶段的结果，并规划下一个开发阶段。

螺旋模型保留了生存周期模型中系统地、按阶段逐步进行软件开发和“边开发、边评审”的风格，而且还引入了风险分析，并把制作原型作为风险分析的主要措施。用户一直关心、参与软件开发并对阶段性的软件产品提出评审意见，这有利于保证软件产品的质量是。

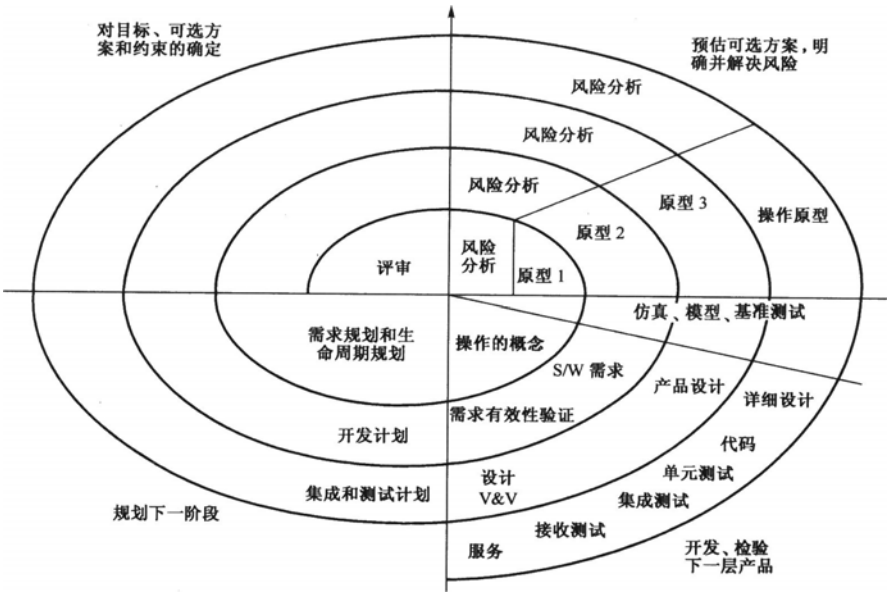

图 2.4 螺旋模型

2.2 数据库应用系统生命周期

2.2.1 DBAS 软件组成

一个数据库应用系统 DBAS 是面向某个特定应用领域、实现特定功能的计算机软件、硬件的集成体，它包括的软件有：

（1）以文件形式存储在磁盘等外设存储介质上的数据库 DB，DB 用于存储应用领域数据。

（2）数据库管理系统 DBMS，如关系数据库引擎 SQL Server、DB2、Oracle、Sybase 等。DBMS 通过查询处理与优化、存储管理、事务管理等机制，提供了对数据的查询、删除、插入、修改等基本管理功能。

（3）数据库应用软件。数据库应用软件根据具体应用领域的业务处理逻辑，实现了具体数据处理功能。数据库应用软件在内部可看作由一系列软件模块/子系统组成，这些模块/子系统可以分为 2 类：

a. 与数据访问有关的数据库事务模块。这些模块利用 DBMS 提供的数据管理功能，以数据库事务方式直接对数据库中的各类应用数据进行操作，模块粒度比较小。

b. 与数据访问无直接关联的应用模块。在许多与数据处理有关的应用系统中，对数据库中数据访问处理只是整个系统功能的一部分，其它许多功能则与数据库访问无直接关系，这部分功能由与数据访问无关的软件模块来实现，并且模块粒度可以比较大。这类软件模块最典型的例子是人机界面/交互模块，此外还包括对数据进行深层次加工处理的数据处理模块。

（4）支持系统运行的操作系统和其它系统软件（如开发工具、中间件等）

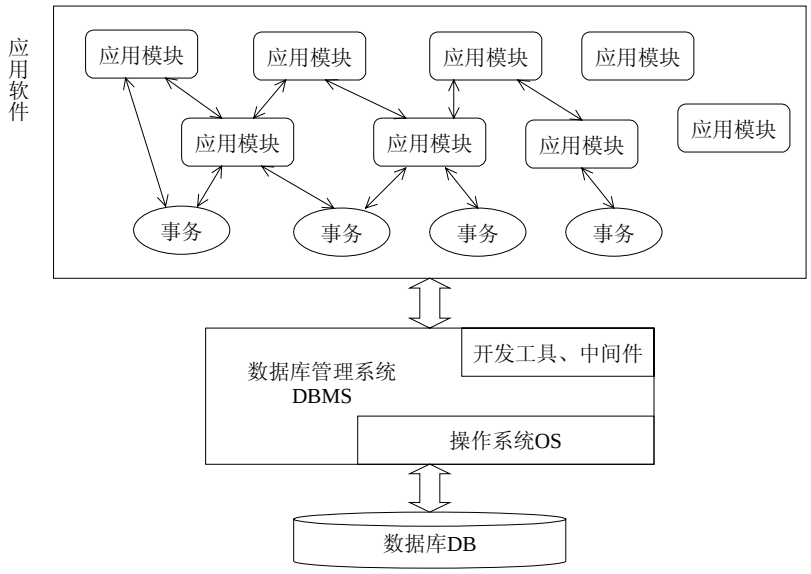

图 2.5　数据库应用系统DBAS软件组成

DBAS设计开发包括硬件和软件2个方面。硬件方面主要涉及到根据系统的功能、性能、存储等需求选择和配置合适的计算机硬件平台（如CPU、内存、存储设备、网络设备等），并与开发好的DBAS软件系统进行集成，组成完整的数据库应用系统。与软件方面相比，硬件设计开发的工作量要小得多。

开发一个实际DBAS时，数据库管理系统DBMS可直接采用商用产品，如关系数据库引擎SQL Server、DB2、Oracle、Sybase等，也可以选用开源DBMS。操作系统可采用MS-Windows、Unix和Linux等。这两类系统软件经过正确的配置、部署后，与系统内其它软件、硬件进行集成，组成完整系统。因此，DBAS的软件设计开发工作主要针对两类软件：数据库和数据库应用软件。

数据库设计的内容是根据数据库三级模式结构，设计应用数据在数据库中的组织和存储方式，即设计数据库模式。

数据库应用软件的设计开发则分为与数据访问有关的事务设计实现和与数据访问无关的软件模块设计开发。

设计开发数据库事务模块时，既要满足具体应用领域对数据库数据访问的要求（这些要求用数据库完整性约束来表示），也要符合数据库事务的 ACID 特征要求。从模块组成来讲，这些事务可以是一条或若干条数据库查询语言 SQL 语句，也可以是宿主语言（如 C 语言）语句与嵌入式 SQL 语句的混合体，或者是由宿主语言语句和数据库访问接口（如 ODBC、JDBC、ADO）语句组成。

事务具体实现方式与系统所采用的数据库管理系统 DBMS 平台、数据库访问方式和接口密切相关。关于数据库事务的相关内容，参见第 9 章。

与数据访问无关的软件模块则是在数据库事务提供的应用数据基础上，根据应用领域具体业务逻辑对这些数据作进一步处理，以实现复杂的数据处理功能；与数据访问无关的软件模块也可以实现与数据库中数据无任何关系的其它应用领域业务逻辑。

以2.8节描述的商场经营管理系统为例。假设该系统采用SQL Server2000数据库平台，用C语言和ADO数据库访问接口开发数据库应用程序。根据需求，该系统需要对商品零售经营管理领域的商品、顾客、员工等12个数据对象进行存储、组织和管理，并在此基础上实现顾客管理、商品入库管理、日常销售管理和查询统计四类功能。这四类管理功能的具

体业务处理流程称为业务（处理）逻辑，是数据库应用程序的设计依据。

假设商场新购进 100 件商品编号为 102.10 的商品（GoodsID=102.10）。根据商场进货的业务处理逻辑，本次采购涉及对 3 个数据对象“库存表（Table_Stock）”、“采购入库单据（Table_InStockBill）”和“采购入库单据明细（Table_InStockBillDetail）”中相关数据的访问和修改。主要流程为：访问“采购入库单据”和“采购入库单据明细”，获取有关本次采购的商品编号（GoodsID=102.10）、数量（quantity=100）和进价（Buyprice）等信息；访问数据库中商品库存表 Table_Stock，将库存表中编号 GoodsID=102.10 的新购进商品的库存量 stocks 增加 100。

可以设计一个“商品进货”事务模块来实现上述进货业务处理逻辑。该事务主要业务步骤为（1）读“采购入库单据”相关信息（2）“读采购入库单据明细”相关信息（3）更新“库存表”中商品库存量。前 2 个步骤可以直接用 2 条 SQL 查询语句 Select 实现；对“库存表”的更新利用 SQL 数据库更新语句 Update，将本商品原有库存量 stocks 增加 100，即 Table_Stock.stock:= Table_Stock.stock +100。对本事务的完整性约束为：库存表中 GoodsID=102.10 的商品的原有库存量加上进货量 100 必须等于该商品更新后新的库存量，可以表示为 stocks_old + quantity = stocks_new。本事务以 C 语言程序的方式实现，程序中包括了访问数据库的嵌入式 Select 和 Update 语句。程序中还包括一些 C 语言变量和语句，用于组合整个事务业务流程，并在事务内的三条 SQL 语句间传递相关参数，如将读“采购入库单据明细”语句 select 得到的商品进货量传递给“库存表”更新语句 Update。

如果商场经营管理层希望进一步了解不同类型的顾客购买商品的喜好、兴趣与消费能力之间关联模式的信息，由于本系统现有的 4 个功能无法满足这一需求，因此需要新开发一个经营分析子系统。该子系统利用从数据库中获得的有关顾客、商品、销售等数据，采用一定的数据挖掘算法，获取管理层希望的信息。该经营分析子系统包括 2 类软件模块：（1）数据库访问事务模块，其功能是从数据库中读取相关数据形成数据文件。（2）数据挖掘模块，其功能是从数据文件中挖掘模式信息。该模块属于与数据库访问无关的软件模块，本模块实现所依据的业务逻辑是具体的数据挖掘算法。

2.2.2 DBAS 生命周期模型

从软件设计与开发角度，数据库应用系统是一类典型的面向数据管理和数据处理的复杂软件系统。数据库应用系统的设计开发应当在满足实际应用需求的前提下，遵循数据库系统三级模式结构所规定的数据库设计范型，按照软件工程所定义的复杂软件系统开发原则，采取工程化的方法，按计划、分步骤地进行，以便保证系统开发质量，降低开发成本，加快开发进度。因此，数据库应用系统的设计开发必须有软件过程模型作为指导。

下面我们给出数据库应用系统生命周期模型，该模型定义了数据库应用系统设计、开发和运行维护整体框架，规定了设计、开发和运行维护各阶段的主要目标、工作内容和所采用的关键技术。

面向应用的 DBAS 的生命周期模型和设计过程如图 2.6 所示，其基本思想如下：

（1）参照软件开发瀑布模型的原理，DBAS 的生命周期由项目规划、需求分析、系统设计、实现与部署、运行管理与维护等 5 个基本活动组成。

（2）将快速原型模型和增量模型的开发思路引入 DBAS 生命周期模型，允许渐进、迭代式地开发 DBAS。在一次迭代开发过程中，通过项目规划、需求分析、系统设计、原型构建等基本开发活动，开发出一个满足用户部分需求的原型系统。然后从该原型出发，在下一次迭代开发过程中构造一个功能更为完善的 DBAS 原型。通过多次迭代，逐步扩展各个原型系统的功能，使之最终满足全部用户需求，形成最终 DBAS 产品。

例如，2.8 节所描述的商场经营管理系统可以采用这种增量、迭代开发方式。首先需

要设计实现数据库，得到一个实现了组织和存储应用领域数据功能的原型系统。在此原型系统基础上，通过后续 4 次迭代过程，逐步在原型系统中增加顾客管理、商品入库管理、日常销售管理和查询统计功能，最终得到一个满足全部用户需求的完整系统。

（3）根据 DBAS 的软件组成和各自功能，细化 DBAS 需求分析和设计阶段，引入了数据组织与存储设计、数据访问与处理设计、应用设计三条设计主线，分别用于设计 DBAS 中的数据库、数据库事务和应用程序。其中，数据库事务设计和应用软件设计属于数据库应用系统行为设计范畴。

（4）根据数据库系统三级模式结构，将 DBAS 设计阶段细分为概念设计、逻辑设计、物理设计三个步骤，每一步的设计内容又涵盖了三条设计主线。

在许多数据库应用系统中，对数据库数据的访问处理只是全部系统功能的一部分，还有其它一些系统功能与数据库并无直接关系，或者数据管理只是这些功能中的一部分。对这类系统功能的设计需要按照应用设计主线，根据具体应用领域的业务处理逻辑来进行。

应用设计主线将数据库应用程序的设计活动细化为应用需求分析、总体设计、程序概要设计、程序详细设计等步骤。整个应用系统/应用程序的设计活动可以与数据库设计、数据库事务设计并行进行。应用设计主线的具体设计目标、内容根据具体应用需求而定。

应用设计主线与数据访问与处理设计主线间有着一定的联系。事务概要设计和详细设计的结果可用于应用程序中与 DB 访问有关的功能模块的概要和详细设计。

（注：图 2.6 有所改动!!!）

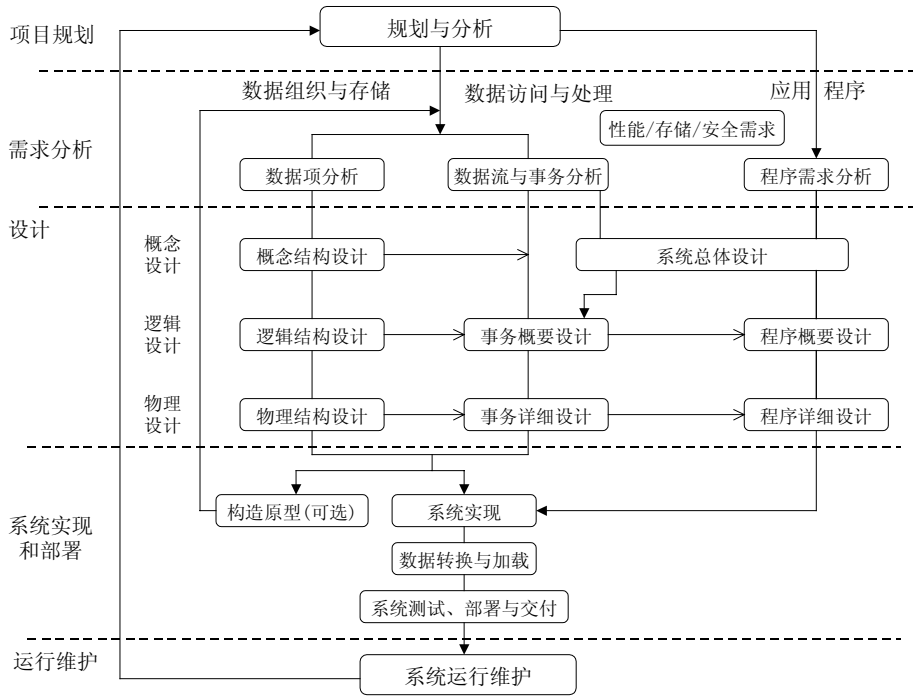

图 2.6 数据库应用系统生命周期模型

以电信网络管理系统设计开发为例。电信网络管理系统是一类面向电信网络管理领域的数据库应用系统。根据电信网络管理要求，该系统应当实现对电信网络（如 GSM/GPRS 移动通信网络，称为被管对象 MO）的配置、性能、故障、安全和计费管理。图 2.7 给出了电信网络管理系统的软件组成结构示意图。

该系统利用数据库存储电信网络的配置数据、性能数据、故障数据、安全数据、计费数据等应用领域数据，因此数据组织与存储设计主线的设计内容是为这些应用数据设计合理的

数据库组织存储模式，也就是设计网管数据库。设计数据库时，首先需要根据电信网络管理 TMN 理论中的信息建模方法，将网络中的被管网元抽象为被管对象 MO。对每个 MO，分别从配置、性能、故障、安全和计费等方面用相关属性进行描述，组成管理信息库 MIB；然后，将 MIB 作为数据对象，为其设计数据库三级模式结构。

存储在网管数据库中的各类网管应用数据由数据库事务通过 ODBC 数据库访问接口进行查询、插入、删除、修改等操作，数据访问与处理主线的设计任务包括设计网管数据访问事务和设计数据库接口。

在网管数据基础上，配置管理、性能管理、故障管理、安全管理和计费管理等应用模块根据电信网络管理 TMN 理论的原理和方法，依据网管业务处理逻辑，实现具体的网络管理功能，并通过人机交互和接口模块与网管人员或其它网管系统进行通信。因此，应用设计主线的设计工作是根据电信网络管理 TMN 理论和具体网管需求，设计配置管理、性能管理、故障管理、安全管理、计费管理、人机界面和对外接口等应用模块。

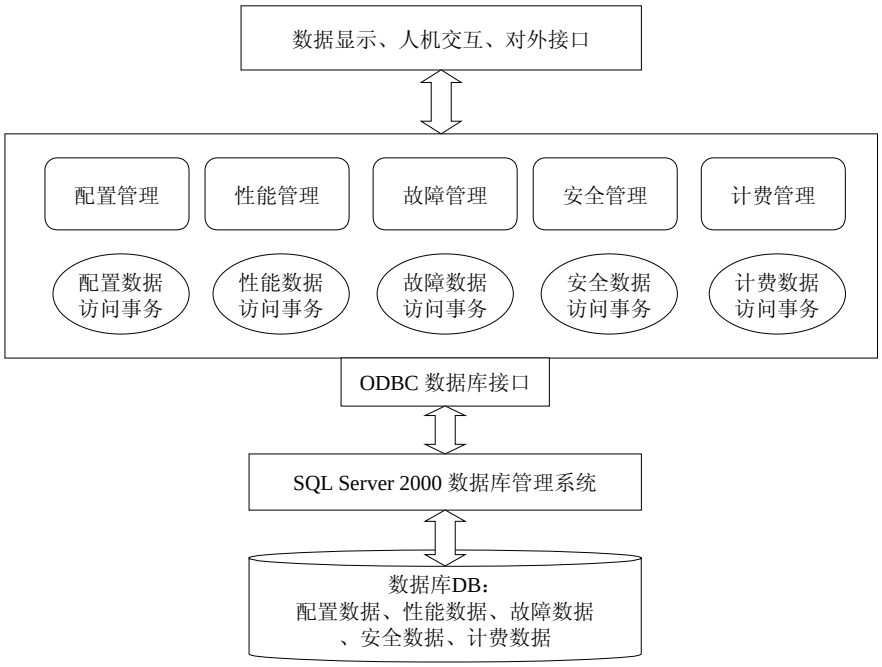

图 2.7 电信网管系统软件组成

2.3 规划与分析

数据库应用系统开发与其它复杂软件系统开发一样，需要人力、物力和时间的投入，需要相应的技术和工具。为使 DBAS 的开发能够按照预定成本、进度、质量要求正常进行，应当通过项目管理对开发成本、进度、质量、人员、风险等进行有效控制。项目管理贯穿在整个 DBAS 生命周期中。

作为 DBAS 生命周期中的第一步，规划与分析的目标是将数据库应用系统作为由计算机硬件、计算机软件和用户组成的复杂人机系统，从项目管理的角度，面向实际应用和用户需求，确定整个数据库应用系统的目标和任务，从技术、操作和经济三方面进行可行性分析，并制定合理的项目开发计划。

规划与分析是数据库应用系统生命周期的起点，规划与分析的对象既包括作为产品的DBAS本身，也包括开发这一产品的项目。图2.8给出了规划与分析的主要工作内容。

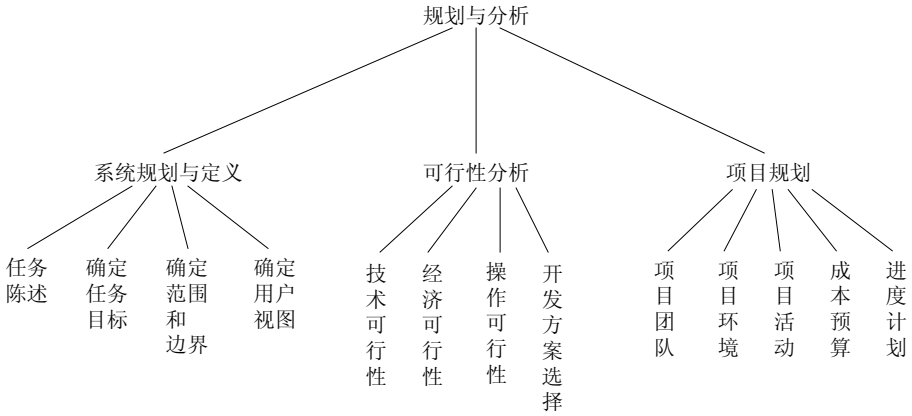

图 2.8 规划与分析

2.3.1 系统规划与定义

系统规划与定义是面向将要开发的数据库应用系统，通过了解用户实际需求，明确该系统需要实现的目标和任务，并从数据管理和数据处理的角度，确定系统中数据库软件的功能、性能范围。

系统规划与定义的具体内容包括任务陈述、确定任务目标、确定范围和边界、确定主要用户视图。

（1）任务陈述。

描述所要开发的DSAS的总体目标。

（2）确定任务目标。

明确为了实现任务陈述所规定的系统总体目标，DBAS应该支持的一系列数据管理和数据处理任务和活动。

（3）确定系统范围和边界。

对于任务目标所规定的各项任务与活动，明确哪些是由DBAS中的软件子系统、硬件子系统完成的，哪些是由DBAS用户或其它（信息）系统完成的；

进一步地，对于分派给软件子系统执行的任务和活动，明确如何在DB、DBMS、应用程序、操作系统等系统软件之间进行任务划分和分布，明确定义这几类软件的各自范围。这里，软件范围是指一个软件应该实现的功能、性能边界。

系统范围和边界定义了DBAS做什么、不做什么、做到什么程度，是DBAS需求分析和系统设计等后续开发步骤的设计依据。

（4）确定用户视图。

应用领域数据存储在数据库中，合法的DBAS用户可以根据自己的需要访问处理这些数据。根据实际应用领域情况将用户进行分类，明确每类用户需要访问的数据库中哪些数据、如何使用这些数据，组成用户所对应的用户视图。这些用户视图表示了不同DBAS用户的数据访问/处理需求。

用户视图所定义的数据是数据库中全部数据的一个子集。不同用户的用户视图大小不同，并且可以有重叠。如图2.9所示。

例如，对商业企业商品零售管理系统而言，该系统的不同用户，如经理、信息主管、售货员、顾客等，对各种商品销售信息的不同数据访问和处理需求如下：

（1） 经理：察看月/季度销售统计数据、成本和利润数据。

（2） 售货员：维护（录入、更新、删除）单件商品销售信息。
（3） 信息主管：进行商品销售信息的统计分析、提供各种统计报表。
（4） 顾客：察看商品品种、商品数量、价格信息。

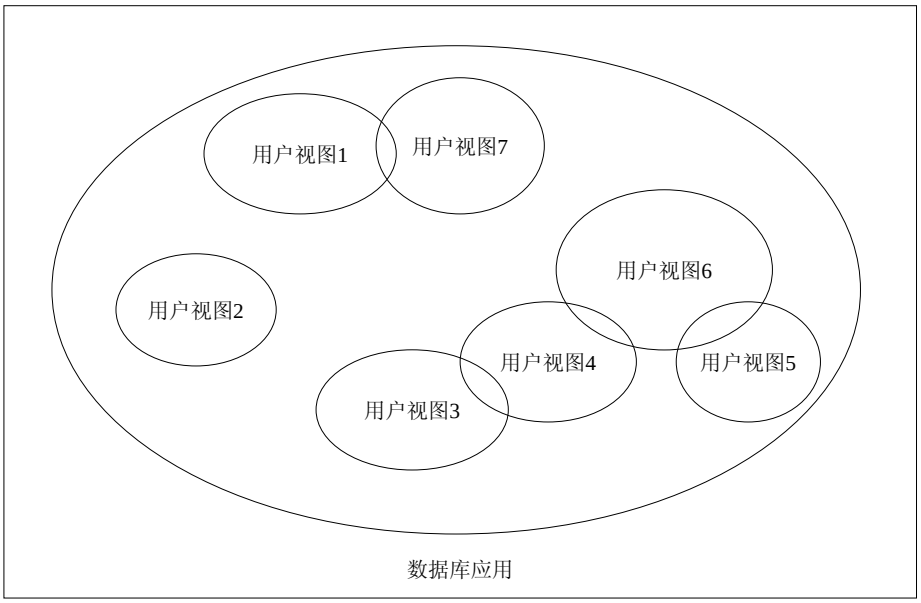

图 2.9 DBAS 中的多用户视图

2.3.2 可行性分析

数据库应用系统开发受到时间、资源、技术等因素的限制。在明确了 DBAS 的任务目标和系统范围之后，需要从技术、经济、操作等方面进行项目可行性分析，评估判断 DBAS 开发项目在现有技术和经济条件下是否可以执行、是否能够达到预期目标，以及为保证项目正常进行需要的各种资源和支撑条件。可行性分析包括经济可行性、技术可行性、操作可行性研究和开发方案选择等四方面。

（1）经济可行性。

经济可行性研究的目标是对项目进行成本效益分析，估算项目开发成本，评估项目经济效益、社会效益和利润，在项目成本和收益间进行合理权衡，并分析项目对其产品或利润的影响。

DBAS 成本包括：（a）系统软硬件购置费用，如 DBMS、计算机、存储设备、网络设备的购置费用。（b）系统开发费用，如人工费用、材料费用、培训费用等。（c）系统安装、运行、维护费用。

（2）技术可行性。

技术可行性研究的目标是根据用户提出的系统功能、性能及实现系统的各项约束条件，对系统软件、系统硬件、技术方案作出评估和选择建议。

硬件可行性研究是分析建议 DBAS 的硬件平台环境和设备，如数据库服务器和应用服务器的体系结构及处理能力、存储设备的容量和数据访问速度、网络环境等，并提出硬件选型建议。

软件可行性研究包括：对可用数据库管理软件 DBMS 和操作系统的选型评估和建议、对中间件和开发环境的选型建议、对数据库应用程序开发模式和编程语言的建议等。

技术方案选择是根据系统技术需求，提出 DBAS 可能采用的合理技术方案或关键技术。例如，考虑 DBAS 体系结构时，可以从第 1 章所讲述的集中式、分布式、客户/服务器、并

行式等几种方案中选择。

（3）操作可行性。

操作可行性研究的目标是论证是否具备 DBAS 开发所需的各类人员资源（项目管理人员、数据库系统分析员、应用编程人员等）、软件资源、硬件资源和工作环境等，以及为支持 DBAS 开发如何去改进加强这几方面资源。

（4）开发方案选择。

开发方案选择的目标是提出并评价实现系统的各种开发方案，从中选出一种用于 DBAS 软件开发。

例如，考虑 DBAS 软件开发模型时，可以选择 2.1 节中的瀑布模型、增量模型、螺旋模型，也可以采用图 2.7 所示的 DBAS 生命周期模型；再如，如果 DBAS 中的某个数据库应用模块的委托开发费用比自行开发费用更低，并且软件质量和开发进度也能得到保证，那么该模块可以采用委托开发的方案。

不同的开发方案对时间、成本、人员、技术、设备的要求都不尽相同，开发出来的 DBAS 在功能、性能、效益等方面也会有所差别。因此，需要利用折衷手段反复比较各种方案的成本-效益，选择一个最佳 DBAS 开发方案。

完成上述四方面的可行性分析工作后，应形成相应的数据库应用系统开发可行性研究报告，并提交给项目管理部门对可行性研究报告进行评审，作为下阶段项目立项和规划的重要依据。

一些软件工程教材和资料给出了软件可行性分析报告的模板。DBAS 可行性分析报告内容可在参照这些模板基础上，按照 DBAS 组成，对系统硬件、数据库管理系统 DBMS、操作系统、开发工具与中间件等分别从技术、经济、资源等方面分别做出评估和选择。

如果 DBAS 开发强调风险控制，可借鉴螺旋模型思想，在可行性分析中加入风险分析，判断在给定的技术、资源、时间等约束条件下，能否设计并实现 DBAS 所预期的功能和性能。同时采取相应的风险管理技术，对项目全过程进行识别、分析和监控，最大限度地降低系统开发风险。

2.3.3 项目规划

一个 DBAS 开发项目经过系统规划与定义、可行性分析 2 个步骤并确定立项后，可以开始项目规划步骤。项目规划是项目管理者对资源、成本和进度做出合理估算，在此基础上制定切实可行的 DBAS 项目开发计划。项目规划包括以下工作内容：

（1）确定项目的目标和范围，根据系统规划与定义的工作内容，具体说明项目的最终产品以及期望的时间、成本、质量目标。

（2）根据 DBAS 软件开发模型，分解和定义整个项目包括的工作活动和任务。

（3）估算完成该项目的规模及所需各种资源。

（4）制定合理的 DBAS 项目计划，包括进度、成本和质量等方面的预测和控制方案。

项目规划的结果应形成数据库应用系统项目计划文档，也就是项目计划书。计划书应包括上述四方面内容，计划书格式可参考相关软件工程文档规范或标准，如 IEEE 1058-1998 标准。

2.4 需求分析

数据库应用系统需求是指用户对 DBAS 在功能、性能、行为、设计约束等方面的期望和要求，也就是希望 DBAS 做什么、做到什么程度等具体要求。DBAS 需求分析是在已经

明确的 DBAS 系统范围基础上，通过对应用问题的理解和分析，采用合适的工具和符号，系统描述 DBAS 的行为特征和约束，并形成需求规范说明文档。该文档称为 DBAS 需求分析规范说明书，是 DBAS 设计和测试阶段的重要依据。需求分析过程由需求获取、需求分析、需求描述与规范说明、需求验证等步骤组成。

根据生命周期中的三条设计主线，DBAS 需求分析主要包括以下内容

（1）数据需求分析。分析用户需要从数据库中获取的信息内容和性质，也就是在数据库中需要存储的应用领域数据。

（2）数据处理需求分析。分析从用户角度，对数据需要做什么样的加工处理。

（3）业务需求分析。从应用设计角度，分析 DBAS 应具有的业务处理逻辑。

（4）分析数据库系统在性能、存储、安全、备份与恢复等方面的要求。

2.4.1 数据与数据处理需求分析

数据需求分析是从数据组织与存储设计角度，辨识应用领域所管理的各类数据项（data items）和数据结构，与数据处理需求分析结果一起，组成数据字典，形成“数据规范说明书”。

数据字典包括数据项、数据结构、数据流、数据存储和处理过程 5 个部分。数据项是数据的最小组成单位，若干个数据项可以组成一个数据结构，数据字典通过对数据项和数据结构的定义来描述数据流、数据存储的逻辑内容。

对数据项的描述包括项名、含义、别名、类型、长度、取值范围、取值含义、与其它项逻辑关系等。数据结构描述包括数据结构名、含义说明、组成（数据项或数据结构）。

数据需求分析是与数据处理需求分析结合在一起的。数据处理需求分析从数据访问和处理的角度，明确对各类数据项所需进行的数据访问操作，分析结果可表示为（1）数据流图（Data Flow Diagram, DFD），或（2）DBAS 应支持的各种数据处理事务规范。

数据流图是一种形式化的数据需求分析技术，利用数据项、数据存储、数据加工和数据流等概念描述对数据的处理。如果采用数据流图表示法，数据需求分析实质是针对数据字典中的数据项，分析各种施加于数据项的数据访问和处理操作，并表示为数据字典中的数据流、数据存储和处理过程。关于基于 DFD 需求分析方法的详细内容可参见第 3 章。

数据处理需求分析结果也可以表示为事务规范。事务规范包括以下几方面事务描述信息（1）事务名称（2）事务描述，指对事务功能、性能、完整性约束等方面的描述（3）事务所访问的数据项（4）事务用户，指启动该事务执行的事件/用户。图 2.10 给出了一个电信网络管理系统中的故障管理领域“基于事件触发的 MO 故障报警事务”规范实例。

1. 事务名称 基于事件触发的 MO 故障报警事务

2. 事务描述：

（1）设置发生故障 MO 的故障属性包中的故障状态位，记录本次故障的故障发生时间、故障类型、故障级别、故障原因等参数。

（2）通过人机界面，向用户发出报警。

（3）根据故障类型，将故障报警次数加 1。

3. 数据项：

被管对象 MO 故障属性包中的故障状态位、故障发生时间、故障类型、故障级别、故障原因

4. 事务用户：

（1）网管系统对重要网络设备 MO 周期性进行故障检测，当发现某个 MO 发生故障时，系统将执行本事务。

（2）如果被管网络设备 MO 自身具有故障检测功能（如自身带有告警监测板），故障发生时可主动向网管系统报告，从而触发本事务的执行。

图 2.10 基于事件触发的 MO 故障报警事务规范

在系统规划与分析阶段，DBAS 开发者已经明确了各类用户的用户视图。因此，可以从这些用户视图出发，针对每个用户视图进行数据及数据处理需求分析，然后汇总各个视图的分析结果得到对系统的完整分析结果。这种方法可以看作是一种多视点、多角度数据及数据处理分析技术。

在某些特殊应用领域开发数据库应用系统时，开发方法和步骤既要遵循图 2.6 中的 DBAS 生命周期模型，也要遵守这些应用领域提出的特殊开发准则，采用一些面向领域的特殊开发技术。

例如，当在计算机网络管理和电信网络管理领域开发一个网络管理数据库应用系统时，在需求分析阶段可以依据计算机网络管理 SNMP 或电信网络管理 TMN 的原理，采用信息建模方法，识别网络内各种被管网元，将其抽象为被管对象 MO (Managed Objects)。对每个 MO，分别从配置、性能、故障、安全等方面用相关属性包进行描述，构造管理信息模型 MIB。MIB 就是需求分析阶段的数据项分析结果，可以用于后续设计阶段中的概念数据库设计和逻辑数据库设计。

大型复杂数据库应用系统涉及的数据项非常多，数据处理功能也非常复杂，导致数据字典规模非常庞大。因此，必须采用相应的 CASE 开发工具。目前，大多数支持数据流分析的 CASE 工具都具备数据字典管理功能。这些功能包括：

（1）一致性检查。当分析人员要求创建新的数据条目并键入名称或别名后，CASE 工具自动进行重名建查，以避免数据流图中不一致的数据定义。

（2）根据已有数据流图生成相关转换的列表。并且随着数据流图的进化，CASE 工具可自功修改该列表，以使数据字典和数据流图保持一致。

（3）自动完成有关数据条目的各种查询。例如：该数据条目在何处使用？修改某一部分数据流图将会影响哪些数据条目？对这些问题的正确同答有助于分析人员在需求模型进化过程中维护模型一致性。

2.4.2 业务处理逻辑需求分析

一个大型复杂数据库应用系统既包括与数据存储、管理和处理有关的功能，也包括许多与数据库和数据访问无直接关系的其它功能，系统中一些软件模块与数据库访问无直接关系（参见图 2.5 和图 2.7）。因此，光有数据需求分析和数据处理需求分析是不够的。

应用领域业务（处理）逻辑需求分析是从 DBAS 高层目标和整体功能出发，分析系统或系统中一些大粒度子系统应具有的业务类型和功能，明确用户或外部系统与 DBAS 的交互模式。业务处理逻辑需求分析主要涉及系统的外部行为，也包括某些系统内部关键特性，如系统某些关键技术的选择、定位或原理。分析对象既可以是与数据管理有关的业务逻辑，也可以是与数据库完全无关的系统业务（如系统采用的人机交互模式）。业务逻辑需求代表了应用程序的功能、性能需求，为后续系统设计阶段的应用程序总体设计提供了重要依据。

数据和数据处理需求分析是面向用户的，业务逻辑需求分析则主要面向系统开发者。分析结果可采用多种方式表述。可以用自然语言来描述系统业务逻辑、流程、算法原理等；也可以采用 Petri 网、自动机等形式化方法；或者采用某种特定的领域相关的形式化、半形式化描述机制。

例如，一个电信网络管理系统的网管**业务可以**分为为基于数据库的网管数据组织、配置管理、性能管理、故障管理、安全管理和计费管理等几大类，可以用国际电联 ITU 推荐的形式化描述工具 SDL 语言（Specification and Definition Language，基于扩展有限状态机模型）和 MSC 图来刻画系统中这几类业务的功能、总体流程和对外接口。

2.4.3 性能需求分析

功能需求描述了一个系统应当做什么，性能需求描述了系统应当做到什么程度。DBAS需求分析阶段的另外一项重要工作是分析 DBAS 应具有的性能指标，主要包括：

（1）数据库查询响应时间，或数据访问响应时间，指用户向数据库系统提交数据访问请求到查询结果返回用户的时间。

（2）系统吞吐量，指系统在单位时间内可以完成的数据库事务或数据查询的数量。

根据事务处理性能委员会（Transaction Processing Performance Council, TPC）定义的数据库系统基准程序标准，系统吞吐量可表示为每秒事务数 TPS。

（3）允许的并发访问最大用户数量，指在保证单个用户查询响应时间的前提下，系统最多允许多少用户访问数据库。

（4）每 TPS 代价值，Price per TPS，用于衡量系统性价比的指标。对商业数据库应用系统，既要有好的系统性能，又要有合理的使用成本。

以 2.8 节描述的商场经营管理系统为例，规定其性能指标为：最多允许 30 个用户同时并发访问数据库，并且当用户向系统提交查询请求后，系统应在 5s 内给出查询结果

DBAS 的性能指标是系统软硬件设计开发的重要依据。影响 DBAS 性能的主要因素有：

（1）系统硬件资源，如 CPU 数量与速度、I/O 系统容量与速度、内存大小、内存缓冲区大小等。

（2）网络通信设备性能，如网络带宽、连接速度、数据传输速度等。

（3）操作系统环境，如对并发进程/线程的支持程度、文件子系统和 I/O 子系统的性能等。

（4）数据库的逻辑设计和物理设计质量，如关系数据库中好的关系模式结构、合理的数据库存取组织和存取路径；数据库配置参数，如数据库存储空间大小等。

（5）DBMS 的配置和性能，如 DBMS 采用的查询优化策略、索引优化策略、数据库管理配置参数（内存配置选项、I/O 配置选项、数据库缓冲区配置选项）

（6）数据库应用程序自身，如程序中的事务机制、并发控制机制的使用方法、故障恢复机制的使用、数据访问模式、存储过程的使用、程序质量等。

2.4.4 其它需求分析

除了需要对数据库应用系统的从功能和性能两方面进行需求分析外，还需要考虑 DBAS 的其它需求，包括存储需求、安全性需求、备份和恢复需求。

1. 存储需求分析

存储需求分析是指估计 DBAS 系统需要的数据存储量，以 DB 所存储的数据总量（byte）来度量。包括：

（1）初始数据库大小，指 DBAS 刚投入运行时的数据存储总量。

（2）数据库增长速度，指运行过程中，DB 内数据量的变化情况（增加、减小）。

数据存储总量的估算可采用如下方法：根据数据字典中每个数据项的结构描述信息，估计每个数据项的体积（以 byte 为单位），将所有数据项的体积累加。

2. 安全性需求分析

安全性分析考虑以下几个问题：

（1）DBAS 系统应达到的安全控制级别。

根据 DBAS 应用场所的实际数据安全性要求，按照可信（trusted）计算机系统评测标准，确定 DBAS 应达到的安全控制级别。

根据《中华人民共和国计算机信息系统安全保护条例》和美国 NCSC 颁布的《可信计算机系统评估标准关于可信数据库系统的解释》，安全控制级别分为 A、B、C、D 四级，

从 D 到 A 级，安全性要求和安全控制手段逐步增强。如果 DBAS 用于数据安全性要求不高的一般场合，可将级别定位在 C 级，如 C2 级。如果应用于军队、政府部门等高保密场合，可将级别定位在 B 级，如 B1、B2 级。

DBAS 的整体安全级别取决于 DBMS、操作系统和其它系统软件的安全级别。这些软件的安全级别必须相互配套，才能保证整个 DBAS 的安全级别。

（2）各类用户的数据视图和视图访问权限。

在实际应用中，DBAS 的不同用户对数据的访问需求是不一样的。可以根据系统数据安全性需求，规定各类用户允许访问的数据库数据（即用户的数据视图）、以及对这些数据的访问权限（如查询、插入、删除、修改等）。

（3）DBAS 应有的口令保护机制。用于控制用户登录系统。

3. 备份和恢复需求分析

这是从数据库系统可靠性和系统故障恢复角度提出的系统需求，包括：

（1）DBAS 运行过程中数据库备份的时间点和备份周期。

（2）所需备份的数据，是全部数据库数据（包括应用数据、索引、日志等），还是其中一部分。

（3）备份方式是采用增量备份、还是海量备份。

以 2.8 节描述的商场经营管理系统，可以规定该系统以 30 天为周期，在周日晚上 24 点，采用增量备份方式，对销售单据、销售明细单据、采购入库单据、采购入库单据明细和库存表进行备份；其它数据以 60 天为周期，在周日晚上 24 点，进行增量备份

2.4.5 需求获取技术

参与需求分析的人员主要是系统分析人员和数据库用户。用户根据自己的需要提出需求，分析人员需要通过与用户的充分交流，调查清楚用户实际要求，与用户达成共识，获取用户需求，在此基础上才能进行需求分析。

需求获取的主要工作内容包括：

（1）调查用户需求。开发人员与各种层次的用户进行充分地交流和沟通，采取开调查会、跟班作业观察用户工作流程、查阅文献、填写调查表和交流询问等方式，尽量清楚地理解用户的问题和要求。

（2）分析和整理所获取的信息。开发人员对用户提供的各种问题和要求进行归纳整理，借助一些工具和方法，从用户一般性的陈述中提取用户的真正需求，并由此确定软件的功能、性能、接口关系、约束条件等。

（3）形成文档化的描述，并获得用户和开发人员的一致理解和认同。

2.5 系统设计

根据 DBAS 生命周期模型，数据库应用系统设计包括 DBAS 概念设计、 DBAS 逻辑设计、DBAS 物理设计三个顺序设计步骤/活动，每个步骤中的设计活动按照数据组织与存储设计、数据访问与处理设计、应用设计三条主线进行，设计的 DBAS 应具有较好的逻辑独立性和物理独立性。

2.5.1 概念设计

DBAS 概念设计包括数据库概念结构设计和 DBAS 系统总体设计。

1．概念结构设计

概念结构设计也称为概念数据库（Conceptual database design）设计，是依据数据需求分析阶段得到的需求结果（一般表示为数据字典和数据流图），分析辨识需要组织存储在数据库中的各类应用领域数据对象的特征及其相互间关联关系，并表示为外模式。概念数据库独立于数据库逻辑结构，也独立于具体的 DBMS。

设计概念结构通常采用四种策略：

（1）自顶向下。首先定义全局概念结构的框架，然后逐步细化。

（2）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。

（3）逐步扩张。首先定义最重要的核心概念结构，然后以此为核心向外扩充，逐步生成其它概念结构，直至得到系统的总体概念结构。

（4）混合策略。自顶向下和自底向上相结合，首先确定全局框架，划分为若干个局部概念结构，独立设计每个局部概念结构，再采取自底向上策略合并各个局部概念结构，得到最终全局概念模型。

实际应用中，可以根据具体应用特点，采用不同的策略。例如，对于组织机构，因其固有的层次结构，可以采用自顶向下的设计方法；对于已实现计算机管理的业务，可以以此为核心，采用逐步扩张策略；而自底向上是实际应用中最常采用的策略。

E-R 方法利用 E-R 图表示数据库的概念模式，是最著名最常用的概念结构设计方法。自底向上的 E-R 方法具体步骤如下：

（1）选择局部应用。

需求分析阶段会得到大量数据，这些数据分散杂乱，许多数据会应用于不同的处理，数据与数据之间的关联关系也较为复杂。为了确定需要建模的实体、属性和联系，可以从数据流图出发，选择适当层次的数据流图，让这一层的每一部分对应一个局部应用，实现某一项功能。从这一层入手，开始局部 E-R 图设计。也可以根据规划与分析阶段得到的用户视图，为各个用户设计局部 E-R 图。

（2）分别设计局部 E-R 图。

划分好各个局部应用之后，对于每一局部应用（其所用到的数据已收集在数据字典中），依照该局部应用的数据流图，从数据字典中提取出数据，使用抽象机制，确定局部应用中的实体、实体的属性、实体标识符和实体间的联系及其类型。

事实上，在形成数据字典的过程中，数据结构、数据流和数据存储都是根据现实事物来确定的，都已经基本上对应了实体及其属性。以此为基础，加以适当调整，增加联系及其类型，就可以设计局部 E-R 图。

（3）局部 E-R 图合并。

根据局部应用设计好各局部 E-R 图之后，通过合并各局部 E-R 图可以得到整个系统的全局 E-R 图，完成概念数据库设计。

合并过程中需要解决局部 E-R 图间存在的冲突，消除局部 E-R 图之间存在的信息冗余，使之成为能够被整个系统所有用户都理解和接受的统一、精炼的全局概念模型。局部 E-R 图之间的冲突有属性冲突、命名冲突、结构冲突等。

合并 E-R 图是将具有相同实体的两个或多个 E-R 图合而为一，在合成后的 E-R 图中把相同实体用一个实体表示。合成后的实体的属性是所有分 E-R 图中该实体的属性的并集，并以此实体为中心，加入其它局部 E-R 图。再将合成后的 E-R 图作为局部 E-R 图，合并剩余的局部 E-R 图，从而最终构成一张全局 E-R 图。

局部 E-R 图合并过程中需要对其进行优化，包括实体类型的合并、冗余属性的消除和冗余联系的消除等。

2. 系统总体设计

一个大型数据库应用系统是由硬件和软件组成的复杂系统，在设计上应依据自上而下、由简到繁、逐步求精的原则。DBAS 软件包括操作系统、数据库管理系统、开发环境和中间件、应用软件。根据软件粒度，应用软件分为系统、子系统、功能模块、事务四个级别。

作为 DBAS 应用设计主线的第一步，系统总体设计从系统规划与分析结果（特别是其中的技术可行性分析内容）和系统需求分析内容出发，确定系统总体框架，作为系统后续设计活动的基础。系统总体设计的内容有：

（1）确定 DBAS 体系结构。将系统从功能、层次或结构角度划分为多个子系统，定义各子系统应实现的功能，明确各子系统间的控制和交互关系（接口关系）。

规划与分析阶段得到的任务陈述和任务目标是系统划分的重要依据。

（2）系统硬件平台和操作系统、数据库管理系统 DBMS 等系统软件的选型和配置设计。

系统硬件平台、操作系统、数据库管理系统是影响 DBAS 功能和性能的重要因素，其选型和配置设计可以根据规划与分析阶段的技术可行性分析结果和需求分析阶段的系统性能需求来进行。

（3）应用软件总体设计。确定软件体系结构，将应用软件划分为一系列软件子系统，定义子系统间的信息交互方式。软件体系结构有分层结构、客户/服务器结构、MVC 结构等；进一步地，划分各软件子系统的模块结构。

（4）对需求分析阶段识别出的业务处理逻辑进行初步设计，细化业务逻辑流程，分析所处理的业务数据和处理方式，明确采用的关键技术和算法等。

（5）关键技术方案的选型和初步设计。

2.5.2 逻辑设计

数据库应用系统逻辑设计包括以下三方面内容。

1. 数据库逻辑结构设计。

数据库逻辑结构设计是指从数据库概念结构出发，设计数据库的逻辑结构或逻辑模式，也称为逻辑数据库设计。

逻辑结构设计与 DBMS 采用的数据模型类型密切相关，如关系模型、层次模型、网状模型，但与具体的 DBMS 系统实现无关。由于在现阶段，E-R 图是数据库概念结构的主要表示方法，绝大多数 DBMS 都是基于关系数据模型的，因此数据库逻辑结构设计的主要内容是在 E-R 图的基础上设计数据库关系模式。逻辑数据库设计主要步骤为：

（1）E-R 图转换为初始关系模式。

（2）对初始关系模式进行优化，即根据关系规范化理论对模式进行规范化处理。一般要求静态关系为第一范式，动态关系至少或尽可能为第三范式。

（3）检查关系表对数据库事务的支持性，以判断关系模式的设计是否有遗漏。

（4）确定关系模式的完整性约束，包括健完整性约束、值域约束和空值约束、参照完整性约束和与应用相关的业务规则约束。

（5）从数据安全性和独立性出发，根据系统规划与分析阶段得到的用户视图，设计逻辑数据库中面向各类用户的基于关系模式的用户视图。

2. 应用程序概要设计。

将 DBAS 中的应用软件模块，按照逐步求精、信息隐藏和功能细化原则，进一步划分为子模块，组成应用软件的系统-子系统-模块-子模块层次结构，其中，直接访问数据库的模块/子模块抽象为数据库事务；确定各模块的功能和输入输出数据，设计模块使用的数据结构，定义模块交互的接口关系和交互流程。

3. 数据库事务概要设计。

事务概要设计的任务是面向具体应用领域，根据需求分析阶段识别出的各种 DBAS 事务，设计与具体 DBMS 和实现方法无关的事务数据处理流程，明确事务所访问的各关系表。事务中对数据库数据的查询、插入、删除、修改操作用与具体 DBMS 平台无关的两个元操作 read、 write 抽象表示。

数据库事务和应用程序概要设计具体内容参见第 7 章。

2.5.3 物理设计

数据库应用系统物理设计由以下设计活动组成。

1. 数据库物理结构设计。

数据库中的数据以数据文件形式存放在外存物理设备上，数据库物理结构主要指数据文件在外存上的存储结构和存取方法，它依赖于系统具体的硬件环境、操作系统和 DBMS。

在具体的硬件环境、操作系统和 DBMS 约束条件下，为数据库的逻辑结构（如关系数据库中的关系模式）设计符合应用要求的物理结构的过程，就是数据库物理结构设计。其目标是设计一个占用存储空间少、具有较高的数据访问效率和较低的维护代价的数据库物理模式。数据库物理结构设计包括：

（1）数据库逻辑模式转换，即将描述数据库逻辑结构的关系模式转换为目标 DBMS 平台可支持的基本关系表，设计关系表的完整性约束，定义相应的用户视图。

（2）设计数据存储结构，包括设计DB数据文件中记录的物理结构和记录间的相互组织关系。记录的物理结构是指纪录的组成、数据项类型和长度、关系表元组到物理记录的映射关系。记录的组织关系可以是顺序存储、哈希存储、堆存储和B^+树存储。

（3）设计数据存取方法或存取路径，主要是为 DB 文件设计索引字段和索引，以提高文件访问速度。

（4）数据分布设计。包括：确定数据库中应用数据、索引、日志、备份、聚集在系统内物理存储设备上的存放位置；派生属性数据分布和关系模式去规范化考虑。

（5）安全模式设计，包括：利用用户名/口令等机制设计 DBS 的系统安全方案；通过数据库系统视图机制和授权机制为用户分配对数据库对象的访问权限。

（6）确定系统配置。根据应用环境和上述物理设计结果，合理设置和调整 DBMS 和操作系统的存储分配参数，提供系统软硬件平台的初始配置信息。

（7）物理模式评估。对物理数据库设计结果从时间、空间、维护代价等方面进行评估，从多种可行方案中选择合理的数据库物理结构。

2. 数据库事务详细设计。

根据事务概要设计的得到的事务流程，利用 SQL 语句、数据库访问接口（如 JDBC、ODBC 或其它 DBMS 特定 API），采用高级程序设计语言或 DBMS 提供的事务实现机制，在具体 DBMS 平台和开发环境下，设计数据库事务。事务详细设计需要将事务概要设计中的 read 和 write 元操作替换为 DBMS 支持的查询、插入、删除、修改等具体数据库访问操作或数据库访问 API 调用。

3. 应用程序详细设计。

根据应用软件概要设计中定义的各模块功能和输入输出数据需求，结合具体的程序设计环境和机制，设计各模块的内部处理流程和算法、具体数据结构、对外详细接口等。

数据库物理结构设计的详细内容可参见第 6 章，事务和应用程序的详细设计在第 7 章有具体阐述。

2.5.4 数据库管理系统 DBMS 产品

数据库管理系统 DBMS 是数据库应用系统中最重要的数据管理软件，技术可行性研究的一项重要工作就是对 DBMS 的评估和选型。设计者应根据实际应用需要，综合考虑 DBMS 的功能、性能、价格等因素，为 DBAS 选择合适的 DBMS。DBAS 物理设计也与目标 DBMS 平台密切相关

DBMS 有开源产品和商用产品之分，前者包括 MySQL、PostgreSQL、Firebird、SQLite、MaxDB、Berkeley DB 和 Derby，后者如 IBM DB2、Oracle、Sybase、SQL Server|、Kingbase。

商用 DBMS 产品一般面向大中规模数据管理应用领域，数据管理功能完备、性能优越。并且许多商用 DBMS 提供了好的应用集成开发环境，为 DBAS 开发提供了技术和工具支持，例如 Sybase 公司的数据库开发工具 PowerBuilder 和数据库建模工具 PowerDesign、IBM 公司的可用于数据库设计和建模的软件集成开发环境 Rational。

商用 DBMS 的缺点在于：在某些具体应用领域，并非所有产品功能都能被用户用到，使用中存在功能浪费现象，导致产品性价比不佳；产品售价和技术服务费用相对昂贵。

开源 DBMS 产品的特点在于轻便、易用、易管理、数据库读取操作快捷，可以以较低价格甚至免费得到。开源 DBMS 一般用于数据存储量不超过 200G 的中小型数据应用领域。

开源 DBMS 虽然源代码开放，但并不意味着使用完全免费。这里需要注意开源产品所遵守的相关开放协议，例如 BSD 协议、GPL 协议、Apache 协议等。但不可否认的是，即使是收费，开源产品的花费仍然极低。例如，MySQL 采用双重授权策略，对于不愿公开自己源代码的使用者，需要付费使用 MySQL，而对于开放源代码的使用者，可以基于 GNU 的公共许可协议 GPL 来使用。许多大企业目前也开始使用开源数据库，这些大用户通过交纳服务费来获取产品全面技术支持，这在一定程度上改变了软件商业模式。

与商用DBMS相比，开源DBMS的不足之处在于功能相对简单，性能上也存在一定不足。例如，号称全球最受欢迎的开源数据库MySQL的目前版本就不支持子选择、事务处理、引用完整性、触发器、存储过程以及视图等标准数据库管理功能

高端客户开发大型复杂数据库应用系统时可以考虑采用商用化 DBMS 平台。对于中小规模的数据库应用系统，如果比较在意开发和使用费用，并且不需要复杂的数据管理功能的话，可以本着“够用即是好的”原则，选用合适的开源 DBMS 产品。

2.6 实现与部署

数据库应用系统开发人员根据 DBAS 设计结果，建立数据库，编写应用程序，集成 DBAS 软硬件，组成完整的 DBAS。系统经测试和试运行，验证在功能、性能等方面达到设计指标后，可以交付客户运行。这个过程称为数据库应用系统的系统实现与部署。

DBAS 实现与部署包括以下一些工作内容。

1. 建立数据库结构

建立数据库结构是指根据数据库逻辑结构和物理结构设计结果，采用系统所选用的 DBMS 提供的数据定义语言 DDL 编写建库脚本，经运行调试后，创建数据库中各类数据对象的结构，包括数据库、基本表、视图、索引和约束等。

建库过程中所创建的数据库对象的逻辑结构和物理结构信息作为数据库元数据存放在 DBMS 的数据字典（又称为系统目录）中。

SQL 语言是关系数据库标准语言，提供了数据定义、数据操纵、数据控制等功能。利用 SQL 语句 Create Table/Create View/Create Index 可以在数据库中建立关系表的逻辑结构和物

理结构。目前，不同的关系 DBMS 都不同程度地实现了对 SQL 标准的支持，但在语法格式和功能上又有各自特点，特别是一些商用关系 DBMS。因此，利用 SQL 语句建库时，必须参考具体的关系 RDBMS 的参考手册。

系统开发者除了利用 SQL 语句建立数据库外，还可以利用一些商用数据库管理系统提供的管理工具来图形化、交互式地创建数据库对象。例如，SQL Server 数据库系统提供了 2 类具有建库功能的系统管理工具：企业管理器和查询分析器。其中，查询分析器通过交互输入并执行 T-SQL 语句实现对系统的管理，可以利用 SQL Create 语句通过查询分析器创建数据库；企业管理器则提供了另一种交互式、图形化、非 SQL 语句方式的数据库创建机制，用户可以在管理器界面上直接输入关系表的名字、属性等结构信息，方便快捷地创建数据库。

2. 数据加载

数据加载是指筛选应用领域数据，将其转化为符合数据库逻辑结构和物理结构的格式，然后录入数据库中。

数据录入可采用以下几种方式：

（1）利用 SQL 语句 Insert 逐条录入，或编写含 Insert 语句的批处理程序批量录入数据。

（2）利用 DBMS 提供的工具将数据文件中的大量数据导入数据库中。例如，SQL Server 配备有“导入/导出数据”组件，提供了对多种数据格式数据的格式转换和导入/导出功能。

（3）如果需要从旧数据库中将数据转入新建立的数据库，可以编写数据库转换程序，实现数据库间的数据导入/导出。

数据加载可以是根据软件测试或系统试运行时的要求，加载测试数据或部分应用数据，以测试验证系统的功能和性能；也可以是在整个 DBAS 测试无误后，加载应用领域全部数据，为系统部署和正式交付运行做准备。

3. 事务和应用程序的编码实现及测试

在 DBAS 应用软件中，与数据库操作有关的事务、存储过程和触发器程序可以利用 SQL 语言来实现，也可以利用 DBMS 自带的一些管理工具。例如，可以用 SQL Server 数据库的企业管理器快速创建存储过程。

C、C++、Java 等高级程序设计语言可用于实现应用程序中的复杂数据处理功能和与数据库访问无关的业务逻辑。并且可通过在这些语言中嵌入 SQL 语句或数据库访问接口（如 ODBC、JDBC、ADO）实现对数据库的操作。

近年来相继出现了一些具有高级程序语言设计功能的数据库应用集成开发环境，如 PowerBuilder 和 Delphi 等，可以支持数据库应用系统的快速开发。

对开发出的软件模块和数据库事务，需要进行软件模块测试，判断这些模块是否在功能、性能方面达到设计要求。对在测试中发现的各种错误或不足，需要通过软件调试发现问题并进行改进。

软件模块的测试可以采用软件工程中常用的软件测试技术，设计各种不同测试用例和测试场景，通过黑盒测试检验各软件模块是否符合软件概要设计中对模块功能的要求，利用白盒测试检验软件模块的内部工作过程是否按照软件详细设计定义的模块工作流程正常进行。

4. 系统集成、测试与试运行

DBAS应用软件测试完成后，可以与数据库、数据库管理系统、操作系统和其它系统软件一起集成部署在DBAS硬件平台上，并对数据库管理系统和操作系统这两类系统软件进行正确的配置，组成完整的数据库应用系统DBAS。

下一步需要在试运行过程中对 DBAS 进行系统级联合测试。DBAS 加载实验数据后进入试运行阶段。本阶段通过模拟实际应用环境，执行测试场景，验证 DBAS 在功能上是否符合需求规范定义的业务逻辑要求、性能上是否满足查询响应时间和吞吐量等指标的要求，检查 DBAS 的存储需求和安全需求是否得到满足，分析系统是否达到系统预期设计目标。

测试由数据库设计人员、应用开发人员和用户一同进行。针对测试阶段出现的各种问题，需要及时从软硬件等方面改进完善系统。

5. 系统部署

在实际应用运行环境下部署开发完毕的 DBAS，加载应用领域真实数据；对 DBAS 用户进行培训，正式启动系统运行并交付用户使用。

2.7 运行管理与维护

数据库应用系统投入运行标志着系统开发任务的基本完成和系统运行维护工作的开始。DBAS 运行过程中应用环境可能发生改变，用户和系统需求也可能不断变化，同时数据库物理存储在运行过程中也会发生调整。因此为了保证数据库系统的正常运行，对 DBAS 的评估、调整、修改、改进等管理和维护工作贯穿在 DBAS 运行过程中，DBAS 的运行管理和维护是其生命周期过程的一个重要组成部分。

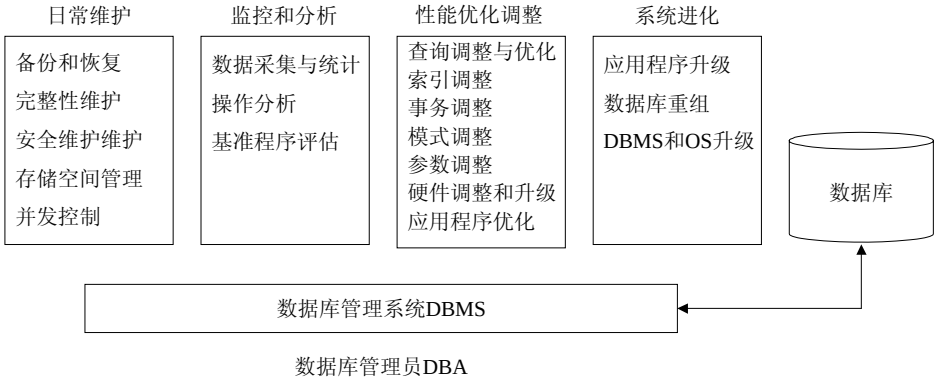

图 2.11DBAS 运行管理与维护

数据库运行管理与维护的主要工作内容包括日常维护、系统监控与分析、系统性能优化调整、系统进化。这些工作主要由数据库管理员 DBA 负责，也涉及其他设计开发人员。

2.7.1 日常维护

DBAS 日常维护的目标是保证在现有软硬件配置下，系统能够正常执行并有效处理各类故障，以满足系统预期的功能和性能要求。日常维护主要包括以下工作。

1. 数据库的备份和恢复

数据库的备份和恢复是系统正式运行后最重要的维护工作之一。DBA 要针对不同的应用要求制定不同的备份计划（如备份的时间点和备份周期、增量备份或海量备份），定期对数据库和日志文件进行备份，以便在发生系统故障和介质故障时，能利用数据库备份和日志备份文件，采取相应的恢复机制，尽快将数据库恢复到一致性状态，并尽可能减少故障对数据库数据的破坏。

2. 完整性维护

当应用环境发生变化时，数据库的完整性约束条件也会变化，此时需要 DBA 根据实际情况调整数据完整性约束，以满足用户要求。

例如，对于商品关系表 Table_Goods(GoodsID, GoodsClassID, GoodsName, ProductionDate, SaleUnitPrice, TotalStorage, GDescription)，如果其中的“商品描述信息 Gdescription”属性需要更多的描述信息，DBA 可以将该属性的数据类型由 varchar(80) 改为

varchar(120).

3. 安全性维护

DBA 应当根据用户实际需要授予用户不同的数据库访问权限。此外，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，比如有的数据原来是机密，现在可以公开查询了，或者是系统中用户的密级发生变化。这些都需要 DBA 根据实际情况修改原有的安全控制策略。

4. 存储空间管理

DBA 需要定期检查所有数据库文件的存储空间占用情况；对于快要用完分配空间的文件，需要扩展文件的磁盘空间；不断删除一些临时文件，回收磁盘空间，保证存储空间的可用性；如果一些授权用户需要建立一次性临时表，DBA 为其分配一定的存储空间，并根据使用情况调整或回收临时表所占空间。

5. 并发控制

DBA 在线监控数据库系统中各种锁的使用情况、以及可能出现的事务死锁情况。商用 DBMS 提供了多种检查数据库当前状态的管理工具。DBA 可以利用这些工具，查看各种锁的占有情况和等待该锁的事务；可以通过手动干预, 找出陷于死锁状况的事务，选择中止处于死锁状态的一个或多个事务，以支持其余事务的继续进行。

2.7.2 系统性能监控和分析

在数据库运行过程中，DBA 需要收集各种系统统计数据，监测系统各项性能指标是否达到要求。如果性能指标偏离预期值，需要通过一定的分析手段找出影响系统性能的各种因素。此外，通过系统告警监测还可以发现系统中一些潜在问题，有利于不断改进完善系统。

需要收集的统计数据涉及磁盘空间、表剩余空闲空间、内存使用和 I/O 性能等方面，这些统计数据反映了系统运行状况。统计数据可以通过 2 种途径收集得到：（1）由 DBMS 本身自动收集和存储统计数据。许多 DBMS 产品都提供了监测系统状态和性能参数的工具，利用这些工具可以方便地得到系统运行过程中一系列统计数据和性能参数值。（2）通过监控系统得到统计数据。

需要监测的系统性能指标包括用户查询响应时间、并发访问用户数、系统吞吐量等。

快照监控和事件监控是监控数据库系统性能和状态的两种常用方法。快照监控可提供数据库系统当前活动状态的快照/写真；通过事件监控，可捕获无法通过快照监控得到的短时间段内的系统信息。

DBA 还可以根据数据库系统日志文件，通过审计手段分析施加于数据库的各种操作，以发现系统变化和运行趋势及可能潜在问题，有利于系统维护。

监测系统性能指标的另外一种方法是利用事务处理委员会 TPC (Transaction Processing Council)定义的基准程序（Benchmark）来评估系统 TPS（transaction per second，事务/秒）等性能指标，并在此基础上考虑系统的性能/价格比（TPS/$）。

2.7.3 系统性能优化调整

如果通过系统监控分析，发现系统无法满足预期指标，需要根据影响性能的可能原因，通过性能调优来恢复系统的合理性能指标。系统性能监控和性能调优是 DBAS 运行维护的一项重要工作，系统性能调优的手段有数据查询调整与优化、索引调整、事务调整、数据库模式调整、DBMS 和操作系统参数调整、应用程序优化、硬件配置调整和升级等。

商用 DBMS 提供了功能丰富的数据库系统优化实用工具，用于系统监控和调优。例如，SQL Server 数据库系统提供了 SQL 事件探查器、系统监视器、企业管理器等工具用于监视引擎事件（如死锁）和与服务器进程相关的资源使用情况；DB2 数据库带有数据库系统监控器，可以采用快照监控和事件监控方法，监控 DB2 数据库使用情况。

如果一个查询发出过多磁盘 I/O 操作，或查询规划显示有很多相关索引未被使用，那么该查询效率不高，可考虑对其进行调整和优化；另外，如果某类频繁出现的用户查询涉及到某些视图，为提高查询速度，可考虑对这些视图使用物化（实体化）视图。但使用物化视图又有可能带来对视图的维护开销。因此，物化视图的使用需要在查询性能和维护代价间进行权衡折衷。

调整数据库索引是指利用 DBMS 提供的跟踪分析工具，确定事务执行所使用的各种索引，分析索引使用情况，根据统计数据，调整索引，提高索引使用效率。例如，若某类查询运行缓慢，可在此查询涉及的基本表中建立合适的索引；对于一些冗余、无用的索引可及时删除；当数据库访问模式发生变化时，还可以适当添加新索引。

模式调整涉及物理模式调整和逻辑模式调整。物理模式调整通过更改数据库文件组织形式和物理文件记录的分配来提高数据库的数据存储和存取效率。例如，SQL Server 2000 提供了索引优化功能，可在运行期间根据统计信息调整、优化、重建索引，以提高数据库文件存取效率。

逻辑模式调整可考虑以下几方面

（1）已达到第三范式的基本表，不要进一步规范化为 BCNF。数据结构规范化的级别越高，关系表分解得越多，检索一条信息的数据访问将不得不访问更多的表，执行多个连接操作，执行速度相对越慢。

（2）在分布式数据库环境下，如果通过数据查询模式，得知一个基本表的某些部分比其它部分访问更频繁，可以按水平分区或垂直分区方式适当拆分基本表并合理分布到不同物理存储位置，以加快数据访问速度。

（3）在某些情况下，对基本表进行去规范化处理。

关于数据库系统运行监控和性能调优的详细内容，可参见第 10 章。

2.7.4 系统升级

DBAS 在运行使用过程中，可能会根据用户提出的新需求从功能、性能等方面进行改进和扩充，对 DBAS 进行系统升级。

1. 改进应用程序

根据用户要求，对数据库应用程序从功能和性能等方面进行的扩充、完善，经过仔细的调试和测试，集成到原有系统中。DBA 需要建立和维护应用程序测试环境，管理测试过程，保证改进后的应用程序的正确性。

2. 数据库重组

数据库运行一段时间后，由于记录的不断增、删、改，会使数据库的物理存储变坏，从而降低数据库存储空间的利用率和数据的存取效率，使数据库的性能下降。这时 DBA 就要对数据库进行重组或部分重组（只重组频繁增、删的表）。

数据库重组不会改变原设计的数据逻辑结构和物理结构，只是按原设计要求重新安排存储位置，回收垃圾，减少指针链，提高系统性能。DBMS 一般都提供了供重组织数据库使用的实用程序，帮助 DBA 重新组织数据库。

当数据库应用环境发生变化，例如，增加新的应用或新的实体，取消某些已有应用，改变某些已有应用，这些都会导致实体及实体间的联系也发生相应的变化，使原有的数据库设计不能很好地满足新的需求，因此需要适当调整数据库的逻辑模式和物理模式，例如，增加新的数据项，改变数据项的类型，删除冗余列，改变数据库的容量，增加或删除索引，修改完整性约束条件等。DBMS 通过 DDL 提供了修改数据库结构的功能。

应保证数据库模式修改的正确性，例如不违反完整性约束和业务规则。对于重大的数据库模式修改，应先建立数据库测试环境，在此环境中对模式修改后的数据库样本进行完备的测试，验证模式修改的正确性。只有通过测试的模式修改，才允许应用到实际数据库系统。

3. DBMS 和 OS 版本升级

在 DBAS 运行过程中，它所依赖的数据库管理系统和操作系统平台有可能需要升级。此时需要 DBA 提供相应的技术支持，并对升级后的系统进行配置和测试。

2.8 应用案例

本节描述一个简化的商场经营管理系统的业务需求及其数据库设计要求，作为本教材后续各章节所讨论的各种 DBAS 设计与开发技术的应用案例。并且，按照 2.2.2 节讲述的原理和方法，对该系统进行系统分析及规划。

2.8.1 需求描述

商业零售企业需要对日常经营活动中涉及到的商品、顾客、员工等各类数据进行有效管理。随着商品零售业的蓬勃发展，商场规模的日益增大，其经营管理愈加复杂，各类业务数据量渐趋庞大，单纯的人工数据管理方式已无法胜任。

以计算机数据库技术为基础的商场日常经营管理系统，也即一种面向商业零售企业日常经营管理的数据库应用系统。

1. 功能需求

该系统应实现**顾客管理**、**商品入库管理**、**日常销售管理**和**查询统计**四类功能，整个系统工作于商场的局域网环境下。

该商场是支持会员制的商场，将顾客分为普通顾客和会员顾客两类。会员顾客持有会员卡，系统记录他们的详细信息。会员卡可以用来积分，会员顾客凭此积分参加日后本商场举行的各种优惠或兑奖活动。

商品采购完成以后，将进行商品上架工作，并通过商品入库管理系统记录相关进货信息，记录每次入库单信息，包括入库单号、日期和经手人，并保存商品名称、数量、进价、供应商等信息。此外还记录所有商品的现有库存信息。

商品上架以后，顾客在商场进行采购活动，采购交易最终通过销售人员在销售终端上完成，销售终端上运行的销售管理系统应能记录商场的每一次销售行为以及销售商品明细，同时进行会员顾客的会员卡积分和库存量减少操作。

查询统计模块可以根据各种属性进行顾客和会员卡的查询统计、入库单据和入库明细的查询统计、商品库存信息的查询统计、销售单据和销售明细的查询统计。

因本系统为一个简化的商场业务系统模型，不考虑如财务核算、退货等其它功能的实现。

2．非功能性需求概述

根据商场经营管理的特点，本系统应满足以下几项性能要求：

（1）实时性，要求整个系统对基本销售业务实现秒级响应；

（2）并发性，满足多个终端同时操作数据库的要求；

（3）交互性，快速友好的人机交互界面；

（4）安全性，保证数据存储和传输的绝对安全，及时进行数据库备份；

（5）稳定性，保证整个系统的长期稳定运行。

（6）数据处理要求，商场每天平均业务约为 1 万次/天，系统应能在线保存至少一个年的销售数据。

2.8.2 数据对象

通过对商品零售经营管理领域的商品、顾客、员工等几类数据对象进行分析，可以归纳出本系统需要管理的 13 个客观数据对象。这 13 个数据对象分别具有不同的属性特征，用下述 13 张表格给出，表中的每一列描述了数据对象的某一方面的特征。

1．顾客（Table_Customer）

字段	字段名称	数据类型	可否为空	计量单位	精度要求	备注
CustomerID	顾客编号	varchar(8)	NOT NULL			
CardID	会员卡号	varchar(8)	NOT NULL			
CName	姓名	varchar(10)	NOT NULL			
Sex	性别	char(1)	NULL			M：男，F：女
Age	年龄	int	NULL			
IdentityCard	身份证	varchar(20)	NOT NULL			
Address	地址	varchar(50)	NULL			
Postcode	邮编	varchar(6)	NULL			
Tel	电话	varchar(15)	NOT NULL			

主键：CustomerID；

外键：CardID reference Table_Card (CardID)

2．会员卡（Table_Card）

字段	字段名称	数据类型	可否为空	计量单位	精度要求	备注
CardID	会员卡号	varchar(8)	NOT NULL			
StartDate	有效起始日期	datatime	NOT NULL			
EndDate	有效截止日期	datetime	NOT NULL			
Score	积分	int	NOT NULL			
State	状态	char(1)	NOT NULL			0 正常，1 禁用，2 挂失

主键：CardID

3．商品类别（Table_GoodsClass）

字段	字段名称	数据类型	可否为空	计量单位	精度要求	备注
GoodsClassID	商品类别编号	varchar(8)	NOT NULL			
GoodsClassName	类别名称	varchar(50)	NOT NULL			
GDescription	类别描述信息	varchar(80)	NULL			

主键：GoodsClassID

4．商品（Table_Goods）

字段	字段名称	数据类型	可否为空	计量单位	精度要求	备注

GoodsID	商品编号	varchar(8)	NOT NULL			
GoodsClassID	商品类别编号	varchar(8)	NOT NULL			
GoodsName	品名	varchar(50)	NOT NULL			
ProductionDate	生产日期	datetime	NOT NULL			
SaleUnitPrice	单价	money	NOT NULL	元	百分位	
TotalStorage	库存量	decimal(10,3)	NOT NULL			
GDescription	商品描述信息	varchar(80)	NULL			

主键：GoodsID；

外键：GoodsClassID reference Table_GoodsClass (GoodsClassID)

5．销售人员（Table_Salesperson）

字段	字段名称	数据类型	可否为空	计量单位	精度要求	备注
SalespersonID	销售人员编号	varchar(8)	NOT NULL			
SName	姓名	varchar(10)	NOT NULL			
Sex	性别	char(1)	NULL			M：男，F：女
Age	年龄	int	NULL			
IdentityCard	身份证	varchar(20)	NOT NULL			
Address	地址	varchar(50)	NULL			
Postcode	邮编	varchar(6)	NULL			
Tel	电话	varchar(15)	NOT NULL			

主键：SalespersonID。

6．收银台（Table_Cashier）

字段	字段名称	数据类型	可否为空	计量单位	精度要求	备注
CashierID	收银台编号	varchar(8)	NOT NULL			
State	状态	int	NOT NULL			0 正常，1 禁用，2 故障
CDescription	收银台描述信息	varchar(80)	NULL			

主键：CashierID。

7．销售单据（Table_SaleBill）

字段	字段名称	数据类型	可否为空	计量单位	精度要求	备注
SaleBillID	销售单编号	varchar(8)	NOT NULL			
CardID	会员卡号	varchar(8)	NULL			普通顾客为空
Score	本次积分	int	NULL			普通顾客为空
PayAmount	付款总金额	money	NOT	元	百分位	

			NULL			
SaleDate	销售日期	datetime	NOT NULL			
SalespersonID	销售人员编号	varchar(8)	NOT NULL			
CashierID	收银台编号	varchar(8)	NOT NULL			
SBDescription	销售单据描述	varchar(80)	NULL			

主键：SaleBillID；

外键：CardID reference Table_Card(CardID),
SalespersonID reference Table_Salesperson (SalespersonID),
CashierID reference Table_Cashier (CashierID)

8．销售单据明细（Table_SaleBillDetail）

字段	字段名称	数据类型	可否为空	计量单位	精度要求	备注
SaleBillID	销售单据编号	varchar(8)	NOT NULL			
GoodsID	商品编号	varchar(8)	NOT NULL			
UnitPrice	单价	money	NOT NULL	元	百分位	
Quantity	数量	int	NOT NULL	件		
Amount	总价	money	NOT NULL	元	百分位	打折之前总价
Discount	折扣率	int	NOT NULL			取值范围[0，100]
DiscountAmount	折扣总价	money	NOT NULL	元	百分位	打折之后总价

联合主键：SaleBillID，GoodsID；

外键： SaleBillID reference Table_SaleBill (SaleBillID),
GoodsID reference Table_Goods (GoodsID)

9．供应商信息（Table_Provider）

字段	字段名称	数据类型	可否为空	计量单位	精度要求	备注
ProviderID	供应商编号	int	NOT NULL			
PName	名称	varchar(50)	NOT NULL			
Address	地址	varchar(50)	NULL			
Postcode	邮编	varchar(6)	NULL			
Linkman	联系人	varchar(20)	NULL			
Tel	电话	varchar(15)	NULL			
Fax	传真	varchar(15)	NULL			
Web	网址	varchar(50)	NULL			
Email	电子邮箱	varchar(50)	NULL			
Bank	开户银行	varchar(20)	NULL			

AccountName	账户名称	varchar(20)	NULL			
Account	账户账号	varchar(20)	NULL			

主键：ProviderID。

10．库存表（Table_Stock）

字段	字段名称	数据类型	可否为空	计量单位	精度要求	备注
GoodsID	商品编号	varchar(8)	NOT NULL			
Stocks	库存量	int	NOT NULL	件		
Description	库存描述信息	varchar(80)	NULL			

主键：GoodsID；

外键：GoodsID reference Table_ Goods (GoodsID)

11．采购入库单据（Table_InStockBill）

字段	字段名称	数据类型	可否为空	计量单位	精度要求	备注
InStockBillID	入库单据编号	varchar(8)	NOT NULL			
InStockDate	入库日期	datetime	NOT NULL			
Operator	经手人	varchar(20)	NOT NULL			
ISBDescription	入库单据描述	varchar(80)	NULL			

主键：InStockBillID。

12．采购入库单据明细（Table_InStockBillDetail）

字段	字段名称	数据类型	可否为空	计量单位	精度要求	备注
InStockBillID	入库单据编号	varchar(8)	NOT NULL			
GoodsID	商品编号	varchar(8)	NOT NULL			
BuyPrice	进价	money	NOT NULL	元	百分位	
Quantity	数量	int	NOT NULL	件		
ProviderID	供应商编号	int	NOT NULL			
Buydate	购买日期	datetime	NOT NULL			

联合主键：InStockBillID，GoodsID；

外键：InStockBillID reference Table_InStockBill (InStockBillID)，

GoodsID reference Table_Goods (GoodsID)，

ProviderID reference Table_Provider (ProviderID)

13. 商品价格变动表（Table_PriceHistory）

字段	字段名称	数据类型	可否为空	计量单位	精度要求	备注
GoodsID	商品编号	varchar(8)	NOT NULL			
Startdate	开始日期	datetime	NOT NULL			
Enddate	结束日期	datetime	NOT NULL			
SalesUnitPrice	销售价格	money	NOT NULL			
Notes	注释	varchar(80)	NOT NULL			

联合主键：GoodsID, StartDate, EndDate；

外键：GoodsID reference Table_Goods (GoodsID),

需要特别指出的是，上述各表之后列出的主键和外键并非数据对象本身的直接特征，而是通过数据库概念设计和逻辑设计得到的、刻画数据对象相互间联系的描述信息，是将数据对象转换为关系模式之后对关系模式的完整性约束要求。具体设计方法参见第 4 章和第 5 章相关内容。此处只是为了说明各表之间的联系才特意列出了表的主键和外键。

2.8.3 数据对象间关联关系

上述 13 个数据对象间存在着一定的联系。一个正确的商品经营管理数据库既要存储这 13 个数据对象本身信息，也需要合理地组织管理这些数据对象之间的关联关系。因为商品经营管理系统中的顾客管理、商品入库管理、日常销售管理和查询统计等业务管理功能不仅需要访问数据库中的数据对象，也需要考虑些数据对象之间的关联关系。

考虑“顾客”和“会员卡”这 2 个数据对象。根据实际情况我们知道：商场可以向它的顾客发放会员卡，持有会员卡的顾客在购买商品时可以获得一定优惠。因此，“顾客”和“会员卡”这 2 个数据对象间的存在直接联系，这种联系可以命名为“持有”。分析联系“持有”的特征，我们可以进一步地获得以下信息：

（1）并不是所有的顾客都有会员卡，也就是并非每一个“顾客” 数据对象都一定通过“持有”这个联系与某个“会员卡”数据对象关联在一起。

（2）商场一旦发放一张会员卡，则这张会员卡一定对应于某个持有并使用它的顾客。也就是每一个“会员卡” 数据对象都一定通过“持有”这个联系与某个“顾客”数据对象关联在一起。

（3）一个顾客可以没有会员卡，也可以持有一张会员卡，但不允许持有多张会员卡。商场发放的每张会员卡只能由一个且只有一个顾客来使用。

当一个会员卡顾客为他购买的某种商品付款时，系统首先访问数据库中该顾客相关信息，如顾客编号、会员卡号、顾客姓名等。然后通过会员卡号访问数据库中该顾客持有的会员卡信息，根据会员卡有效期、积分、卡状态等信息决定该会员卡是否有效，如果有效可以进一步根据卡内积分和优惠折扣策略计算此次顾客购买本商品时的价格优惠幅度，完成本次交易。

识别分析商品零售经营管理领域中的各类数据对象及其联系是构建正确的商品零售经营管理数据库的关键，属于 DBAS 开发过程中概念数据库设计范畴。可以采用实体-联系模型中的“实体”这一建模元素表示上述“顾客”和“会员卡”数据对象，采用“联系” 这一建模元素表示“持有” 联系及其特征。

下图为描述“顾客”和“会员卡”的实体-联系图。具体含义参见第 4 章对 E-R 模型的说明。

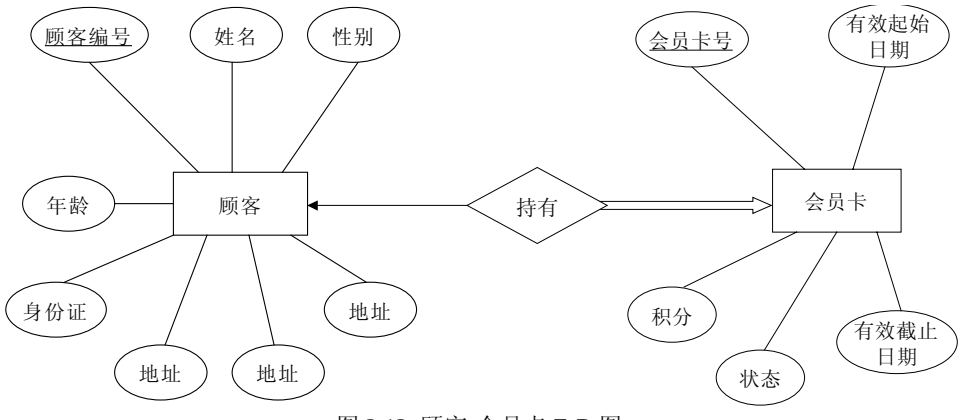

图 2.12 顾客-会员卡 E-R 图

通过分析各个数据对象的具体特征和相互间关联关系，可以建立起多个表示为 E-R 图

的用户局部视图。合并这些局部视图得到系统完整的数据库概念结构，在此基础上可以进行数据库的逻辑设计和物理设计。

2.9 小结

软件工程是指导计算机软件开发和维护的工程科学，是为了解决大型复杂软件系统开发时的软件危机问题而提出的。软件工程的目标是采用现代工程的概念、原理、技术和方法进行大型复杂软件的开发、管理、维护和更新，以提高软件质量，加快软件开发进度和降低开发费用。这一目标通过可以软件项目管理活动实现。

软件工程定义了软件生命周期模型，并根据软件开发模型对软件开发过程进行有效管理。常见的软件开发模型有瀑布模型、原型模型、增量模型、螺旋模型等。

数据库应用系统是面向数据管理和处理领域的复杂软件系统。根据软件工程基本原理和数据库系统三级模式结构，本章提出了 DBAS 生命周期模型，将 DBAS 生命周期划分为规划与分析、需求分析、设计、系统实现和部署、运行维护等 5 个阶段，并规定了各阶段的主要目标、工作内容和采用的关键技术。该模型针对 DBAS 需求分析和设计，引入数据组织与存储设计、数据访问与处理设计和应用设计三条设计主线，并将 DBAS 设计阶段细分为概念设计、逻辑设计、物理设计三个步骤，定义了 DBAS 开发过程整体框架和开发活动各阶段间的关系。DBAS 生命周期模型是 DBAS 项目管理的理论指导准则。

本章所描述的商场经营管理系统可以作为后续各章节所讨论的各种 DBAS 设计与开发技术的应用案例。

习题

2.1 名词解释

软件工程，软件生命周期，软件开发生命周期；软件过程，软件开发模型；

数据库应用系统需求；

2.2 简述瀑布模型、快速原型模型和增量模型的基本原理和开发步骤。

2.3 数据库应用系统的软件包括哪几种类型？

2.4 简述数据库应用系统生命周期模型的基本原理。

2.5 规划与分析的主要工作内容是什么？

2.6 DBAS 需求分析主要包括哪些内容？

2.7 DBAS 概念设计包括哪些内容？

2.8 DBAS 逻辑设计包括哪些内容？简述数据库逻辑结构设计的主要设计步骤。

2.9 DBAS 物理设计包括哪些内容？数据库事务详细设计的工作内容是什么？

2.10 DBAS 的系统实现和部署包括哪些工作内容？

2.11 DBAS 的运行与维护包括哪些工作内容？