目　录

第 1 章　数据库基础

随着社会信息化进程的加快，以数据库系统为核心的办公自动化系统、信息管理系统、决策支持系统等得到了广泛应用，作为计算机应用人员，只有掌握数据库系统的基础知识，熟悉数据库管理系统的特点，才能开发出适用的、水平较高的数据库应用系统，为社会各行各业的信息化建设添砖加瓦。

1-1　数据库基本概念

1-1-1　数据处理

数据（Data）是对客观事物的某些特征及其相互联系的一种抽象化、符号化表示。例如：李明出生日期为 1963 年 9 月 17 日，身高 1.75m，体重 65kg，部门代码 A01，职称是副教授，其中李明、1963 年 9 月 17 日、1.75m、65kg、A01、副教授等都是数据，它们描述了该人的某些特征。

从上述可知，数据不仅包括数字、字母、文字及其他特殊字符组成的文本形式的数据，而且还可包括图形、图像和声音等多媒体数据。

现实世界中的数据往往是原始的、非规范化的，但它是数据的原始集合，通过对这些原始数据的处理，才能产生新的数据（信息）。这一处理包括对数据的收集、记录、分类、排序、存储、计算/加工、传输、制表和递交等操作，这就是数据处理的概念。

经过处理的数据是精炼的数据，能够反映事物或现象的本质和特征及其内在联系。人类社会处理数据的发展过程可以分为如下 3 个阶段。

（1）手工处理阶段：这一阶段使用简单的手工工具（例如，算盘等），处理效率低，能处理的数量少而且可靠性差。

（2）机械处理阶段：这一阶段的主要特征是使用了比第一阶段先进得多且比较有效的工具（例如，19 世纪 80 年代发明的卡片制表机等），处理效率高，且可靠性也有较大的提高。

（3）电子处理阶段：这一阶段使用电子计算机进行数据处理，为数据处理展现了广阔的前景。计算机不仅处理速度快、存储容量大、输入/输出灵活，而且可以把人的手工操作降低到最小程度。电子数据处理方法不仅适应了不断提高的社会生产力的需要，而且给社会生产力的发展以推动和促进。

1-1-2　数据模型

现实世界中客观存在并且相互区别的事物称为实体。实体可以是具体的人、事、物（例如：一名教师，一个部门，或一台计算机等），也可以是抽象的概念或事件（例如：讲一门课，给某单位供应一台设备等）。

同类型实体的集合称为实体集。例如，学校全体教职工构成一个学校的教职工实体集，

学校全体学生构成一个学校的学生实体集。

实体的特性称为属性，属性是实体之间相互区别的标志，一个实体可以由若干个属性来刻画。例如，教职工实体可以用职工编号、姓名、性别、出生日期和职称等属性来描述。

1. 实体联系

实体之间的对应关系称为联系，它反映了现实世界各个事物之间的相互关系。实体之间的联系有以下 3 种类型。

（1）一对一联系（1:1）：如果对于实体集 A 中的每一个实体，实体集 B 中至多有一个（也可以没有）实体与之联系，反之亦然，则称实体集 A 与实体集 B 具有一对一联系，记为 1:1。

例如，一个部门只有一个正职负责人，而一个正职负责人只在一个部门中任职，则部门与正职负责人之间具有一对一联系，联系名设定为“领导”。

（2）一对多联系（1:n）：如果对于实体集 A 中的每一个实体，实体集 B 中有 n 个实体（n≥1）与之联系，反之，对于实体集 B 中的每一个实体，实体集 A 中至多只有一个实体与之联系，则称实体集 A 与实体集 B 有一对多联系，记为 1:n。

例如，一个部门中有若干名职员，而每个职员只在一个部门中任职，则部门与职员之间具有一对多联系，联系名设定为“任职”。

（3）多对多联系（m:n）：如果对于实体集 A 中的每一个实体，实体集 B 中有 n 个实体（n≥1）与之联系，反之，对于实体集 B 中的每一个实体，实体集 A 中也有 m 个实体（m≥1）与之联系，则称实体集 A 与实体集 B 具有多对多联系，记为 m:n。

例如，一项工作同时有若干个职员参与，而一个职员可以同时参与多项工作，则工作与职员之间具有多对多联系，联系名设定为“参与”。

实际上，一对一联系是一对多联系的特例，而一对多联系又是多对多联系的特例。可以用 E-R 图（实体-关系模型）来表示两个实体集之间的这 3 类联系，如图 1-1 所示。

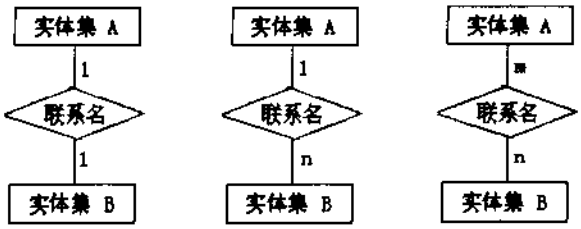

图 1-1　两个实体集之间的 3 类联系

2. 数据模型

模型是现实世界特征的模拟和抽象，而数据模型（Data Model）是现实世界数据特征的抽象。在数据库中用数据模型来抽象、表示和处理现实世界中的数据和信息。

数据模型应满足 3 方面要求：

① 能比较真实地模拟现实世界。

② 容易为人所理解。

③ 便于在计算机上实现。

不同的数据模型实际上是提供模型化数据和信息的不同工具。模据模型应用的不同目的，可以将这些模型划分为两类，分属于两个不同的层次。

第一类：概念模型，也称信息模型，按用户的观点来对数据和信息建模，主要用于数据库设计。

第二类：数据模型，主要包括网状模型、层次模型、关系模型等，它是按计算机系统的观点对数据建模，主要用于 DBMS 的实现。

数据模型的 3 个类别分别介绍如下：

（1）层次模型

层次模型用树形结构来表示各类实体以及实体间的联系。满足下面两个条件的基本层次联系的集合为层次模型。

① 有且只有一个结点没有双亲结点，这个结点称为根结点。

② 根以外的其他结点有且只有一个双亲结点。

在层次模型中，每个结点表示一个实体集，实体集之间的联系用结点之间的连线（有向边）表示，这种联系是父子之间的一对多的联系。

在层次模型中，同一双亲的子女结点称为兄弟结点（Twin 或 Sibling），没有子女结点的结点称为叶结点。

图 1-2 给出了一个层次模型的例子。

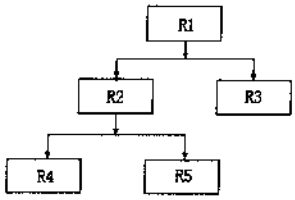

图 1-2　层次模型

其中 R1 为根结点；R2 和 R3 为兄弟结点，是 R1 的子女结点；R4 和 R5 为兄弟结点，是 R2 的子女结点；R3、R4 和 R5 为叶结点。

（2）网状模型

网状模型用网形结构来表示各类实体以及实体间的联系。把满足以下两个条件的基本层次联系集合称为网状模型：

① 允许一个以上的结点无双亲。

② 一个结点可以有多于一个的双亲。

网状模型去掉了层次模型的两个限制，允许多个结点没有双亲结点，允许一个结点有多个双亲结点，此外它还允许两个结点之间有多种联系（称之为复合联系）。

网状模型中每个结点表示一个实体集，结点间的连线表示实体集之间一对多的父子联系。

在网状模型中，子女结点与双亲结点的联系可以不惟一。因此，要为每个联系命名，并指出与该联系有关的双亲记录和子女记录。

例如，图 1-3 中 R3 有两个双亲记录 R1 和 R2，因此把 R1 与 R3 之间的联系命名为 L2，R2 与 R3 之间的联系命名为 L3。

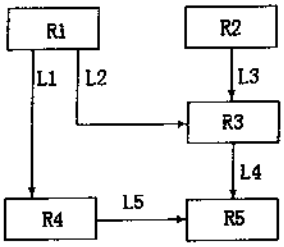

图 1-3　网状模型

（3）关系模型

在用户观点下，关系模型中数据的逻辑结构是一张二维表，它由行和列组成。

以学生登记表（如表 1-1 所示）为例，介绍关系模型中的一些术语。

表 1-1　学生登记表

学号	姓名	性别	出生日期	系名	籍贯
2004020001	李小龙	男	1984.6.21	计算机系	广东省
2004020002	秦英	女	1983.7.28	计算机系	河北省
2004040001	刘志坚	男	1983.1.3	电子系	广东省
……	……	……	……	……	……

关系（Relation）：一个关系对应通常说的是一张二维表。例如，表 1-1 中的这张学生登记表就是一个关系，可以命名为学生关系。

元组（Tuple）：表中的一行即为一个元组。例如，表 1-1 有 3 行，对应 3 个元组。

属性（Attribute）：表中的一列即为一个属性，给每一个属性起一个名称即属性名。例如，表 1-1 中有 6 列，对应的 6 个属性分别为学号、姓名、性别、出生日期、系名和籍贯。

主码（Key）：主码是表中的某个属性组，它可以惟一确定一个元组。例如，表 1-1 中的学号可以惟一确定一个学生，也就可以作为本关系的主码。

域（Domain）：属性的取值范围。例如，性别的域是（男，女），系名的域是一个学校所有系的名称集合。

分量：元组中的一个属性值。例如，表 1-1 中的第一个元组在学号属性上的取值为 2004020001，则 2004020001 就是集一个元组的一个分量。

关系模式：对关系的模述，一般表示为：

关系名（属性 1，属性 2，…，属性 n）

例如，表 1-1 的学生关系可描述为：

学生（学号，姓名，性别，出生日期，系名，籍贯）

在关系模型中，实体以及实体间的联系都是用关系来表示的。

1-1-3　数据库系统

数据库技术是 20 世纪 60 年代末兴起的一种数据管理技术。数据库（DataBase，DB）可直译为存储数据的基地。形式化的定义为：数据库是指存储在计算机外存设备或网络存储设备上的、结构化的相关数据集合。

1. 数据库系统组成

通常将引进数据库技术的计算机系统称为数据库系统（DataBase System，DBS）。一般来说，数据库系统由以下几部分组成。

（1）计算机硬件系统：用来运行操作系统、数据库管理系统、应用程序以及存储数据库的本地计算机系统和网络硬件环境。

（2）数据库集合：存储在本地计算机外存设备或网络存储设备上的若干个设计合理、满足应用需要的数据库。

（3）数据库管理系统：数据库管理系统（DataBase Management System，DBMS）是数据库系统的核心，用于协助用户创建、维护和使用数据库的系统软件。

（4）相关软件：包括操作系统、编译系统、应用开发工具软件和计算机网络软件等。

（5）人员：包括数据库管理员和用户。数据库管理员负责数据库系统的建立、维护和管理。用户可分为专业用户和最终用户，专业用户侧重于设计数据库和开发应用程序，最终用户则侧重于数据库的使用。

2. 数据库系统特点

在数据库技术出现之前，计算机使用文件系统来存储、管理数据，与文件系统比较，数据库系统有如下特点：

（1）数据结构化：在文件系统中，各个文件不存在相互联系。数据库系统则不同，在同一数据库中的数据文件是有联系的，且在整体上服从一定的结构形式。

（2）数据共享：在文件系统中，数据一般是由特定的用户专用。而数据库中的数据不仅可为同一企业或结构之内的各个部门所共享，也可为不同单位、地域甚至不同国家的用户所共享。

（3）数据独立：在文件系统中，数据结构和应用程序相互依赖、相互影响。数据库系统则力求减少这种依赖，实现数据的独立性。

（4）冗余度可控：文件系统中数据专用，每个用户拥有和使用自己的数据，造成许多数据重复，这就是数据冗余。在数据库系统中实现共享后，不必要的重复将删除，但为了提高查询效率，有时也保留少量重复数据，其冗余度可由设计人员控制。

（5）数据统一控制：为保证多个用户能同时正确地使用同一个数据库，数据库系统提供以下数据控制功能：

① 安全性控制：数据库设置一套安全保护措施，保证只有合法用户才能进行指定权限的操作，防止非法使用所造成的数据泄密和破坏。

② 完整性控制：数据库系统提供必要措施来保证数据的正确性、有效性和相容性，当计算机系统出现故障时，提供将数据恢复到正确状态的相应机制。

③ 并发控制：数据库对多用户的并发操作予以控制和协调，保证多个用户的操作不相互干扰。

3. 数据库管理系统

数据库管理系统是数据库系统的核心，其需要解决如下两个问题：

（1）科学地组织和存储数据。

（2）高效地获取和维护数据。

数据库管理系统是位于用户与操作系统之间的一层数据管理软件。它的主要功能包括以下几个方面：

（1）数据定义功能：DBMS 提供数据定义语言（Data Definition Language，DDL），用户通过它可以方便地对数据库中的数据对象进行定义。

（2）数据操作功能：DBMS 提供数据操作语言（Data Manipulation Language，DML），用户可以使用 DML 操作数据实现对数据库的基本操作，如查询、插入、删除和修改等。

（3）数据库的运行管理功能：数据库在建立、运用和维护时由数据库管理系统统一管理、统一控制，以保证数据的安全性、完整性、多用户对数据的并发使用及发生故障后的系统恢复。

（4）数据库的建立和维护功能：包括数据库初始数据的输入、转换功能，数据库的转储、恢复功能，数据库的重组织功能和性能监视、分析功能等。

1-2 关系数据库

1-2-1 关系数据结构定义

关系模型是建立在集合代数的基础上的，下面从集合论角度给出关系数据结构的形式化定义。

1. 域（ Domain）

定义 1-1 域是一组具有相同数据类型值的集合。

例如，自然数、整数、实数、长度小于 25 的字符串集合、{0，1}、大于等于 0 且小于等于 100 的正整数等，都可以是域。

通常用字母 D 来表示域。

2. 笛卡尔积（Cartesian Product）

定义 1-2 给定一组域 D_1，D_2，…，D_n，这些域中可以有相同的。D_1，D_2，…，D_n 的笛卡尔积为：

$$D_1 \times D_2 \times \cdots \times D_n = \{(d_1, d_2, \cdots, d_n) \mid d_i \in D_i, i=1, 2, \cdots, n\}$$

其中，每一个元素（d_1，d_2，…，d_n）叫作一个 n 元组（n-tuple）或简称元组（Tuple），元素中的每一个值 d_i 叫作一个分量（Component）。符号“∈”的含义为“属于”，$d_i \in D_i$ 表示 d_i 是 D_i 集合中的一个元素。

若 $D_i(i=1, 2, \cdots, n)$为有限集，其基数（Cardinal Number）为 $m_i(i=1, 2, \cdots, n)$，则 $D_1 \times D_2 \times \cdots \times D_n$ 的基数 M（元素个数）为：

$M=\prod m_i(i=1, 2, \cdots, n)$

例如，$D_1=\{a,b,c\}$, $D_2=\{A,B\}$, $D_3=\{0,1\}$

则 D_1，D_2，D_3 笛卡尔积的元素个数为 $3\times2\times2=12$，组成如下：

$D_1 \times D_2 \times D_3=\{$（a,A,0）, (a,A,1), (a,B,0),(a,B,1)，（b,A,0）, (b,A,1),
(b,B,0),(b,B,1),(c,A,0),(c,A,1),(c,B,0),(c,B,1)$\}$

3．关系（Relation）

定义 1-3　$D_1 \times D_2 \times \cdots \times D_n$ 的子集叫作在域 D_1，D_2，…，D_n 上的关系，记为：

$R(D_1, D_2, \cdots, D_n)$

这里 R 为关系的名称，n 是关系的目或度(Degree)。当 n=1 时，称该关系为单元关系(Unary relation)。当 n=2 时，则称该关系为二元关系（Binary relation)。关系中的每个元素都是关系中的元组，通常用字母 t 表示。

关系是笛卡尔积的有限子集，所以关系也是一个二维表，表的每行对应一个元组，表的每列对应一个域。由于域可以相同，为了加以区分，必须对每列起一个名字，称为属性（Attribute)。n 目关系必有 n 个属性。

若关系中的某一属性组的值能惟一地标识一个元组，则称该属性组为候选码（Candidate key)。若一个关系有多个候选码，则可选定其中一个为主码（Primary key)。主码的诸属性称为主属性（Prime attribute)。不包含在任何候选码中的属性称为非码属性（Non-key attribute)。在最简单的情况下，候选码只包含一个属性。在最极端的情况下，关系模式的所有属性的组合是这个关系模式的候选码，称为全码（All-key)。

例如，表 1-1 中学号={2004020001, 2004020002, 2004040001}，姓名={李小龙, 秦英, 刘志坚}，性别={男, 女}，系名={计算机系, 电子系}，籍贯={广东省, 河北省}，则学号、姓名、性别、系名和籍贯的笛卡尔积（学号×姓名×性别×系名×籍贯）有 72 个元素，其中只有 3 个元素是有意义的，这 3 个元素就构成了学生(学号, 姓名, 性别, 出生日期, 系名, 籍贯)关系的 3 个元组，属性组{学号}能惟一地标识一个元组，则{学号}为候选码，可以选定其为主码，而学号是主属性。

1-2-2　关系运算

任何一种运算都是将一定的运算符作用在一定的运算对象上，得到预期的运算结果。所以运算对象、运算符、运算结果是运算的 3 大要素。关系运算的运算对象是关系，运算结果亦为关系。

关系代数用到的运算符包括 4 类：集合运算符、专门的关系运算符、比较运算符和逻辑运算符，如表 1-2 所示。

表 1-2　关系运算符

运算符		含义	运算符		含义
集　合	∪	并	z	>	大于
运算符	−	差		≥	大于等于
	∩	交		<	小于
				≤	小于等于
				=	等于
				≠	不等于

续上表

运算符		含义	运算符		含义
专门关系运算符	σ	选择	逻辑运算符	¬	非
	π	投影		∧	与
		连接		∨	或
	÷	除			
	×	广义笛卡尔积			

集合运算将关系看成元组的集合，其运算是从关系的“水平”方向即行的角度来进行，下面介绍关系的并、差和交运算。专门的关系运算不仅涉及行而且涉及列，本书仅介绍常用的选择、投影和连接运算。比较运算和逻辑运算是用来辅助专门的关系运算的，含义与数学中的意义类似，这里不再赘述。

1. 集合运算

并、差、交运算是二目运算。设关系 R 和关系 S 具有相同的目 n（即两个关系都有 n 个属性），且相应的属性取自同一个域，则可以定义并、差、交运算如下。

（1）并（Union）

关系 R 与关系 S 的并记作：

$$R \cup S = \{t \mid t \in R \vee t \in S\}$$

其结果仍为 n 目关系，由属于 R 或属于 S 的元组组成。

（2）差（Difference）

关系 R 与关系 S 的差记作：

$$R - S = \{t \mid t \in R \wedge t \neg \in S\}$$

其结果关系仍为 n 目关系，由属于 R 而不属于 S 的所有元组组成。符号“¬∈”的含义为“不属于”，t¬∈S 表示 t 不是关系 S 中的一个元组。

（3）交（Intersection）

关系 R 与关系 S 的交记作：

$$R \cap S = \{t \mid t \in R \wedge t \in S\}$$

其结果关系仍为 n 目关系，由既属于 R 又属于 S 的元组组成。

表 1-3 中 R、S 为具有相同属性（A、B、C）的关系，R∪S、R-S 和 R∩S 为 3 种集合运算结果。

表 1-3 集合运算例

R			S			R∪S			R-S			R∩S		
A	B	C	A	B	C	A	B	C	A	B	C	A	B	C
a1	b1	c1	a1	b2	c2	a1	b1	c1	a1	b1	c1	a1	b2	c2
a1	b2	c2	a1	b3	c2	a1	b2	c2				a2	b2	c1
a2	b2	c1	a2	b2	c1	a2	b2	c1						
						a1	b3	c2						

2. 专门的关系运算

引入如下几个记号：

① 设关系为 $R(A_1, A_2, \cdots, A_n)$。$t \in R$ 表示 t 是 R 的一个元组。$t[A_i]$ 则表示元组 t 中对应于属性 A_i 的一个分量。

② 若 $A=\{A_{i1}, A_{i2}, \cdots, A_{ik}\}$，其中 $A_{i1}, A_{i2}, \cdots, A_{ik}$ 是 $A_1, A_2, \cdots, A_n$ 中的一部分，A 称为属性列或域列。$t[A]=(t[A_{i1}], t[A_{i2}] \cdots, t[A_{ik}])$ 表示元组 t 在属性列 A 上诸分量的集合。

③ R 为 n 目关系，S 为 m 目关系。$t_r \in R$，$t_s \in S$，$t_r t_s$ 称为元组的连接（Concatenation）。它是一个 n+m′列的元组，前 n 个分量为 R 中的一个 n 元组，后 m′个分量为 S 中的 m 元组去掉与 R 中重复的列分量后的一个 m′元组。

下面给出选择、投影和连接关系运算的定义。

（1）选择（Selection）

选择又称为限制（Restriction）。它是在关系 R 中选择满足给定条件的诸元组，记作：

$$\sigma_F(R)=\{t \mid t \in R \wedge F(t)=\text{"真"}\}$$

其中：F 表示选择条件，它是一个逻辑表达式，取逻辑值“真”或“假”。

逻辑表达式 F 由逻辑运算符¬、∧、∨连接各算术表达式组成。算术表达式的基本形式为：

$$X_1 \ \theta \ Y_1$$

其中：θ 表示比较运算符，它可以是＞、≥、＜、≤、=或≠；

X_1、Y_1 等是属性名，或为常量，或为简单函数；属性名也可以用它的序号来代替。

选择运算实际上是从关系 R 中选取使逻辑表达式为真的元组。这是从行的角度进行的运算。

（2）投影（Projection）

关系 R 上的投影是从 R 中选择出若干属性列组成新的关系。记作：

$$\pi_A(R)=\{t[A] \mid t \in R\}$$

其中：A 为 R 中属性列的集合。

投影操作是从列的角度进行的运算。

（3）连接（Join）

一般来说，进行连接运算的两个关系必须具有相同的属性列，并且根据相同的属性列的取值是否相等来选择构成结果关系的元组，在结果中把重复的属性列去掉。设 R 和 S 具有相同的属性组 B，则连接运算可记作：

$$R \triangleright\!\triangleleft S=\{t_r t_s \mid t_r \in R \wedge t_s \in S \wedge t_r[B]=t_s[B]\}$$

连接操作不仅从行的角度进行运算，而且还需要取消重复列，所以是同时从行和列的角度进行运算。

表 1-4 中 R、S 为关系，$\sigma_{A=a1}(R)$、$\pi_{B,C}(R)$和 $R \triangleright\!\triangleleft S$ 为 3 种集合运算结果。

表 1-4 专门关系运算例

R			S		$\sigma_{A=a1}(R)$			$\pi_{B,C}(R)$		R S			
A	B	C	B	E	A	B	C	B	C	A	B	C	E
a1	b1	5	b1	3	a1	b1	5	b1	5	a1	b1	5	3
a1	b2	6	b2	7	a1	b2	6	b2	6	a1	b2	6	7
a2	b3	8	b3	10				b3	8	a2	b3	8	10
a2	b4	12	b3	2				b4	12	a2	b3	8	2
			b5	2									

1-2-3 关系数据库

在关系数据库中，关系模式是型，关系是值。关系模式是对关系的描述，那么一个关系需要描述哪些方面呢？

关系实质上是一张二维表，表的每一行为一个元组，每一列为一个属性。一个元组就是该关系所涉及的属性集的笛卡尔积的一个元素。关系是元组的集合，因此关系模式必须指出这个元组集合的结构，即它由哪些属性构成，这些属性来自哪些域，以及属性与域之间的映象关系。

其次，一个关系通常是由赋予它的元组语义来确定的。元组语义实质上是一个 n 目谓词（n 是属性集中属性的个数）。凡使该 n 目谓词为真的笛卡尔积中的元素的全体就构成了该关系模式的一个关系。

关系模式是静态的、稳定的，而关系是动态的、随时间不断变化的，因为关系操作在不断地更新着数据库中的数据。

在一个给定的应用领域中，所有实体及实体之间联系的关系的集合构成一个关系数据库。关系数据库也有型和值之分。关系数据库的型也称为关系数据库模式，是对关系数据库的描述，它包括若干域的定义以及在这些域上定义的若干关系模式。关系数据库的值是这些关系模式在某一时刻对应的关系的集合，通常就称为关系数据库。

思考题

1. 什么是数据？什么是数据处理？
2. 实体之间联系有哪 3 种类型？举例说明。
3. 共有哪 3 种数据模型？各有什么特点？
4. 数据库系统由哪几部分组成？DBS 和 DBMS 什么关系？
5. 简述数据库管理系统的主要功能。

第 2 章　Visual FoxPro 系统初步

Visual FoxPro 是 dBASE 数据库家族的最新成员，也是其前身 FoxPro 与可视化程序设计技术相结合的产物。本章将首先介绍 Visual FoxPro 发展历史、特点、界面及工作方式，然后对 Visual FoxPro 辅助设计工具和项目管理器等知识作概要介绍，为学习其余各章节打下良好的基础。

2-1　Visual FoxPro 关系数据库系统

2-1-1　Visual FoxPro 发展历史

Visual FoxPro 的历史可追溯到 20 世纪 80 年代初，其前身是 dBASE 数据库管理系统。1981 年美国 Ashon-Tate 公司推出了微机上能够使用的关系型数据库管理系统 dBASE Ⅱ，由于它功能丰富、简单易学、使用方便而受到广大用户欢迎，得到迅速推广应用。1984 年和 1985 年，Ashon-Tate 公司又陆续推出了 dBASE Ⅲ和 dBASE Ⅲ PLUS，1986 年 dBASE Ⅲ PLUS 被评为美国最佳软件，一直发展到 1989 年推出的 dBASE Ⅳ。

虽然 dBASE 在 16 位机市场风靡一时，但是也存在许多弱点，如运算能力弱、处理速度慢、无数组和自定义函数、菜单单调等，从而限制了它的进一步发展。1987 年美国 Fox software 公司推出了与 dBASE 兼容的 FoxBASE+1.0，不仅功能更强，处理速度也明显提高，很快占领了微机数据库市场。

Fox software 公司为了扩大市场占有率，先后推出了 FoxBASE+2.0、FoxBASE+2.1 版本。1989 年该公司开发了 FoxBASE+的后继产品 FoxPro。1992 年美国 Microsoft 公司收购了 Fox software 公司，使 FoxPro 得到了更迅速的发展。

1993 年 1 月，Microsoft 公司推出了 FoxPro 2.5 for DOS 和 FoxPro 2.5 for Windows 两种版本，使微机关系数据库系统由基于字符界面演变到基于图形用户界面。1994 年发布了 FoxPro 2.6，与 2.5 版相比增加了多种“向导”工具，从而使用户的操作更简化。

随着可视化技术的迅速发展和广泛应用，Microsoft 公司将可视化技术引入了 FoxPro，于 1995 年推出了 Microsoft Visual Studio 组件，它包括 Visual Basic、Visual C 和 Visual FoxPro 等编程工具。

1998 年 Microsoft Visual Studio 6.0 组件发布，它包括 Visual Basic 6.0、Visual C 6.0 和 Visual FoxPro 6.0 等编程工具，将微机数据库技术推向了一个新阶段。

2-1-2　Visual FoxPro 系统特点

FoxPro 发展到 Visual FoxPro，功能日益强大，操作更加灵活。从数据库应用程序的设计方面看，已经从面向过程的结构化程序设计方法发展到面向对象的程序设计方法，下面介绍 Visual FoxPro 6.0 的几个显著特点。