预备知识

一、数据库概述

（一）什么是数据库

数据库是计算机学科领域中发展最为迅速的重要分支，在各行各业中得到了非常广泛的应用。在财务、图书资料、科研项目、银行账目、学籍档案等各个领域，已经建立了成千上万个信息系统，而数据库正是这些信息系统的基础、核心技术。

简单来说，数据库是“按照数据结构来组织、存储和管理数据的仓库”。在日常工作中，常常需要把某些相关的数据放进这样的“仓库”，并根据管理的需要进行相应的处理。例如，企业或事业单位的人事部门常常要把本单位职工的基本情况（职工号、姓名、出生年月、性别、籍贯、工资）存放在如下的职工基本情况表（见表 1）中。

表 1　职工基本情况表

职工号	姓名	出生年月	性别	籍贯	工资
zs100001	蔡碧清	1972-6	女	广东新会	5000
zs100002	蔡洪明	1963-5	男	湖南郴州	4500
zs100003	党少申	1982-3	男	湖北武汉	3100
zs100004	龚自真	1986-10	男	云南大理	3200
zs100005	何少华	1975-3	女	广西百色	4600

表 1 就可以看成是一个数据库。有了这个“数据仓库”我们就可以根据需要随时增加新职工的信息，也可以随时查询某职工的基本情况，也可以查询工资在某个范围内的职工人数等。当这些工作都在计算机上自动进行，我们的人事管理工作的效率就可以得到很大的提高。此外，在财务管理、仓库管理、生产管理中也需要建立众多的这种“数据库”，使其可以利用计算机实现财务、仓库、生产的自动化管理。可以这么说，数据库是所有的信息管理系统的核心组成部分。

J.Martin 给数据库下了一个比较完整的定义：数据库是存储在一起的相关数据的集合，这些数据是结构化的，无有害的或不必要的冗余，并为多种应用服务；数据的存储独立于使用它的程序；对数据库插入新数据，修改和检索原有数据均能按一种公用的和可控制的方式进行。当某个系统中存在结构上完全分开的若干个数据库时，则该系统包含一个“数据库集合”。

使用数据库可以带来许多好处：例如，对于用户来说提高了查询的效率，对于数据管理来说减少了数据的冗余度，节省了数据的存储空间，还可以实现数据资源的充分共享等。此外，

数据库技术还为用户提供了非常简便的使用手段，使用户易于编写有关数据库应用程序。特别是近年来推出的微型计算机关系数据库管理系统如 MS-SQL Server，Oracle，Access 等，操作直观，使用灵活，编程方便，数据处理能力极强。数据库在我国正得到越来越广泛的应用，必将成为经济管理的有力工具。

数据库是通过数据库管理系统（DBMS，Data Base Management System）软件来实现数据的存储、管理与使用的，例如上面所提到的 MS-SQL Server，Oracle，Access 等就是数据库管理系统软件（DBMS）。

数据库通常分为层次式数据库、网络式数据库和关系式数据库三种。而不同的数据库是按不同的数据结构来联系和组织的。本书不打算详细介绍这三种数据库的技术细节，目前市面上常用的数据库还是关系结构模型的，这里简单介绍关系数据库的一些基本概念。

（二）关系数据库

关系数据结构把一些复杂的数据结构归结为简单的二元关系（简单地说就像我们最常见的二维表格形式）。例如表 1 就是一个二元关系。这个六行六列的表格的每一列称为一个字段（即属性），字段名相当于标题栏中的标题（属性名称）；表的每一行是包含了六个属性（职工号，姓名，出生年月，性别，籍贯，工资）的一个六元组，即一个人的记录。这个表格清晰地反映出该单位职工的基本情况。

表中每一行表示一条记录值，每一列表示一个属性（即字段或数据项）。表 1 一共有 5 条记录。每个记录包含 6 个属性。

作为一个关系的二维表，必须满足以下条件：

（1）表中每一列必须是基本数据项（即不可再分解）。

（2）表中每一列必须具有相同的数据类型（例如字符型或数值型）。

（3）表中每一列的名字必须是唯一的。

（4）表中不应有内容完全相同的行。

（5）行的顺序与列的顺序不影响表格中所表示的信息的含义。

由关系数据结构组成的数据库系统被称为关系数据库系统，对于初学者来说，可以这么认为：所谓关系数据库，就是很多关系表格（二维表格）的集合。而对关系数据的操作，就是对关系型数据库管理系统（DBMS）中的某个或多个表格的操作，包括关系表格的创建、数据插入、查询、连接、删除等运算来实现数据的管理。SQL Server、Oracel、Access、MySQL 等就是这类数据库管理系统的典型代表。

（三）SQL 简介

SQL 是英文 Structured Query Language 的缩写，意思为结构化查询语言。SQL 语言的主要功能就是同各种数据库建立联系，进行沟通。按照 ANSI（美国国家标准协会）的规定，SQL 被作为关系型数据库管理系统的标准语言。SQL 语句可以用来执行各种各样的操作，例如更新数据库中的数据，从数据库中提取数据等。目前，绝大多数流行的关系型数据库管理系统，如 Oracle，Sybase，SQL Server，Access 等都采用了 SQL 语言标准。虽然很多数据库都对 SQL 语句进行了再开发和扩展，但是包括 Select，Insert，Update，Delete，Create 以及 Drop 在内的

标准的 SQL 命令仍然可以被用来完成几乎所有的数据库操作。

SQL 数据库的数据体系结构基本上是三级结构，但使用术语与传统关系模型术语不同。在 SQL 中，关系模式（模式）称为“基本表”（base table）；存储模式（内模式）称为“存储文件”（stored file）；子模式（外模式）称为“视图”（view）；元组称为“行”（row）；属性称为“列”（column）。

在正式学习 SQL 语言之前，我们先对 SQL 语言有一个基本认识，介绍一下 SQL 语言的组成：

（1）一个 SQL 数据库是表（Table）的集合，它由一个或多个 SQL 模式定义。

（2）一个 SQL 表由行集构成，一行是列的序列（集合），每列与行对应一个数据项。

（3）一个表或者是一个基本表或者是一个视图。基本表是实际存储在数据库的表，而视图是由若干基本表或其他视图构成的表的定义。

（4）一个基本表可以跨一个或多个存储文件，一个存储文件也可存放一个或多个基本表。每个存储文件与外部存储上的一个物理文件对应。

（5）用户可以用 SQL 语句对视图和基本表进行查询等操作。从用户角度来看，视图和基本表是一样的，没有区别，都是关系（表格）。

（6）SQL 用户可以是应用程序，也可以是终端用户。SQL 语句可嵌入在宿主语言的程序中使用，宿主语言有 FORTRAN，COBOL，PASCAL，PL/I，C 和 Ada 语言等。SQL 用户也能作为独立的用户接口，供交互环境下的终端用户使用。

二、数据库范式设计基本知识

关系数据库设计之时是要遵守一定的规则的，尤其是数据库设计范式。如果不遵守这些范式，往往会出现数据库表组织凌乱、数据大量冗余等问题，由于初学者理解范式往往有些困难，这里先简单介绍 1NF（第一范式），2NF（第二范式），3NF（第三范式）和 BCNF。如果读者对范式的内容觉得吃力或不能理解，可以先跳过直接往下学习，当对数据库有了一定理解以后再来学习范式会更好理解。

（一）第一范式（1NF）

在关系模式 R 中的每一个具体关系 r 中，如果每个属性值都是不可再分的最小数据单位，则称 R 是第一范式的关系。例如：职工号，姓名，电话号码组成一个表（一个人可能有一个办公室电话和一个家里电话号码）。规范成为 1NF 有三种方法：

- 重复存储职工号和姓名。这样，关键字只能是电话号码。
- 职工号为关键字，电话号码分为单位电话和住宅电话两个属性。
- 职工号为关键字，但强制每条记录只能有一个电话号码。

以上三种方法，第一种方法最不可取，按实际情况选取后两种情况。

（二）第二范式（2NF）

如果关系模式 R（U，F）中的所有非主属性都完全依赖于任意一个候选关键字，则称关系 R 是属于第二范式的。

例如：选课关系 SCI（SNO，CNO，GRADE，CREDIT）中 SNO 为学号，CNO 为课程号，GRADE 为成绩，CREDIT 为学分。 由以上条件得出，关键字为组合关键字（SNO，CNO）。

在应用中使用以上关系模式有以下问题：

（1）数据冗余，假设同一门课由 40 个学生选修，学分就重复 40 次。

（2）更新异常，若调整了某课程的学分，相应的元组 CREDIT 值都要更新，有可能会出现同一门课学分不同。

（3）插入异常，如计划开新课，由于没人选修，没有学号关键字，只能等有人选修才能把课程和学分存入。

（4）删除异常，若学生已经结业，从当前数据库删除选修记录。某些门课程新生尚未选修，则此门课程及学分记录无法保存。

原因：非关键字属性 CREDIT 仅函数依赖于 CNO，也就是 CREDIT 部分依赖组合关键字（SNO，CNO）而不是完全依赖。

解决方法：分成两个关系模式 SC1（SNO，CNO，GRADE），SC2（CNO，CREDIT）。新关系包括两个关系模式，它们之间通过 SC1 中的外关键字 CNO 相联系，需要时再进行自然联接，恢复成原来的关系。

（三）第三范式（3NF）

如果关系模式 R（U，F）中的所有非主属性对任何候选关键字都不存在传递信赖，则称关系 R 是属于第三范式的。

例如：S1（SNO，SNAME，DNO，DNAME，LOCATION）各属性分别代表学号，

姓名，所在系，系名称，系地址。

关键字 SNO 决定各个属性。由于是单个关键字，没有部分依赖的问题，肯定是 2NF。但这关系肯定有大量的冗余，有关学生所在系的几个属性 DNO，DNAME，LOCATION 将重复存储，插入，删除和修改时也将产生类似上例的情况。

原因：关系中存在传递依赖造成的。即 SNO -> DNO， 而 DNO -> SNO 却不存在，DNO -> LOCATION，因此关键字 SNO 对 LOCATION 函数决定是通过传递依赖 SNO -> LOCATION 实现的。也就是说，SNO 不直接决定非主属性 LOCATION。

解决目的：每个关系模式中不能留有传递依赖。

解决方法：分为两个关系 S（SNO，SNAME，DNO），D（DNO，DNAME，LOCATION）。

关系 S 中不能没有外关键字 DNO。否则两个关系之间失去联系。

（四）BCNF

如果关系模式 R（U，F）的所有属性（包括主属性和非主属性）都不传递依赖于 R 的任

何候选关键字，那么称关系 R 是属于 BCNF 的。或者关系模式 R，如果每个决定因素都包含关键字（而不是被关键字所包含），则是 RCNF 的关系模式。

例如：配件管理关系模式 WPE（WNO，PNO，ENO，QNT）分别表示仓库号，配件号，职工号，数量。有以下条件

（1）一个仓库有多个职工。

（2）一个职工仅在一个仓库工作。

（3）每个仓库里一种型号的配件由专人负责，但一个人可以管理几种配件。

（4）同一种型号的配件可以分放在几个仓库中。

分析：由以上得 PNO 不能确定 QNT，由组合属性（WNO，PNO）来决定，存在函数依赖（WNO，PNO）-> ENO。由于每个仓库里的一种配件由专人负责，而一个人可以管理几种配件，所以由组合属性（WNO，PNO）才能确定负责人，有（WNO，PNO）-> ENO。因为一个职工仅在一个仓库工作，有 ENO -> WNO。由于每个仓库里的一种配件由专人负责，而一个职工仅在一个仓库工作，有 （ENO，PNO）-> QNT。

找一下候选关键字，因为（WNO，PNO） -> QNT，（WNO，PNO）-> ENO ，因此 （WNO，PNO）可以决定整个元组，是一个候选关键字。根据 ENO->WNO，（ENO，PNO）->QNT，故（ENO，PNO）也能决定整个元组，为另一个候选关键字。属性 ENO，WNO，PNO 均为主属性，只有一个非主属性 QNT。它对任何一个候选关键字都是完全函数依赖的，并且是直接依赖，所以该关系模式是 3NF。

分析一下主属性。因为 ENO->WNO，主属性 ENO 是 WNO 的决定因素，但是它本身不是关键字，只是组合关键字的一部分。这就造成主属性 WNO 对另外一个候选关键字（ENO，PNO）的部分依赖，因为（ENO，PNO）-> ENO，但反过来不成立，而 PNO->WNO，故（ENO，PNO）-> WNO 也是传递依赖。

虽然没有非主属性对候选关键字的传递依赖，但存在主属性对候选关键字的传递依赖，同样也会带来麻烦。如一个新职工分配到仓库工作，但暂时处于实习阶段，没有独立负责对某些配件的管理任务。由于缺少关键字的一部分 PNO 而无法插入到该关系中去。又如某个人改成不管配件了去负责安全，则在删除配件的同时该职工也会被删除。

解决办法：分成管理 EP（ENO，PNO，QNT），关键字是（ENO，PNO）；工作 EW（ENO，WNO），关键字是 ENO。

缺点：分解后函数依赖的保持性较差。如此例中，由于分解，函数依赖（WNO，PNO）-> ENO 丢失了，因而对原来的语义有所破坏。没有体现出每个仓库里一种部件由专人负责。有可能出现一种部件由两个人或两个以上的人来同时管理。因此，分解之后的关系模式降低了部分完整性约束。

一个关系分解成多个关系，要使得分解有意义，起码的要求是分解后不丢失原来的信息。这些信息不仅包括数据本身，而且包括由函数依赖所表示的数据之间的相互制约。进行分解的目标是达到更高一级的规范化程度，但是分解的同时必须考虑两个问题：无损联接性和保持函数依赖。有时往往不可能做到既有无损联接性，又完全保持函数依赖。需要根据需要进行权衡。

从 1NF 直到 BCNF 的四种范式之间有如下关系：

$BCNF \subseteq 3NF \subseteq 2NF \subseteq 1NF$

目的：规范化目的是使结构更合理，消除存储异常，使数据冗余尽量小，便于插入、删

除和更新。

原则：遵从概念单一化“一事一地”原则，即一个关系模式描述一个实体或实体间的一种联系。规范的实质就是概念的单一化。

方法：将关系模式投影分解成两个或两个以上的关系模式。

要求：分解后的关系模式集合应当与原关系模式“等价”，即经过自然联接可以恢复原关系而不丢失信息，并保持属性间合理的联系。

一个关系模式结构分解可以得到不同关系模式集合，也就是说分解方法不是唯一的。最小冗余的要求必须以分解后的数据库能够表达原来数据库所有信息为前提来实现。其根本目标是节省存储空间，避免数据不一致性，提高对关系的操作效率，同时满足应用需求。实际上，并不一定要求全部模式都达到 BCNF 不可。有时故意保留部分冗余可能更方便数据查询。尤其对于那些更新频度不高，查询频度极高的数据库系统更是如此。

在关系数据库中，除了函数依赖之外还有多值依赖，联接依赖的问题，从而提出了第四范式，第五范式等更高一级的规范化要求。一般在数据库设计中比较少用到，这里就不展开讨论了。

三、分销系统的需求分析

（一）分销系统简介

分销管理系统是流通性企业所使用的最常见的信息系统，其目标是实时为大中型生产企业以及商品流通批发企业提供分销价值链合作伙伴、企业分支机构和（或）专卖店的订货、虚拟库存、销售退货等方面的即时和汇总数据，同时这些数据可以按照单品、产品分类、地域、分销机构统计，为企业调整订货计划、产品市场策略等提供更加准确的决策依据。由于互联网的快速发展，分销管理系统更得到广泛的应用。在本书里，将以一个简化的典型分销系统来展开数据库的学习。

（二）分销系统的总体结构

简单的分销系统主要包括采购管理模块、仓库管理模块、销售管理模块和财务管理模块。图 1 是典型分销系统的结构图。

（三）功能描述

1．采购管理模块

采购管理系统主要核算企业采购货品的业务过程，您可以与供应商签订相应的订单，然

后在收到货品时根据订单将货品办理入库手续。采购货款则可以通过付款单予以支付。该模块主要功能为供应商资料和采购订单，具体如图 2、图 3、图 4 所示。

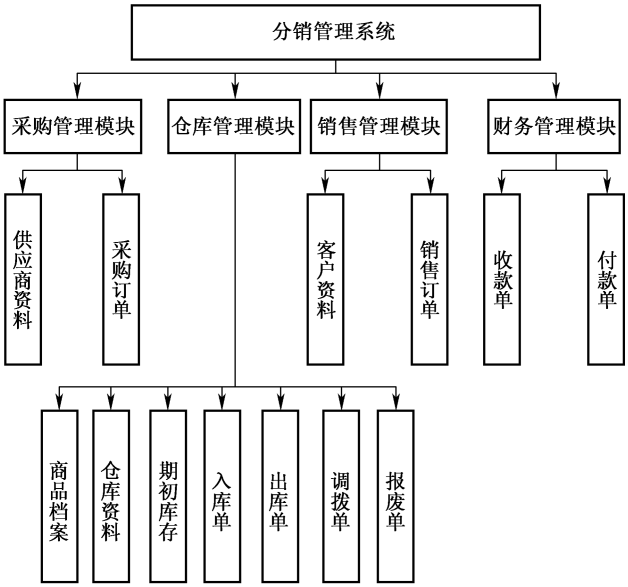

图 1　分销系统结构图

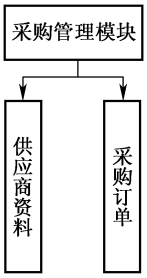

图 2

供应商资料

供应商编码：	G0001	**供应商名称：**	金花食品有限公司
联系人：	张三风	**电话：**	020-88888888
传真：	020-88888888		
地址：	广州市北京路 219 号		

图 3　供应商资料

采购订单

采购订单号： CG0001　　**日期：** 2008-12-25
供应商编码： G0001　　**供应商名称：** 金花食品有限公司
联系人： 张三风　　**联系电话：** 020-88888888
总金额： 1900
备注：

序号	商品编码	商品名称	规格型号	单位	数量	单价	金额	备注
01	SP001	美好时光海苔	10×5 克/包	包	100	15	1500	
02	SP002	恰恰瓜子	250 克/包	包	200	2	400	

图 4　采购订单

2．销售管理模块

销售管理模块提供客户管理、销售订货信息管理等。由分销系统根据现有的库存情况，将对应商品办理出库手续，并通过收款单进行收取客户所欠的货款。该模块主要功能为客户资料和销售订单，具体如图 5、图 6、图 7 所示。

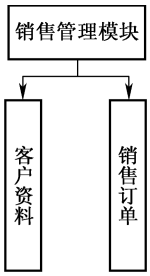

图 5

客户资料

客户编码： C0001　　**客户名称：** 好又多超市
联系人： 张三　　**电话：** 020-85530888
传真： 020-85530888
地址： 中山五路 219 号
送货地址： 中山五路 219 号

图 6　客户资料

销售订单

销售订单号： XS0001　　**日期：** 2008-12-25
客户编码： C0001　　**客户名称：** 好又多超市
联系人： 张三　　**联系电话：** 020-85530888
送货地址： 中山五路 219 号
总金额： 2600
备注：

序号	商品编码	商品名称	规格型号	单位	数量	单价	金额	备注
01	SP001	美好时光海苔	10×5 克/包	包	100	20	2000	
02	SP002	恰恰瓜子	250 克/包	包	200	3	600	

图 7　销售订单

3．仓库管理模块

仓库管理模块主要对商品档案进行维护，并配合销售、采购模块进行相应的出库、入库的操作。仓库管理模块还提供对不同仓库的库存管理，包括期初库存、仓库之间的商品调拨，仓库商品报废的功能。该模块的主要功能包括商品档案、仓库资料、期初库存、入库单、出库单、调拨单和报废单，具体如图 8 到图 15 所示。

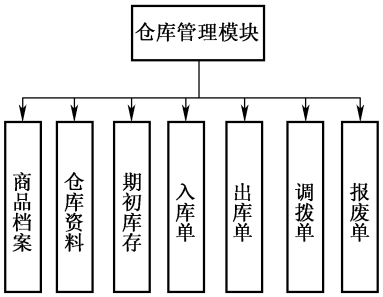

图 8

商品档案

商品编码： SP001　　**商品名称：** 美好时光海苔
规格型号： 10×5 克/包　　**单位：** 包
主供应商编码： C0001　　**参考单价** **15**
备注：

图 9　商品档案

仓库资料

仓库编码： CK0001　　**仓库名称：** 海珠食品仓

仓库位置： 新港西路 219 号

备注： 存放食品类商品

仓位编码	仓位名称	备注
A	A 区	
B	B 区	
C	C 区	

图 10　仓库资料

期初库存

仓库编码： CK0001　　**仓库名称：** 海珠食品仓

商品编码： SP001　　**商品名称：** 美好时光海苔

规格型号： 10×5 克/包　　**单位：** 包

期初数量： 1000　　**期初单价** **15**

期初金额： 15000

备注：

图 11　期初库存

入库单

入库单号： CG0001　　**日期：** 2008-12-25

供应商编码： G0001　　**供应商名称：** 金花食品有限公司

联系人： 张三风　　**联系电话：** 020-88888888

总金额： 1900

备注：

序号	采购订单编号	商品编码	商品名称	规格型号	单位	数量	单价	金额	仓库编码	仓位编码	备注
01	XS0001	SP001	美好时光海苔	10×5 克/包	包	100	15	1500	CK0001	A 区	
02	XS0001	SP002	恰恰瓜子	250 克/包	包	200	2	400	CK0001	B 区	

图 12　入库单

出库单

出库单号： CG0001　　**日期：** 2008-12-25
客户编码： C0001　　**客户名称：** 好又多超市
联系人： 张三　　**联系电话：** 020-85530888
送货地址： 中山五路 219 号
总金额： 2600
备注：

序号	销售订单编号	商品编码	商品名称	规格型号	单位	数量	单价	金额	仓库编码	仓位编码	备注
01	CG0001	SP001	美好时光海苔	10×5 克/包	包	100	20	2000	CK0001	A 区	
02	CG0001	SP002	恰恰瓜子	250 克/包	包	200	3	600	CK0001	B 区	

图 13　出库单

调拨单

调拨单号： DB0001　　**调拨日期：** 2008-12-25
调出仓库编码： CK0001　　**调出仓位编码：** A 区
调入仓库编码： CK0002　　**调入仓位编码：** A 区
调拨人： 吴海
备注：

序号	商品编码	商品名称	规格型号	单位	可用库存	调拨数量	备注
01	SP001	美好时光海苔	10×5 克/包	包	100	20	

图 14　调拨单

报废单

报废单号： BF0001　　**报废日期：** 2008-12-25
报废人： 吴海
备注：

序号	仓库编码	仓位编码	商品编码	商品名称	规格型号	单位	报废数量	报废原因	备注
01	CK0001	A 区	SP001	美好时光海苔	10×5 克/包	包	5	包装破损	

图 15　报废单

4．财务管理模块

财务管理模块主要是对应收款、应付款的管理。包括对应销售模块的收款单和采购模块的付款单，并对应收款、应付款进行维护。该模块主要包括收款单和付款单。具体如图 16 到图 18 所示说明。

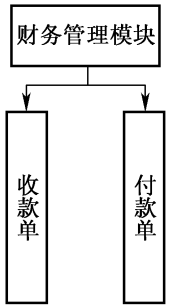

图 16　财务管理模块结构图

收款单

收款单号：	SK0001	**收款日期：**	2008-12-25
客户编码：	C0001	**客户名称：**	好又多超市
应收总额：	50000	**收款金额：**	20000
备注：			

图 17　收款单

付款单

付款单号：	BF0001	**付款日期：**	2008-12-25
供应商编码：	G0001	**供应商名称：**	金花食品有限公司
应付总额：	30000	**付款金额：**	20000
备注：			

图 18　付款单

四、数据库建模分析

数据库建模指的是对现实世界各类数据的抽象组织，确定数据库需管辖的范围、数据的组织形式等直至转化成现实的数据库。将经过系统分析后抽象出来的概念模型转化为物理模型后，在 Visio 或 PowerDesigner 等工具建立数据库实体以及各实体之间关系的过程（实体一般是表）。

在数据库建模时，一般根据现有的表格，对其中的数据进行分析，同时要兼顾数据库设计范式，一般我们设计出来的数据库模型起码要符合 2NF 以上才算是合格的关系数据库。

下面以采购订单为例分析数据库建模的过程。

图 19 是一个采购订单的实际单据。其中，对于供应商的资料：**供应商编码、供应商名称、联系人、联系电话**等，因为对于采购订单来说，这些资料是会重复存储的，而且供应商的资料也可能不止这几个字段，所以，对于供应商的资料，应该另外建一个表“供应商资料”来存储这些数据。接着看下面的有关商品资料的几个字段。

采购订单

采购订单号：CG0001　　**日期**：2008-12-25
供应商编码：G0001　　**供应商名称**：金花食品有限公司
联系人：张三风　　**联系电话**：020-88888888
总金额：1900
备注：

序号	商品编码	商品名称	规格型号	单位	数量	单价	金额	备注
01	SP001	美好时光海苔	10X5 克/包	包	100	15	1500	
02	SP002	恰恰瓜子	250 克/包	包	200	2	400	

图 19　采购订单

序号	商品编码	商品名称	规格型号	单位	数量	单价	金额	备注

可以看出对于一张采购订单来说，可能存在多个商品的信息。根据规范化的需求，则要把这几个字段另外建立一个 “采购订单明细表”来存储这些数据。另外，对于“采购订单”和“采购订单明细表”，我们需要对这两个表的某个字段做关联，即我们熟知的外键关联关系，所以我们对“采购订单明细表”还必须加上一个字段“采购订单号”来标识这些数据是属于哪张采购订单的。另外，对于商品资料的信息，我们还必须另外建立一个表“商品资料”来存储所有的商品的信息。

这样，我们对于如图 19 所示采购订单的分析，可以得出四个数据库的表格：“供应商资料”、“采购订单”、“采购订单明细表”、“商品资料”。具体见表 2 至表 5。

表 2　供应商资料

字段名称	数据类型	是否允许为空	是否为主键	备注
供应商编码	varchar(20)	否	是	
供应商名称	varchar(100)	否	否	
联系人	varchar(20)	否	否	
电话	varchar(50)	否	否	
传真	varchar(50)	是	否	
地址	varchar(200)	是	否	

表 3　采购订单

字段名称	数据类型	是否允许为空	是否为主键	备注
采购订单号	varchar(20)	否	是	
日期	datetime	否	否	
供应商编码	varchar(20)	否	否	外键：供应商资料（供应商编码）
供应商名称	varchar(100)	否	否	
联系人	varchar(20)	否	否	
联系电话	varchar(50)	否	否	
总金额	numeric(12,4)	否	否	
备注	varchar(500)	是	否	

表 4　采购订单明细表

字段名称	数据类型	是否允许为空	是否为主键	备注
采购订单号	varchar(20)	否	组合主键	外键：销售订单（销售订单号）
序号	int	否		
商品编码	varchar(20)	否	否	外键：商品档案（商品编码）
商品名称	varchar(50)	否	否	
规格型号	varchar(100)	否	否	
单位	varchar(8)	否	否	
数量	numeric(12,2)	否	否	
单价	numeric(12,4)	否	否	
金额	numeric(12,4)	否	否	
备注	varchar(500)	是	否	

表 5　商品资料

字段名称	数据类型	是否允许为空	是否为主键	备注
商品编码	varchar(20)	否	是	
商品名称	varchar(50)	否	否	
规格型号	varchar(100)	否	否	
单位	varchar(8)	否	否	
主供应商编码	varchar(50)	是	否	外键：供应商资料（供应商编码）
参考单价	numeric(12,4)	是	否	
备注	varchar(500)	是	否	

对于数据库建模大概是按照上面的原则来进行。当然，怎样的数据库模型是最好的，要根据具体情况来分析。所以对于数据库建模来说，不同的人，不同的经验，不同的客户需求，都可能造成模型的不同。下面，对于整个分销系统，我们分模块来给出数据库模型（不再进行详细的分析），当然这个模型只是简单的模型，只是供读者学习数据库参考用，并不代表该模型能适合实际的客户需求。

（一）采购管理模块

描述采购管理模块的表格一共有三个：供应商资料表、采购订单表和采购订单明细表。其中采购订单表因为一张采购订单可以对应多个商品，根据数据库的规范化，将其拆分成采购订单表和采购订单明细表两个表。使用 PowerDesigner 定义这三个表的字段，以及每个表的主键、外键和引用关系，如图 20 和表 6、表 7、表 8 所示。

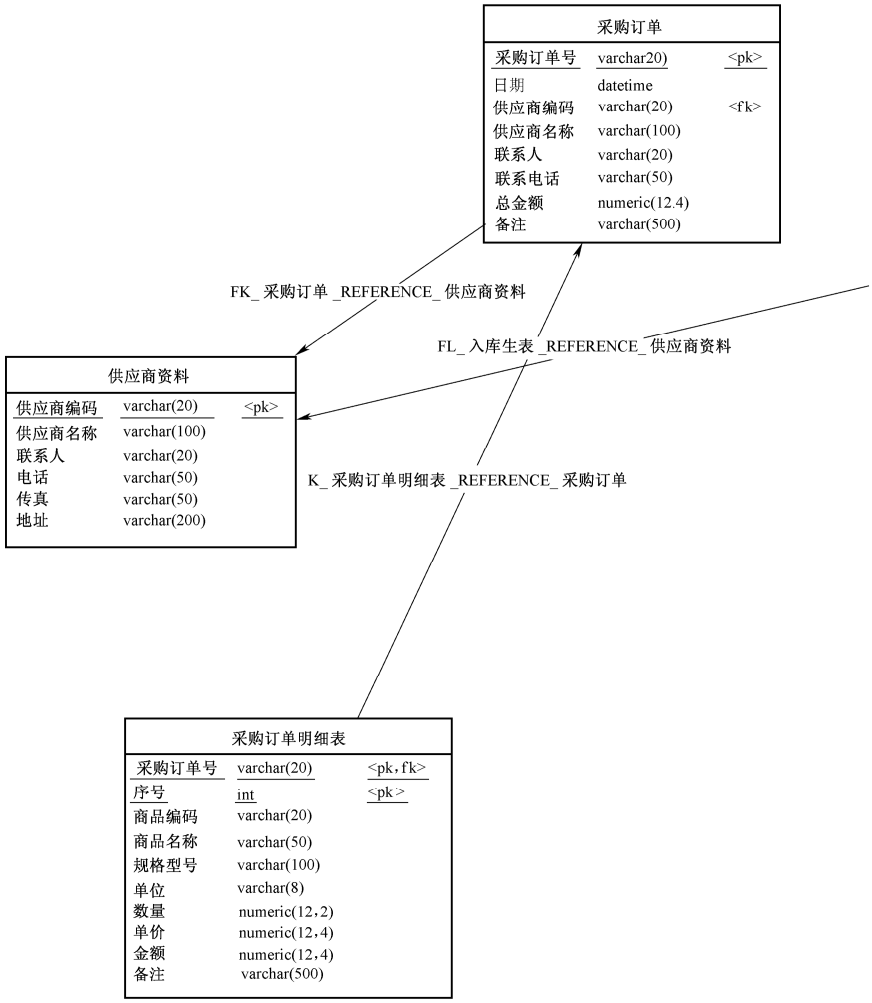

图 20

表 6　供应商资料

字段名称	数据类型	是否允许为空	是否为主键	备注
供应商编码	varchar(20)	否	是	
供应商名称	varchar(100)	否	否	
联系人	varchar(20)	否	否	
电话	varchar(50)	否	否	
传真	varchar(50)	是	否	
地址	varchar(200)	是	否	

表 7　采购订单

字段名称	数据类型	是否允许为空	是否为主键	备注
采购订单号	varchar(20)	否	是	
日期	datetime	否	否	
供应商编码	varchar(20)	否	否	外键：供应商资料（供应商编码）
供应商名称	varchar(100)	否	否	
联系人	varchar(20)	否	否	
联系电话	varchar(50)	否	否	
总金额	numeric(12,4)	否	否	
备注	varchar(500)	是	否	

表 8　采购订单明细表

字段名称	数据类型	是否允许为空	是否为主键	备注
采购订单号	varchar(20)	否	组合主键	外键：销售订单（销售订单号）
序号	int	否		
商品编码	varchar(20)	否	否	外键：商品档案（商品编码）
商品名称	varchar(50)	否	否	
规格型号	varchar(100)	否	否	
单位	varchar(8)	否	否	
数量	numeric(12,2)	否	否	
单价	numeric(12,4)	否	否	
金额	numeric(12,4)	否	否	
备注	varchar(500)	是	否	

（二）销售管理模块

描述销售管理模块的表格一共有三个：客户资料表、销售订单表和销售订单明细表。其中销售订单因为一张销售订单可以对应多个商品，根据数据库的规范化，将其拆分成销售订单和销售订单明细表两个表。使用 PowerDesigner 定义这三个表的字段，以及每个表的主键、外键和引用关系，如图 21 和表 9、表 10、表 11 所示。

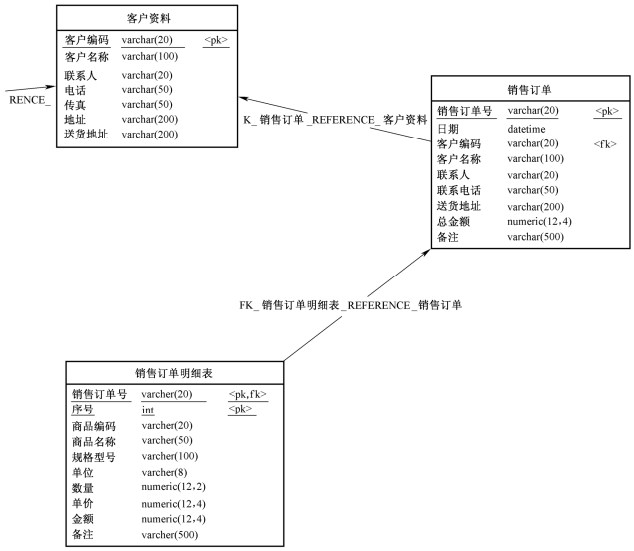

图 21

表 9　客户资料

字段名称	数据类型	是否允许为空	是否为主键	备注
客户编码	varchar(20)	否	是	
客户名称	varchar(100)	否	否	
联系人	varchar(20)	否	否	
电话	varchar(50)	否	否	
传真	varchar(50)	是	否	
地址	varchar(200)	是	否	
送货地址	varchar(200)	是	否	

表 10　销售订单

字段名称	数据类型	是否允许为空	是否为主键	备注
销售订单号	varchar(20)	否	是	
日期	datetime	否	否	
客户编码	varchar(20)	否	否	外键：客户资料（客户编码）
客户名称	varchar(100)	否	否	
联系人	varchar(20)	否	否	
联系电话	varchar(50)	否	否	
送货地址	varchar(200)	否	否	
总金额	numeric(12,4)	否	否	
备注	varchar(500)	是	否	

表 11　销售订单明细表

字段名称	数据类型	是否允许为空	是否为主键	备注
销售订单号	varchar(20)	否	组合主键	外键：销售订单（销售订单号）
序号	int	否		
商品编码	varchar(20)	否	否	外键：商品档案（商品编码）
商品名称	varchar(50)	否	否	
规格型号	varchar(100)	否	否	
单位	varchar(8)	否	否	
数量	numeric(12,2)	否	否	
单价	numeric(12,4)	否	否	
金额	numeric(12,4)	否	否	
备注	varchar(500)	是	否	

（三）仓库管理模块

描述仓库管理模块的表格一共有 13 个：其中仓库资料表、出库单、入库单、盘点表、调拨单、报废单，根据数据库的规范化，将它们都拆分成主表和明细表的形式。使用 PowerDesigner 定义这 13 个表的字段，以及每个表的主键、外键和引用关系，如图 22、图 23 和表 12 至表 23 所示。

图 22

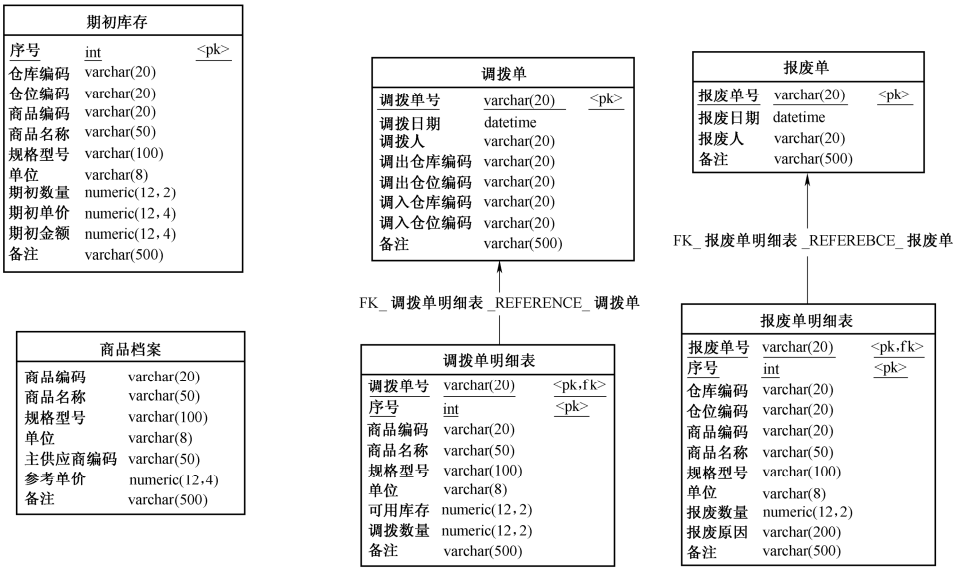

图 23

表 12　商品档案

字段名称	数据类型	是否允许为空	是否为主键	备注
商品编码	varchar(20)	否	是	
商品名称	varchar(50)	否	否	
规格型号	varchar(100)	否	否	
单位	varchar(8)	否	否	
主供应商编码	varchar(50)	是	否	外键：供应商资料（供应商编码）
参考单价	numeric(12,4)	是	否	
备注	varchar(500)	是	否	

表 13　仓库资料

字段名称	数据类型	是否允许为空	是否为主键	备注
仓库编码	varchar(20)	否	是	
仓库名称	varchar(50)	否	否	
仓库位置	varchar(500)	否	否	
备注	varchar(500)	是	否	

表 14　仓位资料

字段名称	数据类型	是否允许为空	是否为主键	备注
仓库编码	varchar(20)	否	组合主键	外键：仓库资料（仓库编码）
仓位编码	varchar(20)	否		
仓位名称	varchar(50)	否	否	
备注	varchar(500)	是	否	

表 15　期初库存

字段名称	数据类型	是否允许为空	是否为主键	备注
序号	int	否	是	标识列
仓库编码	varchar(20)	否	否	外键：仓库资料（仓库编码）
仓位编码	varchar(20)	否	否	外键：仓位资料（仓位编码）
商品编码	varchar(20)	否	否	
商品名称	varchar(50)	否	否	
规格型号	varchar(100)	否	否	
单位	varchar(8)	否	否	
期初数量	numeric(12,2)	否	否	
期初单价	numeric(12,4)	否	否	
期初金额	numeric(12,4)	否	否	
备注	varchar(500)	否	否	

表 16　入库单

字段名称	数据类型	是否允许为空	是否为主键	备注
入库单号	varchar(20)	否	是	
日期	datetime	否	否	
供应商编码	varchar(20)	否	否	外键：客户资料（客户编码）
供应商名称	varchar(100)	否	否	
联系人	varchar(20)	否	否	
联系电话	varchar(50)	否	否	
总金额	numeric(12,4)	否	否	
备注	varchar(500)	是	否	

表 17　入库单明细表

字段名称	数据类型	是否允许为空	是否为主键	备注
入库单号	varchar(20)	否	组合主键	外键：入库单（入库单号）
序号	int	否		
采购订单编号	varchar(20)			外键：采购订单（采购订单号）
商品编码	varchar(20)	否	否	外键：商品档案（商品编码）
商品名称	varchar(50)	否	否	
规格型号	varchar(100)	否	否	
单位	varchar(8)	否	否	

续表

字段名称	数据类型	是否允许为空	是否为主键	备注
数量	numeric(12,2)	否	否	
单价	numeric(12,4)	否	否	
金额	numeric(12,4)	否	否	
仓库编码	varchar(20)	否	否	外键：仓库资料（仓库编码）
仓位编码	varchar(20)	否	否	外键：仓位资料（仓位编码）
备注	varchar(500)	是	否	

表 18　出库单

字段名称	数据类型	是否允许为空	是否为主键	备注
出库单号	varchar(20)	否	是	
日期	datetime	否	否	
客户编码	varchar(20)	否	否	外键：客户资料（客户编码）
客户名称	varchar(100)	否	否	
联系人	varchar(20)	否	否	
联系电话	varchar(50)	否	否	
送货地址	varchar(200)	否	否	
总金额	numeric(12,4)	否	否	
备注	varchar(500)	是	否	

表 19　出库单明细表

字段名称	数据类型	是否允许为空	是否为主键	备注
出库单号	varchar(20)	否	组合主键	外键：出库单（出库单号）
序号	int	否		
销售订单编号	varchar(20)			外键：销售订单（销售订单号）
商品编码	varchar(20)	否	否	外键：商品档案（商品编码）
商品名称	varchar(50)	否	否	
规格型号	varchar(100)	否	否	
单位	varchar(8)	否	否	
数量	numeric(12,2)	否	否	

续表

字段名称	数据类型	是否允许为空	是否为主键	备注
单价	numeric(12,4)	否	否	
仓库编码	varchar(20)	否	否	外键：仓库资料（仓库编码）
仓位编码	varchar(20)	否	否	外键：仓位资料（仓位编码）
金额	numeric(12,4)	否	否	
备注	varchar(500)	是	否	

表 20　调拨单

字段名称	数据类型	是否允许为空	是否为主键	备注
调拨单号	varchar(20)	否	是	
调拨日期	datetime	否	否	
调出仓库编码	varchar(20)	否	否	外键：仓库资料（仓库编码）
调出仓位编码	varchar(20)	否	否	外键：仓位资料（仓位编码）
调入仓库编码	varchar(20)	否	否	外键：仓库资料（仓库编码）
调入仓位编码	varchar(20)	否	否	外键：仓位资料（仓位编码）
调拨人	varchar(20)	否	否	
备注	varchar(500)	是	否	

表 21　调拨单明细表

字段名称	数据类型	是否允许为空	是否为主键	备注
调拨单号	varchar(20)	否	组合主键	外键：调拨单（调拨单号）
序号	int	否		
商品编码	varchar(20)	否	否	外键：商品档案（商品编码）
商品名称	varchar(50)	否	否	
规格型号	varchar(100)	否	否	
单位	varchar(8)	否	否	
可用库存	numeric(12,2)	是	否	
调拨数量	numeric(12,2)	否	否	
备注	varchar(500)	是	否	

表 22　报废单

字段名称	数据类型	是否允许为空	是否为主键	备注
报废单号	varchar(20)	否	是	
报废日期	datetime	否	否	
报废人	varchar(20)	否	否	
备注	varchar(500)	是	否	

表 23　报废单明细表

字段名称	数据类型	是否允许为空	是否为主键	备注
报废单号	varchar(20)	否	组合主键	外键：报废单（报废单号）
序号	int	否		
仓库编码	varchar(20)	否	否	外键：仓库资料（仓库编码）
仓位编码	varchar(20)	否	否	外键：仓位资料（仓位编码）
商品编码	varchar(20)	否	否	外键：商品档案（商品编码）
商品名称	varchar(50)	否	否	
规格型号	varchar(100)	否	否	
单位	varchar(8)	否	否	
报废数量	numeric(12,2)	否	否	
报废原因	varchar(200)	是	否	
备注	varchar(500)	是	否	

（四）财务管理模块

描述财务管理模块的表格一共有两个：收款单、付款单。使用 PowerDesigner 定义这两个表的字段，以及每个表的主键、外键和引用关系，如图 24 和表 24、表 25 所示。

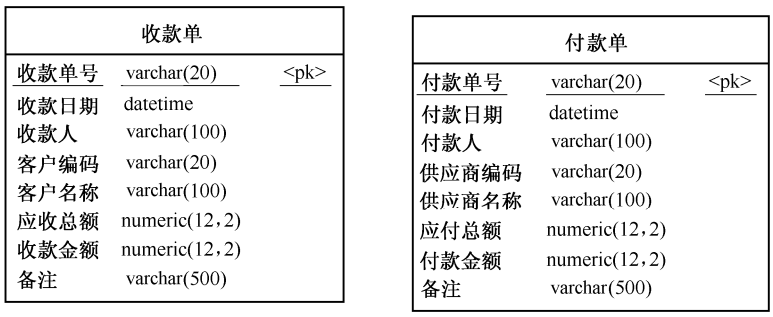

图 24

表 24　收款单

字段名称	数据类型	是否允许为空	是否为主键	备注
收款单号	varchar(20)	否	是	
收款日期	datetime	否	否	
收款人	varchar(100)	否	否	
客户编码	varchar(20)	否	否	外键：客户资料（客户编码）
客户名称	varchar(100)	否	否	
应收总额	numeric(12,2)	否	否	
收款金额	numeric(12,2)	否	否	
备注	varchar(500)	是	否	

表 25　付款单

字段名称	数据类型	是否允许为空	是否为主键	备注
付款单号	varchar(20)	否	是	
付款日期	datetime	否	否	
付款人	varchar(100)	否	否	
供应商编码	varchar(20)	否	否	外键：供应商资料（供应商编码）
供应商名称	varchar(100)	否	否	
应付总额	numeric(12,2)	否	否	
付款金额	numeric(12,2)	否	否	
备注	varchar(500)	是	否	

任务 1 分销系统数据库的设计与生成

面对纷繁复杂的海量信息，如何对其进行有效的管理和利用？数据库技术应运而生，并发展成为一门综合性数据管理技术。由 Microsoft 发布的 SQL Server 产品是一款典型的关系型数据库管理系统，并且应用越来越广泛。

任务目标

- 掌握数据库、数据库管理系统和数据库系统的基本概念。
- 了解数据模型的概念、掌握关系数据结构，掌握关系数据库的定义。
- 了解并基本掌握 E-R 实体联系图。
- 掌握 SQL Server 数据库的基本组成和有关知识。
- 初步了解 Transact-SQL 语句的基础知识。
- 掌握数据库的创建。

教学任务

- 介绍数据库的基本概念。
- 介绍数据库系统模型。
- 介绍关系模型的完整性约束。
- 介绍关系型数据库范式理论。
- 设计分销系统数据库。
- 介绍 SQL Server 数据库的常用对象、数据库结构和系统数据库等基本知识。
- SQL Server Management Studio 的初步使用。
- 用图形化工具创建数据库。
- 使用 Transact-SQL 创建数据库。

1.1 分销系统数据库的规划设计

分销系统对营销业务中的客户资料、销售订单、供应商资料、采购订单、付款单、收款单以及仓库物资等进行管理，设计良好的数据库能给系统的高效运作提供保障。

1.1.1 数据库的基本概念

数据库系统由数据库和数据库管理系统（DataBase Management System，DBMS）两部分组成。

1．一些相关概念

（1）数据。数据是人们用来反映客观世界而记录下来的可以鉴别的物理符号。

数据的概念不再仅是指狭义的数值数据，而是包括文字、声音、图形等一切能被计算机接收且能被处理的符号。数据是事物特性的反映和描述，是符号的集合。

（2）数据处理。数据处理是对各种形式的数据进行收集、存储、加工和传播的一系列活动的总和。

数据是重要的资源，把收集到的大量数据经过加工、整理、转换，从中获取有价值的信息，数据处理正是指将数据转换成信息的过程。

（3）数据管理。数据处理的中心问题是数据管理。数据管理是对数据的分类、组织、编码、存储、检索与维护。

（4）数据库。数据库是存储在一起的相互有联系的数据集合。

数据库是数据库系统的核心和管理对象。数据库中的数据是集成的、共享的、最小冗余的、能为多种应用服务的。

（5）数据库技术。数据库技术是研究如何科学地组织和存储数据，如何高效地获取和处理数据。

（6）数据库技术特点。数据库技术特点是面向整体组织数据逻辑结构，具有较高的数据和程序独立性，具有统一的数据控制功能（完整性控制、安全性控制、并发控制）。

2．数据库管理系统（DBMS）的基本功能

数据库管理系统是一个管理数据库的软件，简称 DBMS。它是数据库系统的核心。DBMS 为用户提供方便的用户接口，帮助和控制每个用户对数据库进行的各种操作。其基本功能如下：

（1）数据库定义。数据库管理系统必须首先能充分定义并管理各种类型的数据项。例如，关系型数据库管理系统必须建立数据库和数据表，定义字段的数据类型、限制以及数据之间的关联等。

（2）数据库处理。数据库管理系统必须能为用户提供对数据库存取的能力，这些能力包括增加、删除、修改和查询等。有时候并不是所有的要求都可以由数据库管理系统提供，因此需要编制相应的应用程序来满足特殊的需求。

（3）数据库控制。数据库管理系统的核心工作是对数据库的运行进行管理，包括：

1）数据库安全性控制功能。应该具备创建用户账号、相应的口令以及设置权限等功能。这样就可以使每个用户只能访问他们拥有访问权限的数据，从而避免不必要的人为损失，以保证数据库中数据的安全。

2）数据库完整性控制功能。完整性是数据的准确性和一致性的量度。

3）并发控制功能。数据库是提供给多个用户共享的，因此用户对数据的存取可能是并发的，即多个用户可能使用同一个数据库，因此数据库管理系统应能对多个用户的并发操作加以控制、协调。

4）数据库恢复功能。数据库中数据的安全除了可能受到人为破坏以外，同时还会受到意外事件破坏的威胁，因此数据库管理系统需要为用户提供准确、方便的备份功能。这样，就可以根据需要备份数据，并且在意外事件发生而导致数据丢失的情况下，将数据损失降至最低。

（4）数据字典。数据字典（Data Dictionary，DD）中存放着对实际数据库各级模式所做的定义，即对数据库结构的描述。对数据库的使用和操作都要通过查阅数据字典来进行。

3．数据库管理系统（DBMS）的基本功能

数据库管理系统将具有一定结构的数据组成一个集合，它主要具有以下几个特点：

（1）数据的结构化。数据库中的数据并不是杂乱无章、毫不相干的，它们具有一定的组织结构，属于同一集合的数据具有相似的特征。

（2）数据的共享性。在一个单位的各个部门之间，存在着大量的重复信息。使用数据库的目的就是要统一管理这些信息，减少冗余度，使各个部门共同享有相同的数据。

（3）数据的独立性。数据的独立性是指数据记录和数据管理软件之间的独立。数据及其结构应具有独立性，而不应该去改变应用程序。

（4）数据的完整性。数据的完整性是指保证数据库中数据的正确性。可能造成数据不正确的原因很多，数据库管理系统正是通过对数据性质进行检查而管理它们。

（5）数据的灵活性。数据库管理系统不是把数据简单堆积，而是在记录数据信息的基础上具有很多的管理功能，如输入、输出、查询、编辑、修改等。

（6）数据的安全性。根据用户的职责，不同级别的人对数据库具有不同的权限，数据库管理系统应该确保数据的安全性。

现在，比较流行的常用数据库管理系统有 Microsoft SQL Server、Oracle、Sybase 等。

1.1.2　数据库系统模型

数据库系统模型是指数据库中数据的存储结构。它是反映客观事物及其联系的数据描述形式。

数据模型通常由数据结构、数据操作和完整性约束等三部分组成，分别描述数据库系统的静态特性、动态特性和完整性约束条件。

在数据库系统的发展史上，最有影响的数据库模型有三个：层次模型、网状模型、关系模型。

1．层次型数据库

这种模型描述数据的组织形式像一棵倒置的树，它由节点和连线组成，其中节点表示实体。树有根、枝、叶，在这里都称为节点，根节点只有一个，向下分支，它是一种一对多的关系。如国家的行政机构、一个家族族谱的组织形式都可以看做是层次模型。

此种类型数据库的优点是数据结构类似于金字塔，层次分明、结构清晰、不同层次间的数据关联直接简单；缺点是数据将不得不以纵向向外扩展，节点之间很难建立横向的关联，因

此不利于系统的管理和维护。

2．网络型数据库

这种模型描述事物及其联系的数据组织形式像一张网，节点表示数据元素，节点间联线表示数据间联系。节点之间是平等的，无上下层关系。如学校中的“教师”、“学生”、“课程”、“教室”等事物之间有联系但无层次关系，可认为是一种网状结构模型。

此种类型数据库的优点是能很容易地反映实体之间的关联，同时还避免了数据的重复性；缺点是这种类型关联错综复杂，而且数据库很难对结构中所谓关联性进行维护。

3．关系型数据库

关系型数据库使用的存储结构是多个二维表格，即反映事物及其联系的数据描述是以平面表格形式体现的。

在关系模型中把实体集看成一个二维表，每一个二维表称为一个关系。在每个二维表中，每一行称为一条记录，用来描述一个对象的信息；每一列称为一个字段，用来描述对象的一个属性。数据表与数据库之间存在相应的关联，这些关联将用来查询相关的数据。例如，表 1-1 所示是一个学生关系。

表 1-1　学生关系

学号	姓名	性别	年龄	入学时间	专业	家庭住址
0001	王小强	男	18	2007-09-01	网络技术	孝义路 68 号
0002	林志生	男	19	2007-09-01	信息工程	桥东路 26 号
0003	张长芳	女	18	2007-09-01	会计电算化	瑞富路 43 号
…	…	…	…	…	…	…

（1）关系型数据库的内部结构。关系型数据库是由数据表之间的关联组成的。其中：

- 数据表通常是一个由行和列组成的二维表，每一个数据表分别说明数据库中某一特定的方面或部分的对象及其属性。
- 数据表中的行通常叫做记录或元组，它代表众多具有相同属性的对象中的一个。
- 数据表中的列通常叫做字段或属性，它代表相应数据库中存储对象的共有的属性。

（2）关系型数据库的基本原则。一个关系表必须符合某些特定条件，才能成为关系模型的一部分。

- 存储在单元中的数据必须是原始的，每个单元只能存储一条数据。
- 存储在列下的数据必须具有相同数据类型。
- 每行数据是唯一的。
- 列没有顺序。
- 行没有顺序。
- 列有一个唯一性的名称；
- 实体完整性原则（主键保证），不能为空；其中主键是能唯一标识行的一列或一组列的集合。

- 引用完整性原则（外键），不能为空。其中外键是一个表中的一列或一组列，它们在其他表中作为主键而存在。一个表中的外键被认为是对另一个表中主键的引用。

1.1.3　数据完整性

数据的完整性是保证数据的正确性、一致性和相容性。为了保证数据库的完整性，DBMS必须提供一种机制来检查数据库中的数据是否满足语义的要求，这种功能称为完整性检查。这些加在数据库数据上的语义约束条件称为数据库完整性约束条件。数据库完整性约束条件应该作为模式的一部分存入数据库中。

数据完整性分为四类：实体完整性（Entity Integrity）、域完整性（Domain Integrity）、参照完整性（Referential Integrity）、用户定义的完整性（User-defined Integrity）。

1．实体完整性（Entity Integrity）

实体完整性规定表的每一行在表中是唯一的实体。表中定义的 UNIQUE、PRIMARY KEY 和 IDENTITY 约束就是实体完整性的体现。

2．域完整性（Domain Integrity）

域完整性是指数据库表中的列必须满足某种特定的数据类型或约束。其中约束又包括取值范围、精度等规定。表中的 CHECK、FOREIGN KEY 约束和 DEFAULT、NOT NULL 定义都属于域完整性的范畴。

3．参照完整性（Referential Integrity）

参照完整性是指两个表的主关键字和外关键字的数据应对应一致。它确保了有主关键字的表中对应其他表的外关键字的行存在，即保证了表之间的数据的一致性，防止了数据丢失或无意义的数据在数据库中扩散。参照完整性是建立在外关键字和主关键字之间或外关键字和唯一性关键字之间的关系上的。在 SQL Server 中，参照完整性作用表现在如下几个方面：

- 禁止在从表中插入包含主表中不存在的关键字的数据行。
- 禁止会导致从表中的相应值孤立的主表中的外关键字值改变。
- 禁止删除在从表中有对应记录的主表记录。

4．用户定义的完整性（User-defined Integrity）

不同的关系数据库系统根据其应用环境的不同，往往还需要一些特殊的约束条件。用户定义的完整性即是针对某个特定关系数据库的约束条件，它反映某一具体应用所涉及的数据必须满足的语义要求。SQL Server 提供了定义和检验这类完整性的机制，以便用统一的系统方法来处理它们，而不是用应用程序来承担这一功能。其他的完整性类型都支持用户定义的完整性。

1.1.4　关系型数据库范式理论

关系数据库范式理论是在数据库设计过程中将要依据的准则，数据库结构必须要满足这

些准则，才能确保数据的准确性和可靠性。这些准则则被称为规范化形式，即范式。

在数据库设计过程中，对数据库进行检查和修改并使它返回范式的过程叫做规范化。下面是规范化的一个例子：

顾客名字	购买商品	商品价格
柳倩	鞋子	￥45
王永华	袜子	￥12
刘秀清	裤子	￥70

如果上面这个表用于保存物品的价格，而你想要删除其中一个顾客，这时你就必须同时删除一个价格。规范化就是要解决这个问题，你可以将这个表化为两个表，一个用于存储每个顾客和他所买物品的信息，另一个用于存储每件产品和其价格的信息，这样对其中一个表做添加或删除操作就不会影响另一个表。

范式按照规范化的级别分为 5 种：第一范式（1NF）、第二范式（2NF）、第三范式（3NF）、第四范式（4NF）和第五范式（5NF）。在实际的数据库设计过程中，通常需要用到的是前三类范式，下面将对它们分别介绍。

1．第一范式（1NF）

第一范式要求每一个数据项都不能拆分成两个或两个以上的数据项。

例如，如果关于员工的关系中有一个工资属性，而工资又由更基本的两个数据项：基本工资和岗位工资组成，则这个员工的关系模式就不满足 1NF。

满足第一范式是关系模式规范化的最低要求，否则，将有许多基本操作在这样的关系模式中实现不了，如上述的员工关系模式就实现不了按基本工资的 10%给每位员工加薪。

2．第二范式（2NF）

数据库表中不存在非关键字段对任一候选关键字段的部分函数依赖（部分函数依赖指的是存在组合关键字中的某些字段决定非关键字段的情况），也即所有非关键字段都完全依赖于任意一组候选关键字，那么该数据表满足第二范式。

这里为了便于描述，对于关系表将采用另一种表述方式。例如，对于如下名为 Books 的关系表：

书名	作者	出版社	ISBN 号	价格

其另一种表述为：

Books(书名, 作者, 出版社, ISBN 号, 价格)

要深刻理解第二范式，请看下面的案例。

假定选课关系表为 SelectCourse(学号, 姓名, 年龄, 课程名称, 成绩, 学分)，关键字为组合关键字(学号,课程名称)，则在这个选课关系表中存在如下决定关系：

(学号,课程名称) → (姓名,年龄,成绩,学分)

这个数据表能否满足第二范式呢？答案是：不满足。

在这个选课关系表中还事实存在如下决定关系：

(课程名称) → (学分)

(学号) → (姓名,年龄)

也就是在这个选课关系表中存在这样的情况：组合关键字中的字段决定非关键字，即是不符合第二范式了。

由于不符合第二范式，这个选课关系表会存在如下问题：

（1）数据冗余：同一门课程由 n 个学生选修，学分就重复 n-1 次；同一个学生选修了 m 门课程，姓名和年龄就重复了 m-1 次。

（2）更新异常：若调整了某门课程的学分，数据表中所有行的学分值都要更新，否则会出现同一门课程学分不同的情况。

（3）插入异常：假设要开设一门新的课程，暂时还没有人选修。这样，由于还没有学号关键字，课程名称和学分也无法记录入数据库。

（4）删除异常：假设一批学生已经完成课程的选修，这些选修记录就应该从数据库表中删除。但是，与此同时，课程名称和学分信息也被删除了。很显然，这也会导致插入异常。

把选课关系表 SelectCourse 改为如下三个表：

学生：Student(学号, 姓名, 年龄)；

课程：Course(课程名称, 学分)；

选课关系：SelectCourse(学号, 课程名称, 成绩)。

这样的数据库表是符合第二范式的，消除了数据冗余、更新异常、插入异常和删除异常。另外，所有单关键字的数据库表都符合第二范式，因为不可能存在组合关键字。

3．第三范式（3NF）

如果一个表已经满足第二范式，而且该数据表中的任何两个非主键字段的数值之间不存在函数依赖关系，那么该数据表满足第三范式。

例如，在某工资数据表中，“奖金”字段的数值是“工资”字段数值的 25%，因此，这两个字段之间存在着函数依赖关系，所以该数据表不满足第三范式。

又如，某学生关系表为 Student(学号,姓名,年龄,所在学院,学院地点,学院电话)，关键字为单一关键字“学号”，则该表有如下决定关系：

(学号) → (姓名,年龄,所在学院,学院地点,学院电话)

由于关键字为单一关键字，该数据库是符合第二范式的，但不符合第三范式，因为存在如下决定关系：

(学号) → (所在学院) → (学院地点, 学院电话)

即存在非关键字段“学院地点”、“学院电话”对关键字段“学号”的传递函数依赖。

把该学生关系表分为如下两个表：

学生：student(学号, 姓名, 年龄, 所在学院)；

学院：college(学院, 地点, 电话)。

这样的数据库表是符合第三范式的。

实际上，第三范式就是要求不要在数据库中存储可以通过简单计算得出的数据。不但可以节省存储空间，而且在拥有函数依赖的一方发生变动时，避免了修改成倍数据的麻烦，同时

也避免了在这种修改过程中可能造成的人为的错误。

从以上的叙述中可以看出，数据表规范化的程度越高，数据冗余就越少，而且造成人为错误的可能性就越小；同时，规范化的程度越高，在查询检索时需要做出的关联等工作就越多，数据库在操作过程中需要访问的数据库以及之间的关联也就越多。因此，在数据库设计的规范化过程中，要根据数据库需求的实际情况，选择一个折中的规范化程度。

1.1.5 分销系统数据库设计

数据库系统设计包括数据模型设计以及围绕数据模型的应用程序开发两大部分工作，这里主要介绍数据模型设计，也就是设计一组二维表，定义这些表的列名、列的数据类型以及表的数据完整性约束规则。

数据库设计过程一般包括以下步骤：需求分析、概念设计、逻辑设计、物理设计和实施维护。

1．分销系统数据库的需求分析

用户的需求具体体现了各种信息的提供、保存、更新和查询，这就要求数据库结构能充分满足各种信息的输出和输入。收集基本数据、数据结构以及数据处理的流程，组成一份详尽的数字字典，为后面的具体设计打下基础。

在分析调查了有关分销系统信息需求的基础上，将得到如图 1-1 所示的分销系统部分数据流图。

通过对分销系统的业务流程和数据流的分析，设计如下所示的数据项和数据结构。

客户资料：包括客户编号、客户名称、联系人、电话、传真、地址和备注等数据项目。

销售订单：包括销售订单号、日期、客户编号、客户名称、联系人、联系电话、送货地址、总金额等数据项目。

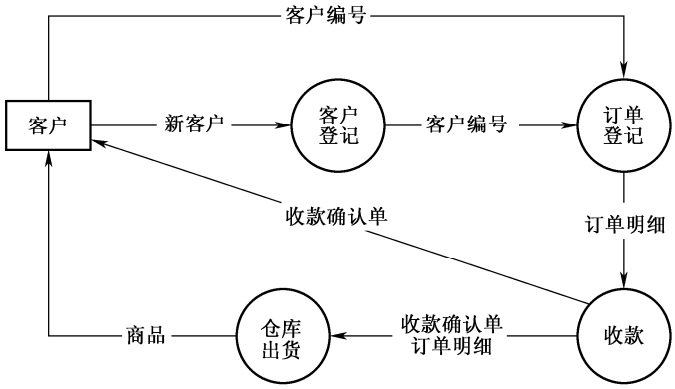

图 1-1　分销系统部分数据流图

销售订单明细：包括销售订单号、序号、商品编号、商品名称、规格型号、单位、数量、单价、金额等数据项目。

收款单：收款单号、收款日期、收款人、客户编号、客户名称、应收金额、收款金额和备注等数据项目。

出库单：出库单号、日期、客户编号、客户名称、联系人、联系电话、送货地址、总金额等数据项目。

出库明细：出库单号、序号、销售订单号、商品编号、规格型号、单位、数量、单价、金额等数据项目。

2．分销系统数据库的概念模型设计

根据上面的设计规划出的实体有客户实体、销售订单实体、销售订单明细实体、收款单实体、出库单实体、出库明细实体等。部分实体具体的描述 E-R 图如图 1-2、图 1-3、图 1-4 所示。

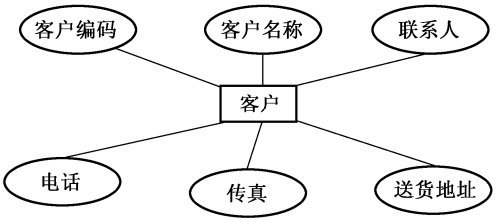

图 1-2　客户实体 E-R 图

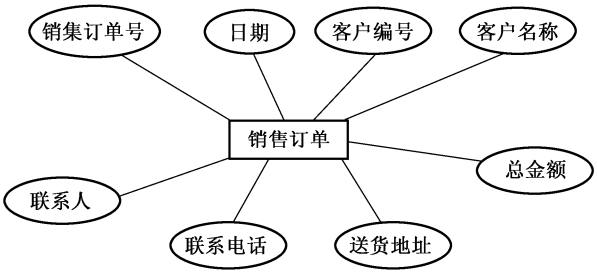

图 1-3　销售订单实体 E-R 图

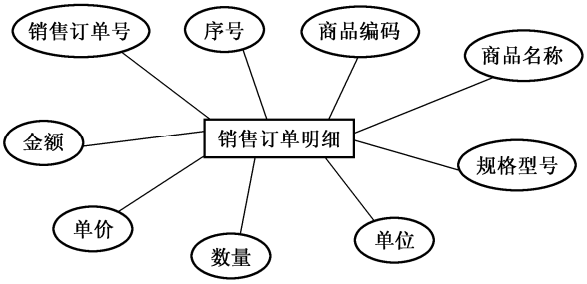

图 1-4　销售订单明细实体 E-R 图

对于其他实体 E-R 图，类似的很容易画出，实体与实体之间的关系也不难得出。

3．分销系统数据库的逻辑设计

现在需要把数据库概念结构转化为 SQL Server 数据库系统所支持的实际数据模型，也就是数据库的逻辑结构。

分析逻辑概念中的实体与实体之间的关系，再形成数据库中表与表之间关系。

分销系统的部分表设计结果如表 1-2、表 1-3、表 1-4、表 1-5 所示。每个表格表示在数据库中的一个表。

表 1-2　客户资料

字段名称	数据类型	是否允许为空	是否为主键	备注
客户编码	varchar(20)	否	是	
客户名称	varchar(100)	否	否	
联系人	varchar(20)	否	否	
电话	varchar(50)	否	否	
传真	varchar(50)	是	否	
地址	varchar(200)	是	否	
送货地址	varchar(200)	是	否	

表 1-3　销售订单

字段名称	数据类型	是否允许为空	是否为主键	备注
销售订单号	varchar(20)	否	是	
日期	datetime	否	否	
客户编码	varchar(20)	否	否	外键：客户资料（客户编码）
客户名称	varchar(100)	否	否	
联系人	varchar(20)	否	否	
联系电话	varchar(50)	否	否	
送货地址	varchar(200)	否	否	
总金额	numeric(12,4)	否	否	
备注	varchar(500)	是	否	

表 1-4　销售订单明细

字段名称	数据类型	是否允许为空	是否为主键	备注
销售订单号	varchar(20)	否	组合主键	外键：销售订单（销售订单号）
序号	int	否		
商品编码	varchar(20)	否	否	
商品名称	varchar(50)	否	否	
规格型号	varchar(100)	否	否	

续表

字段名称	数据类型	是否允许为空	是否为主键	备注
单位	numeric(12,4)	否	否	
数量	numeric(12,4)	否	否	
单价	numeric(12,4)	否	否	
金额	numeric(12,4)	否	否	
备注	varchar(500)	是	否	

表 1-5　收款单

字段名称	数据类型	是否允许为空	是否为主键	备注
收款单号	varchar(20)	否	是	
收款日期	datetime	否	否	
收款人	varchar(100)	否	否	
客户编码	varchar(20)	是	否	
客户名称	varchar(100)	是	否	
应收金额	numeric(12,2)	是	否	
收款金额	numeric(12,2)	否	否	
备注	varchar(500)	是	否	

1.2　分销系统数据库的创建

数据库主要是存储数据的表的集合以及其他的数据库对象。

SQL Server 可以同时支持许多数据库，每一个数据库既可以存储与另一个数据库相关的数据，也可以存储不相关的数据。

1.2.1　SQL Server 数据库基本知识

SQL Server 数据库中的数据在逻辑上被组织成一系列对象，当一个用户连接到数据库后，他所看到的是这些逻辑对象，而不是物理的数据库文件。

1．数据库对象

SQL Server 数据库中的数据在逻辑上被组织成一系列对象，当一个用户连接到数据库后，他所看到的是这些逻辑对象，而不是物理的数据库文件。

SQL Server 中有以下数据库对象：关系图、表、视图、存储过程、规则、用户自定义的数据类型、用户自定义函数等。

2．标识符命名规则

SQL Server 中，常规标识符命名规则如下：

（1）标识符长度是 1～128 个字符。

（2）第一个字符必须是大小写字母，以及来自其他语言的字母字符，也可以是汉字、下划线（_）、at 符号（@）或数字符号（＃）。标识符以单个字符@开始，表示为局部变量或参数，以两个字符@@开始，表示为全局变量；标识符以单个字符#开始，表示为临时数据库对象，以两个字符##开始，表示为全局临时数据库对象。

（3）后续字符可以是字母、基本拉丁字母、十进制数字、at 符号（@）、美元符号（$）、数字符号（＃）或下划线（_）。

（4）标识符不能是 Transact-SQL 语句的保留字。

（5）不允许嵌入空格或其他特殊字符。

3．数据库结构

（1）数据库文件。SQL Server 用文件来存放数据库，数据库中的所有数据和对象，如表、触发器和视图，都存储在下面这些操作系统文件中，数据库文件有三类。

- 主数据文件（Primary）：用来存放数据，该文件是数据库的起点，每个数据库都必须有且只有一个主数据文件。
- 次数据文件（Secondary）：也用来存放数据，这些次数据文件是可选的，它们可以存储那些不在主数据文件中的全部数据和对象。一个数据库可以没有也可以有多个 Secondary 文件。
- 事务日志文件（Transaction Log）：存放事务日志，这些日志文件保存了用于恢复数据库的全部事务日志信息。每个数据库都必须有一个或多个日志文件。

一般情况下，一个简单的数据库可以只有一个主数据文件和一个日志文件。如果数据库很大，则可以设置多个 Secondary 文件和日志文件，并将它们放在不同的磁盘上。

默认情况下，数据库文件存放在\MSSQL\DATA\目录下，数据文件名为“数据库名_Data.MDF”，日志文件名为“数据库名_Log.LDF”。数据库的创建者可以在创建时指定其他的路径和文件名，也可以添加 Secondary 文件和更多的日志文件。

（2）数据库文件组。文件组就是文件的集合。为了管理和数据分配的方便，允许多个数据库文件组成一个组，对它们整体进行管理。比如，可以将三个数据文件（data1.mdf、data2.mdf 和 data3.mdf）分别创建在三个盘上，这三个文件组成文件组 fgroup1，在创建表的时候就可以指定一个表创建在文件组 fgroup1 上。这样该表的数据就可以分布在三个盘上，在对该表执行查询时，可以并行操作，大大提高了查询效果。

（3）文件和文件组规则。SQL Server 的数据库文件和文件组必须遵循以下规则：

- 一个文件或文件组只能被一个数据库使用，不能用于多个数据库。
- 一个文件只能属于一个文件组。
- 一个数据库的数据信息和日志信息不能放在同一个文件或者文件组中，数据文件和日志文件总是分开的。
- 日志文件永远也不能是任何文件组的一部分，日志文件总是分开的。

（4）数据库空间管理。SQL Server 可管理的最小空间是以页为单位的，每一页的大小是8KB，即 8192 字节。在表中，每一行数据不能跨页存储。这样，表中每一行的字节数不能超过 8192 个字节。每 8 个连续页称为一个簇，即簇的大小是 64KB。每个表或者索引最少要占一个簇的空间，也就是说每一个表或者索引最小也是 64KB。

4．事务日志

每个数据库都有一个相关的事务日志，事务日志记录了 SQL Server 所有的事务和由这些事务引起的数据库的变化，即事务日志记录了对数据库的所有修改操作。在数据库中数据的任何改变写到磁盘之前，这个改变首先在事务日志中做了记录。

事务日志记录了每一个事务的开始、对数据的改变和取消修改的足够信息，随着对数据库的操作，日志是连续增加的。SQL Server 使用数据库的事务日志来恢复事务。所以事务日志具有以下三个作用。

（1）恢复单个事务。当执行 ROLLBACK 语句，或 SQL Server 发现错误时，回滚未完成的事务所做的修改。

（2）在 SQL Server 启动时恢复所有未完成的事务。当 SQL Server 出错停机时，事务的改变可能一部分被写入数据库文件，而另一部分还没有被写入，这样就造成了数据库中数据的不一致。事务日志可以使 SQL Server 重新启动时回滚所有未完成的事务，以保证数据库的一致状态。

（3）恢复数据库时，将数据库向前滚动到出错前一秒的状态。数据库从全库备份或差异备份恢复后，利用事务日志可以回滚所有未完成的事务，使数据库恢复到出错前一秒的状态。

5．系统数据库

打开 SQL Server Management Studio，在创建任何数据库之前，展开对象资源管理器的“数据库”节点，可以看到已经有了“系统数据库”节点，该节点下有四个数据库。它们是 SQL Server 的系统数据库。与用户数据库不同，系统数据库是在安装 SQL Server 时由安装程序自动创建的。

SQL Server 有四个系统数据库，分别是 master 数据库、tempdb 数据库、model 数据库和 msdb 数据库。下面介绍系统数据库。

（1）master 数据库。记录了 SQL Server 系统级的信息，包括系统中所有的登录账户、系统配置信息、所有数据库信息、所有用户数据库的主文件地址等，这些信息都记录在 master 数据库的表中，为了与用户创建的表相区别称为系统表，表名都以“sys”开头。

master 数据库中还有很多系统存储过程和扩展存储过程。

（2）tempdb 数据库。用于存放所有连接到系统的用户的临时表和临时存储过程，以及 SQL Server 产生的其他临时性的对象。tempdb 是 SQL Server 中负担最重的数据库，因为几乎所有的查询都可能需要使用它。

在 SQL Server 关闭时 tempdb 数据库中的所有对象都被删除，每次启动 SQL Server 时会重新创建 tempdb 数据库。

tempdb 数据库可以按照需要自动增长，每次系统启动时，tempdb 数据库都被重置为默认的大小（8.0MB）。在以后的工作中，当 tempdb 数据库空间不够时，系统都将自动扩展 tempdb

数据库的大小。

（3）model 数据库。model 数据库是系统所有数据库的模板，这个数据库相当于一个模子，所有在系统中创建的新数据库的内容，在刚创建时都和 model 数据库完全一样。

刚刚完成 SQL Server 安装时，model 数据库就已经有了 18 个表以及一些视图和存储过程，因此用户创建的每个数据库中都将有这些对象。这 18 个表是另一类的系统表，它们的表名也是以“sys”开头的，其内容是有关数据库的结构等重要信息。

（4）msdb 数据库。SQL Server 代理（SQL Server Agent）使用 msdb 数据库来安排报警、作业，并由操作员记录。

6．估算数据库的空间需求

作为数据库管理员，主要任务之一就是创建数据库，并且需要为每个文件指定容量。必须尽可能准确地估算数据库容量，以免浪费磁盘空间资源或者因估计不足造成数据库的空间不够。

许多因素会影响数据库最终的大小，在估算数据库容量时要考虑如下因素：

- 每行记录的大小。
- 记录数量。
- 表的数量。
- 索引的数量及索引大小。
- 数据库的对象的数量和大小。
- 事务日志的大小。
- 数据库的计划增加量。

7．确定数据库的数目

当一个企业或部门确定要建立数据库系统之后，接着就要确定这个数据库系统与企业中其他部分的关系。因此，需要分析企业的基本业务功能，确定数据库支持的业务范围，是建立一个综合的数据库，还是建立若干个专门的数据库。

从理论上讲，我们可以建立一个支持企业全部活动的包罗万象的大型综合数据库，也可以建立若干个支持范围不同的公用或专用数据库。

一般来讲，前者难度较大，效率也不高；后者比较分散，但相对灵巧，必要时可通过联接操作将有关数据联接起来，而数据的全局共享一般可利用建立在数据库上的应用系统来实现。在各种规模的企业当中，同时在不同部门内维护众多中小型数据库已成为一种常见现象。通常情况下，这些数据库或服务器被置于 IT 部门负责日常维护管理的系统范畴之外，因此，它们无法达到企业在设计、实现及维护方面所制订的 IT 标准。

1.2.2 使用 SQL Server Management Studio 创建数据库

使用数据库存储数据，首先要创建数据库。前面已介绍过，一个数据库必须至少包含一个数据文件、一个事务日志文件。所以创建数据库就是创建主数据库文件和事务日志文件。在 SQL Server 中，可以使用 SQL Server Management Studio 创建数据库，也可以使用 Transact-SQL

创建数据库。

任务 1-1：使用 SQL Server Management Studio 创建分销系统数据库。

详细步骤如下：

（1）启动 SQL Server Management Studio，登录服务器类型为“数据库引擎”，并使用 Windows 或 SQL Server 身份验证建立连接。

（2）连接成功后，在对象资源管理器中，右击“数据库”节点，从弹出的快捷菜单中选择“新建数据库”命令，打开“新建数据库”窗口，如图 1-5 所示。

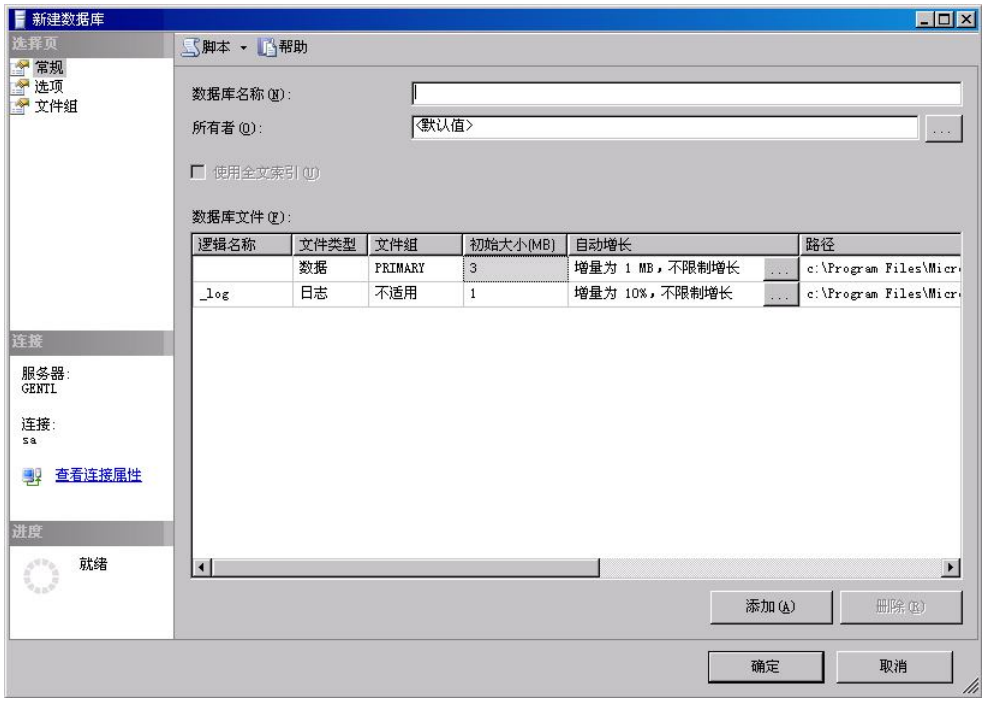

图 1-5　新建数据库窗口

（3）默认显示的是“常规”选项卡。首先，在“数据库名称”文本框中输入数据库名称“分销系统”。再输入数据库所有者，用户可以使用“默认值”，也可应通过单击文本框右边的“浏览”按钮，选择所有者。

（4）在下面的“数据库文件”列表中，列出了数据库包含的主数据文件和事务日志文件的逻辑名称、文件类型、文件组、初始大小、自动增长值和路径等。用户可以自行设置初始大小、自动增长值和数据库存放的位置。

（5）在“选项”选项卡中，可以定义所创建数据库的排序规则、恢复模式、兼容级别、恢复、游标等其他杂项。

（6）在“文件组”选项卡中，可以查看数据库中的所有文件组，包括主文件组和次文件组。还可以通过单击“添加”或“删除”按钮来添加或删除次文件组。

（7）全部设置完毕后，单击“确定”按钮即可完成分销系统数据库的创建。在对象资源管理器中，右击“数据库”节点，从弹出的快捷菜单中选择“刷新”命令。展开“数据库”节点，可看到创建好的“分销系统”数据库在列表当中。

1.2.3 使用 Transact-SQL 创建数据库

在 SQL Server 中，可以使用 Create Database 语句来创建数据库。该语句语法格式如下：

```
CREATE DATABASE database_name
    [ON
    {[PRIMARY](NAME = logical_file_name,
    FILENAME = 'os_file_name'
    [,SIZE = size]
    [,MAXSIZE = max_size]
    [,FILEGROWTH = groth_increment])
    }[,…n]
    ]
FILEGROUP filegroup_name < filespec > [ ,...n ]
[LOG ON
    { (NAME = logical_file_name,
    FILENAME = 'os_file_name'
    [,SIZE = size]
    [,MAXSIZE = max_size]
    [,FILEGROWTH = growth_increment])
    }[,…n]
    ]
    [FOR RESTORE]
```

下面详细介绍一些选项的含义：

（1）PRIMARY：该选项是一个关键字，用来指定主文件组中的主文件。

主文件组中不仅包含了数据库系统表中的全部内容，而且还包含了没有在用户文件组中包含的全部对象。一个数据库只能有一个主文件。在默认情况下，即在没有指定PRIMARY关键字时，列在语句中的第一个文件就是主文件。

（2）NAME：该选项用来指定数据库的逻辑名称。

这是在 SQL Server 系统中使用的名称，是数据库在 SQL Server 中的标识符。

（3）FILENAME：该选项用来指定数据库所在文件的操作系统文件名称和路径。

在 os_file_name 中的路径必须是 SQL Server 所在服务器上的一个文件夹。该操作系统文件名与 NAME 的逻辑名称是一一对应的。

（4）SIZE：该选项用来指定数据库操作系统文件的大小。

在指定文件大小的时候既可以用 MB 作单位，也可以使用 KB 作单位。如果没有指定单位，那么系统默认的单位是 MB。文件最小是 1MB，也就是说，数据库所在的文件不能小于 1MB。在默认情况下，数据库数据文件的大小是 1MB，数据库日志文件的大小也是 1MB。

（5）MAXSIZE：该选项用来指定操作系统文件可以增长的最大尺寸。

在指定文件增长尺寸的时候，既可以使用 MB 作单位，也可以使用 KB 作单位。如果没有指定单位，那么系统默认的单位是 MB。如果没有指定文件可以增长的最大尺寸，那么系统的增长是没有限制的，可以占满整个磁盘空间。

（6）FILEGROWTH：该选项用来指定文件的增量。

当然该选项不能与 MAXSIZE 选项有冲突。该选项指定的数据值为零时，表示文件不能增长。该选项可以用 MB、KB 和百分比指定。

任务 1-2：创建一个 student 数据库。

CREATE　DATABASE　student

任务 1-3：创建一个 CUSTOMER 数据库，该数据库的主数据文件的逻辑名称是 CUSTOMER_DATA，操作系统文件是 CUSTOMER_DATA.MDF，大小是 15MB，最大是 30MB，以 20%的速度增加；该数据库的日志文件的逻辑名称是 CUSTOMER_LOG，操作系统文件是 CUSTOMER_LOG.LDF，大小是 3MB，最大是 10MB，以 1MB 的速度增加。

```
CREATE DATABASE  customer
ON
    PRIMARY (NAME = customer_data,
    FILENAME='e:\yxl\customer_data.mdf',
SIZE = 15MB,
    MAXSIZE = 30MB,
    FILEGROWTH=20%)
LOG ON
   (NAME = customer_log,
    FILENAME = 'e:\yxl\customer_log.ldf',
     SIZE = 3MB,
    MAXSIZE = 10MB,
    FILEGROWTH = 1MB)
```