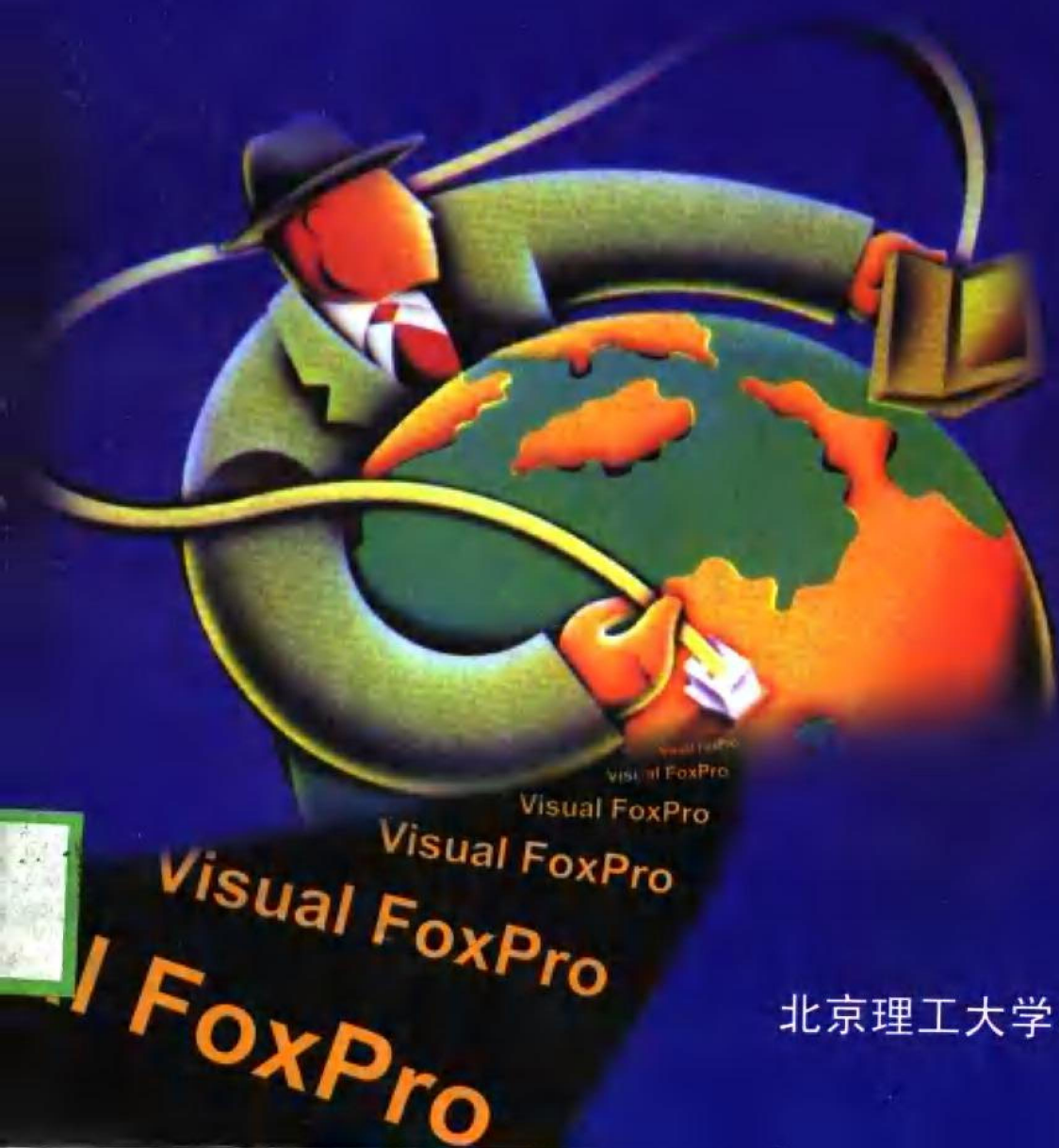


中文

Visual FoxPro 5.0

编程指南

鲲鹏工作室 编著



北京理工大学出版社

内 容 简 介

本书全面地介绍了中文 Visual FoxPro(VFP) 5.0 的相关内容, 并对基本应用类型进行分类, 由浅入深地讲述了 VFP 的实际操作及其具体功能的实现。

本书在结构上分为两个层次, 第一层次是简单应用层, 读者学完这部分后, 对 VFP 5.0 就有了一些基本的概念, 并能解决基本的工作需要。第二层次是高级应用层, 掌握了这部分内容, 读者就能上升到程序员的水平, 随心所欲地遨游在 VFP 5.0 的世界里。当然读者也可根据自己的需要选学有关内容。

本书可作为计算机用户学习 Visual FoxPro 5.0 应用和编程的教材, 也可供从事计算机工程与应用的读者参考。

图书在版编目 (CIP) 数据

中文 Visual FoxPro 5.0 编程指南/鲲鹏工作室编著. —北京: 北京理工大学出版社, 1998. 7
ISBN 7-81045-417-X

I. 中… II. 鲲… III. 关系数据库-数据库管理系统/Visual FoxPro 5.0 IV. TP311.13

中国版本图书馆 CIP 数据核字 (98) 第 10191 号

责任印制: 刘季昌 责任校对: 陈玉梅

北京理工大学出版社出版发行

(北京市海淀区白石桥路 7 号)

邮政编码 100081 电话 (010) 68912824

各地新华书店经售

北京房山先锋印刷厂印刷

787 毫米×1092 毫米 16 开本 23.25 印张 566 千字

1998 年 7 月第 1 版 1998 年 7 月第 1 次印刷

印数: 1—4000 册 定价: 34.50 元

※图书印装有误, 可随时与我社退换※

前 言

在过去很长的一段时间内，dBASE、Foxplus、Foxbase、Foxpro 等产品一直深受用户喜爱。尤其是微软公司推出的 Visual FoxPro(VFP)系列，以灵活的界面、表单、菜单的设计给我们带来了极大的方便。VFP 3.x 及其后继的版本，使用可视化的编程技术，使我们的开发工作更加方便；由于更新了数据库的概念并支持客户机/服务器编程，使得它在大型网络中与其他数据库配合无间；另外它使用了面向对象编程的概念，可以像 C 语言一样有能力满足各种各样的复杂要求，为程序的再开发节省了大量的人力和物力。

本书共分三个部分：第一部分是 VFP 的简介，介绍了 VFP 的一些近况及其应用；第二部分是速成篇，立足于让读者尽快学会 VFP 的使用，基本上能应付工作需要；第三部分是进阶篇，主要是供不满足于简单应用层次的读者查阅，以期提供一定的帮助。

编写这本书的目的是为了将 VFP 介绍给广大的读者，使大家能和我们一样领略到 VFP 的魅力。在写这本书时，我们立足于中文 VFP 5.0，尽量用简洁明了的语言介绍 VFP 的各种应用。如果你是 VFP 的初学者，就没有必要从旧的版本学起，你会发现这本书是很好的入门老师。如果你已是熟练的数据库程序员，这本书也可以成为你的益友，伴你进入一个新的天地。

本书由鲲鹏工作室编写，参加编写和资料整理工作的有姜汉龙、王竞、季宏、罗华烨、李节、冯勇、徐卫晨、赵志立、程风、甘心萍、金子、胡曼丽、谷丰收、莫阳、杜士和、范迪安、朱少敏、郭梧秀、宋林凌、祝福、郑武、刘达吉、张洪民、赵笑林、邹峰等。由于我们水平有限，时间又很仓促，书中定有许多不足之处，敬请广大读者批评指正。

编 者

1997.12

目 录

第一篇 速成篇

第 1 章 FoxPro 和数据库	(3)	5.4 表单的管理	(50)
1.1 Visual FoxPro 的起源	(3)	5.4.1 表单的保存	(50)
1.2 Visual FoxPro 的特点	(3)	5.4.2 表单的运行	(50)
1.3 VFP 数据库的概念	(4)	5.4.3 管理表单的多个实例	(50)
1.4 VFP 数据库的基本内容	(5)	5.4.4 创建单文档和多文档界面	(52)
1.5 VFP 数据库中的文件	(6)	5.5 表单的修改	(54)
第 2 章 数据库的建立	(7)	5.5.1 打开表单设计器(Form Designer)	(54)
2.1 表的建立	(7)	5.5.2 打开修改表单文件时需要的工具箱	(55)
2.1.1 表的生成	(7)	5.5.3 坐标的设定与利用	(55)
2.1.2 表的浏览	(10)	5.5.4 选择需要修改的目标对象	(55)
2.1.3 表的编辑	(10)	5.5.5 选择性地显示目标对象的属性	(56)
2.2 表的索引的建立和删除	(13)	5.5.6 修改目标对象的属性	(57)
2.3 数据库(DBC)的建立	(16)	5.6 调色板的使用	(59)
2.4 视图(VIEW)的建立	(18)	5.6.1 改变界面的颜色	(60)
2.4.1 为什么要建立视图	(18)	5.6.2 定义用户自己的颜色	(60)
2.4.2 建立一个视图	(18)	5.7 利用 Form 工具美化用户应用程序	(61)
2.4.3 联接的建立	(20)	5.7.1 建立一个新表单文件	(61)
2.4.4 视图的输出和储存	(22)	5.7.2 加入一个标题“标签(Label)”对象	(61)
2.4.5 视图文件的浏览和修改	(23)	5.7.3 让标签具有立体感	(62)
第 3 章 查询文件的生成	(25)	5.7.4 加入图形对象	(62)
3.1 概述	(25)	5.7.5 完成一个封面	(63)
3.2 查询文件的生成	(25)	5.8 Timer 的添加	(64)
3.3 定义结果	(27)	第 6 章 报表的生成	(66)
3.4 设置输出字段的次序	(28)	6.1 建立一个报表格式文件	(66)
3.5 组织输出结果	(29)	6.2 设计标题	(67)
3.5.1 排序查询结果	(29)	6.3 设计报表内容输出	(68)
3.5.2 分组查询结果	(30)	6.4 设计页号	(69)
3.6 运行查询文件	(31)	6.5 加上表格线	(70)
3.7 查询文件的其他输出方式	(31)	6.6 调整和修饰	(70)
3.8 利用向导建立查询文件	(33)	6.6.1 激活对象	(70)
第 4 章 菜单的建立	(37)	6.6.2 设定字型和字号	(71)
4.1 简介	(37)	6.6.3 输出窗口和边框的调整	(71)
4.2 具体步骤	(37)	6.6.4 直线段的定义和使用调色板	(71)
第 5 章 表单的生成	(42)	6.6.5 利用圆角框作台头修饰	(72)
5.1 概述	(42)	6.6.6 打印设置	(72)
5.2 新表单的创建	(42)	第二篇 进阶篇	
5.3 在表单中添加对象以及对象的处理	(47)	第 7 章 基本命令简介	(77)
5.3.1 在表单中添加对象	(47)	7.1 命令概述	(77)
5.3.2 对象的处理	(49)		

7.1.1 范围参数	(77)	8.2.7 判别编译命令(# IF... # ENDIF)	(103)
7.1.2 FOR 和 WHILE	(78)	8.2.8 嵌入文件(# INCLUDE)	(105)
7.1.3 了解工作区	(78)	8.3 存储器变量	(106)
7.2 命令窗口	(78)	8.3.1 变量的命名	(106)
7.2.1 RUN 命令	(78)	8.3.2 全程变量(GLOBAL MEMORY VARIABLE)	(108)
7.2.2 SET DEFAULT TO 命令	(78)	8.3.3 专用变量(PRIVATE VARIABLE)	(109)
7.2.3 命令窗口快捷操作	(79)	8.3.4 区域性变量(REGIONAL VARIABLE)	(110)
7.2.4 输入试用命令	(79)	8.3.5 局部变量(LOCAL VARIABLE)	(112)
7.3 建立新表的命令	(80)	8.3.6 系统变量(SYSTEM VARIABLE)	(113)
7.3.1 CREAT 命令	(80)	8.3.7 其他存储器变量管理命令	(113)
7.3.2 MODIFY 命令	(80)	8.3.8 存储器变量的储存及取出	(114)
7.3.3 建立和修改其他类型文件的命令	(80)	8.4 宏与名称表达式	(116)
7.4 处理数据的命令	(81)	8.4.1 宏替换(&)	(116)
7.4.1 USE 命令	(81)	8.4.2 名称表达式[()]	(118)
7.4.2 DISPLAY 命令	(81)	8.4.3 执行函数命令(=)	(120)
7.4.3 BROWSE 命令	(82)	8.5 索引及记录处理	(120)
7.4.4 添加空格和替换数据的命令	(82)	8.5.1 查找单个记录	(121)
7.4.5 INSERT 命令	(83)	8.5.2 子句介绍	(123)
7.4.6 GO 命令	(83)	8.5.3 使用 REPLACE 命令操作数据	(125)
7.4.7 删除命令	(84)	8.5.4 在索引中建立 FOR 子句	(126)
7.4.8 DISPLAY 和 LIST 命令	(84)	第9章 结构、子程序和函数	(127)
7.5 索引	(86)	9.1 条件结构	(127)
7.5.1 索引是什么	(86)	9.1.1 IF...ELSE...ENDIF 语句	(127)
7.5.2 控制索引	(86)	9.1.2 DO CASE...ENDCASE 语句	(131)
7.5.3 什么时候建立索引	(86)	9.2 循环结构	(133)
7.5.4 建立索引	(87)	9.2.1 固定次数循环(FOR...ENDFOR)	(133)
7.5.5 设置索引顺序	(87)	9.2.2 当型循环(DO WHILE...ENDDO)	(136)
7.6 退出 Visual FoxPro 命令	(88)	9.2.3 数据库扫描(SCAN...ENDSCAN)	(138)
第8章 程序设计的基本知识	(89)	9.3 子程序与函数	(140)
8.1 从一个简单的程序开始	(89)	9.3.1 子程序及函数的意义	(140)
8.1.1 作好编程的准备	(89)	9.3.2 系统内部函数介绍	(141)
8.1.2 开始键入程序	(89)	9.3.3 子程序及自定义函数	(145)
8.1.3 储存及放弃	(90)	9.3.4 子程序及自定义函数的命名	(147)
8.1.4 程序的运行	(90)	9.3.5 子程序和函数的执行与结束	(147)
8.1.5 中断程序	(91)	9.3.6 程序和函数的执行顺序	(149)
8.1.6 程序的暂停	(91)	9.3.7 函数的返回值	(151)
8.1.7 继续执行程序	(92)		
8.1.8 基本输出命令(?)	(92)		
8.2 数据类型、运算符及预编译指令	(93)		
8.2.1 数据类型	(93)		
8.2.2 各类运算符	(94)		
8.2.3 表达式	(95)		
8.2.4 命令和函数的使用	(96)		
8.2.5 定义常数(# DEFINE)	(100)		
8.2.6 取消常数定义(# UNDEF)	(102)		

9.4 参数的传递方式	(151)	12.2.9 清除缓冲区(FFLUSH())	(222)
9.4.1 改变参数传递方法	(152)	12.2.10 改变文件大小(FCHSIZE())	(222)
9.4.2 两种传递方法的比较	(152)	12.2.11 关闭低级文件(FCLOSE())	(223)
第10章 数组	(154)	12.2.12 取得文件函数执行错误码 (FERROR())	(223)
10.1 数组的使用	(154)	12.3 低级文件函数的使用	(223)
10.2 数组的声明	(154)	12.3.1 数据表保护	(223)
10.3 数组元素的使用	(157)	12.3.2 备注文件的救回	(226)
10.4 数组元素的处理	(159)	12.3.3 模拟拷贝程序	(228)
10.4.1 数组元素的排序(ASORT()) ..	(159)	12.3.4 修改 WIN.INI	(229)
10.4.2 元素的查找(ASCAN())	(161)	12.3.5 解救结构化索引文件的数据表	(231)
10.4.3 元素的删除(ADEL())	(163)	12.4 再谈句柄	(231)
10.4.4 元素的插入(AINS())	(164)	第13章 基本输入输出界面的程序设计	(234)
10.4.5 数组的传递	(165)	13.1 界面概述	(234)
10.4.6 综合举例	(167)	13.2 最基本的屏幕输入输出	(236)
10.5 另一种记录存取技巧	(171)	13.3 定位输出命令(SAY)	(238)
10.5.1 使用一般存储器变量	(171)	13.3.1 屏幕的定位问题	(238)
10.5.2 使用数组的方法	(173)	13.3.2 输出数据	(240)
10.6 数组与记录的处理	(175)	13.3.3 设定字型、大小及样式	(240)
10.6.1 从记录到数组	(175)	13.4 定位输入命令(GET)	(241)
10.6.2 从数组到记录	(178)	13.5 启动 GET 对象	(246)
10.6.3 两组命令的比较	(181)	13.5.1 基本的 READ	(246)
第11章 结构化查询语言——SQL	(182)	13.5.2 循环读取	(246)
11.1 查询命令 SELECT——SQL	(182)	13.5.3 保留读取状态	(246)
11.1.1 基本命令	(183)	13.5.4 设定读取边框	(247)
11.1.2 其他关键字的使用	(190)	13.6 其他有关屏幕的命令	(247)
11.1.3 高级数据查询	(191)	13.6.1 清除屏幕命令(CLEAR)	(247)
11.2 其他 SQL 命令	(206)	13.6.2 等待命令(WAIT)	(247)
11.2.1 建立暂存数据表(Cursor)	(206)	13.7 组件设计	(248)
11.2.2 建立实体数据表	(209)	13.7.1 按钮(Push Button)	(248)
11.2.3 修改数据表结构	(212)	13.7.2 复选框(CHECK BOX)	(250)
11.2.4 插入记录	(214)	13.7.3 单一选择钮(RADIO BUTTON)	(251)
11.2.5 删除记录	(215)	13.7.4 显示图形	(253)
11.2.6 设置字符串比较方式	(215)	13.7.5 隐藏式按钮(INVISIBLE BUTTON)	(254)
第12章 低级文件的控制	(216)	13.7.6 下拉式列表框(LIST)	(254)
12.1 低级文件的概念	(216)	13.7.7 弹出式菜单(POPUP)	(257)
12.2 低级文件函数	(217)	13.7.8 文字编辑区(TEXT EDITING REGION)	(258)
12.2.1 建立文件(FCREATE())	(217)	13.7.9 微调控制项(SPINNER)	(258)
12.2.2 打开文件(FOPEN())	(218)	13.8 各命令共用的参数介绍	(260)
12.2.3 移动文件指针(FSEEK())	(218)		
12.2.4 取得文件尾部的信息(FEOF())	(219)		
12.2.5 FGETS()函数	(219)		
12.2.6 读取数据(FREAD())	(220)		
12.2.7 写入数据(一)(FPUTS())	(220)		
12.2.8 写入数据(二)(FWRITE())	(221)		

13.8.1 设定对象的显示区域	(260)	15.5.2 使用方法程序代码添加类和创建类	(304)
13.8.2 对象的卫士	(261)	15.5.3 创建表定位按钮集合	(306)
13.8.3 对象的另一个卫士 (VALID) ..	(261)	15.5.4 创建基于新类的子类	(308)
13.8.4 新旧转换法	(262)	15.5.5 定义表格控件	(310)
第 14 章 菜单的设计	(265)	15.5.6 创建对象的引用	(312)
14.1 以前的一些菜单设计技术	(265)	15.5.7 创建成员数组	(313)
14.2 简单型菜单	(269)	15.5.8 使用对象存储数据	(314)
14.3 光条式菜单	(271)	15.6 VFP 中的事件	(316)
14.4 常驻型菜单	(275)	15.6.1 Visual FoxPro 中的事件	(316)
14.5 快速菜单	(279)	15.6.2 容器事件和对象事件	(316)
14.6 下拉式菜单	(281)	15.6.3 类和控件事件	(318)
14.6.1 使用的命令	(281)	15.6.4 追踪事件序列	(318)
14.6.2 设定选项标记	(285)	15.6.5 Visual FoxPro 事件顺序	(322)
14.6.3 设置选项能否被选取	(286)	15.6.6 为事件指派代码	(323)
14.7 释放各类菜单及选项	(287)	第 16 章 高级界面的设计	(325)
第 15 章 面向对象的程序设计概念	(289)	16.1 选项按钮	(325)
15.1 Visual FoxPro 中的对象	(289)	16.2 下拉式列表框	(327)
15.2 Visual FoxPro 中类的概念	(290)	16.3 编辑区	(331)
15.2.1 类与对象的关系	(290)	16.4 页框对象	(334)
15.2.2 Visual FoxPro 中类的特性	(290)	16.5 列表框组件	(336)
15.2.3 Visual FoxPro 基类	(291)	16.6 微调控件项	(338)
15.2.4 Visual FoxPro 类的层次	(292)	16.7 图形控件	(341)
15.2.5 类和容器的层次结构	(293)	16.8 图对象	(344)
15.3 类的创建	(295)	第 17 章 键盘与鼠标的控制	(346)
15.3.1 创建新类	(295)	17.1 按键的取得	(346)
15.3.2 修改类定义	(295)	17.1.1 READKEY()函数	(346)
15.3.3 新类的进一步修改	(296)	17.1.2 取得最后一键	(348)
15.3.4 为类指定设计时的外观	(299)	17.1.3 INKEY()函数	(351)
15.3.5 使用类库文件	(299)	17.1.4 多重按键拦截	(353)
15.4 类和任务匹配	(301)	17.1.5 按键的储存及回存	(358)
15.4.1 创建有多个组件的控件	(301)	17.2 鼠标的控制	(358)
15.4.2 创建非可视类	(301)	17.2.1 鼠标按钮的拦截	(358)
15.4.3 设置属性	(301)	17.2.2 取得鼠标目前的位置	(360)
15.4.4 调用方法程序	(302)	17.2.3 取得鼠标的移动状态	(360)
15.5 从程序的角度讨论类	(303)	17.2.4 鼠标的按键状态	(362)
15.5.1 以编程方式定义类	(303)		

第一篇

速成篇



第1章 FoxPro 和数据库

1.1 Visual FoxPro 的起源

80 年代, 在个人计算机的数据库市场上, Ashton-Tate 的 dBase 占据支配地位。在我国, 也有大量用户在使用 dBase。dBase 经历了一系列的演变过程, 很多公司对其不足之处进行了大量的改进。而从 dBase 发展到 Foxbase, 使用户所享有的产品在速度和程序设计能力两方面都有了较大的提高。尽管如此, 这些产品都有一个难以克服的缺点: 难以使用, 界面不友好。

FoxPro1.0 于 1989 年代替了 Foxbase, 后来 Fox Holding 公司将其发展成为 FoxPro1.2 版至 2.0 版, 都为 Dos 版本, 当时在 PC 机数据库管理系统上, FoxPro 与 dBase 是分庭抗礼的。数年前, 微软 (Micorsoft) 公司收购了 Fox Holding。由于微软公司在可视化面向对象程序设计方面的实力, FoxPro 像坐了快车一样, 迅速地被改造为 Visual FoxPro, 将其他产品远远地抛在后面。

FoxPro 经过从 2.0 版 (Dos) 到 2.6 版 (Dos/Win) 多次的改版, 已具备了相当独到的功能。从 3.0 版开始 FoxPro 正式改名为 Visual FoxPro, 和其他 Visual 系列软件一样, Visual FoxPro 为一般用户和编程人员都提供了很大的方便。

1.2 Visual FoxPro 的特点

在 1989 年, 当 FoxPro 1.0 代替 Foxbase 时, 使用 Fox 语言的编程人员经历了一次巨大的编程策略的转变, 而 Visual FoxPro (VFP) 中面向对象的编程策略的转变比 1989 年由 Foxbase 到 FoxPro 的转变还要大。如果把前一次转变看成是由曲线发展到直线发展的转变, 那这一转变就可以看成是由直线发展到跳跃式发展的一个转变, 这对于适应当今世界对软件产品日益增长的需求, 有着不可估量的意义。

在 VFP 的第一个版本中, 面向对象的程序设计方法使得许多程序的运行速度似乎变慢了, 这主要是由于那些新特性的开销。在 VFP 5.0 中, 开发者们改进了大部分特性和运行的方法, 程序的运行速度要快一些, 基于对象的代码也运行得更快。可以说, VFP 5.0 这一版的公布使选择 VFP 的软件人员对自己的眼光大感欣慰。

如果以前用户一直使用 FoxPro 2.x, 那 VFP 会使你大开眼界。尽管 VFP 可运行早期 FoxPro 的 Windows 版本中已存在的许多应用程序, 但如果要想真正了解它的内涵, 就要结合新特性来使用它。它的新特性表现在以下几个方面:

- 面向对象的编程方法。它向人们展示类的概念, 以及各种类的属性、事件和方法, 使人们体会到了编程天地的广阔。现在, 开发者可使用任何基类来开发补充 VFP 中内

建特性的新类或创建新类。用户可将这些类建成类库并在应用程序中使用它，使自己程序设计这辆车开得更快。

- 重新定义的数据库引入了数据字典的概念，它使开发人员可以使用一个“中央存储仓库”来存储有关表格的信息。它使用户定义长字段和表格名成为可能。更为重要的是，它使开发者可以使用存储过程为每张表格指定缺省值、有效经验值以及触发器等。引用的一致可以使用户更为方便地使用它，因为它可以直接建在表格中而不需要在应用程序中编程。
- 向导的集合有所扩展以便处理更多的任务，它增加了构建器以帮助完成其他的工作，它使用户对表单对象定义属性等操作变得非常容易。
- 项目管理器也被重新设计，它还结合了目录管理器的最好的特性以及拥有了编译应用程序的能力。用户在使用它时能够编译并运行单独的模块。
- 以一个新的调试器代替跟踪和调试窗口。新的调试器增加的功能可使用户非常方便地查看内存变量、对象属性和调用栈顺序。同时该调试器最大的两个新特性是跟踪事件以确定它们被触发的顺序的功能以及确定跟踪作用次数的能力。
- 对 SQL 语句的增强使定义左外、右外和完全外部连接更容易。
- VFP 可以将可视类库编译成 OLE 服务器。任何可使用 OLE 自动化的应用程序在这些服务器中都能使用这些类库。
- VFP 的编译器现在使用了语法着色，它将代码中的不同部分用不同的颜色显示。这使用户在输入时快速识别输入的变量是否真的是一条命令或函数的名称。它也使代码中不同的部分，如注释部分，更加醒目。它使你觉得编程工作简直就是在完成一件艺术品。
- 使用字段映象，用户可以方便地做到：将表格中的字段拖到表单中或使用表格向导时不再需要为每个字段使用文本框（以前这样试过的用户一定会记得这有多麻烦）。现在用户可为每种字段类型指定任一个 VFP 基类。它使每个人都想试一试开发自己的类并使用它们。
- 除字段映象之外，VFP 在定义表单的标题、格式、输入屏蔽及字段注释等时还能自动使用从数据库容器中得到的其他信息。这使你在使用时又多了一个助手。
- 可视化原文件安全集成到 VFP 专业版中。在这儿用户可以为每个应用程序建立特定的版本，创建分支的应用程序，或者将这些分支合到一起加以使用，这对于现今世界对程序日益增长的需求有广泛的适应性。
- VFP Internet 向导可用于在 Internet 和用户的数据间创建界面，它可以使你溶合到 Internet 中去，在全球日益网络化的今天，有了它就有了更多地向世界展示的机会。
- 在 VFP 语言和对象模块中增加很多新的属性、事件、方法、命令和系统变量，它使你用起来更加得心应手。

1.3 VFP 数据库的概念

Visual FoxPro 5.0 废除了以往 FoxBase 以及 FoxPro 2.x 中不尽合理的数据库概念，而代之以真正的数据库概念。

以往的软件是以 DBF 作为数据库概念,这等于一个 DBF 文件就是一个存有记录的文件,而 3.x 版本将 DBF 视为一个数据表,对应的 DBF 文件储存的不单单是一个个记录,它还包括任意类型的数据、数据表之间的关联、数据的查询集合以及数据的来源等,因此 3.x 版本中的数据库将是一个具有数据字典功能的文件,这才是一个真正意义上的数据库。

在新的数据库概念中,原来的 DBF 变成了库中的一个表,不属于任何库的表称为自由表。这种新的库模型可以将表、表间的关系及内部的程序封装在一起,同时还定义许多存在逻辑关系的数据子集合,使用起来更加得心应手。

此外,这种新型的关系型数据库由于在结构上与 SQL 等标准兼容,从而使数据交换和互操作性的实现更加合理和方便。

1.4 VFP 数据库的基本内容

一个数据库 (DataBase Container) 包含如下信息:

- (1) 数据表 (Table) —— 它是一个记录的集合,在数据表中可以将格式化的记录按一定的次序排列,也可设置专用的索引,使记录按索引的次序排列。
- (2) 记录 (Record) —— 记录是一个数据表的最小处理单位,它可以由一个或几个字段组合而成,例如学生成绩统计表上的一个记录是由“学号”,“姓名”及“成绩”组合而成,它的字段数为 3 个。
- (3) 字段 (Field) —— 字段是记录的一部分,它储存了一些特定的数据,例如学生成绩统计表中的学号字段储存了学生的学号,“姓名”及“成绩”字段分别储存了学生的姓名和各课的成绩。这里每个记录的同一字段的长度一经设定后就不再改变。
- (4) 字符 (Character) —— 一个字符在计算机中占有一个字节,例如:“S”这个字母就是一个字符,它在存储器中占有一个字节,而一个字段是由几个字节组成。
- (5) 字段类型 (Type of Field) —— 每一个字段都拥有自己的长度以及储存的信息,例如:学号字段储存学生的学号,其字段类型可为“字符型”,“成绩”字段储存了学生各课的成绩,其字段类型可为“浮点数值型”,类型关系到处理数据的能力和范围,故在数据表的设计中应特别注意。

表 1.1 列出了 VFP 中的字段类型,以备查询。

表 1.1 数据类型一览

类 型	最大长度	备 注
字符型(Character)	*254	可储存任何字符
数值型(Numeric)	*8	储存数值数据
浮点数值型(Float)	8	储存数值数据
货币型(Currency)	8	储存货币数据
整数型(Integer)	4	只储存整数数据
双精度型(Double)	8	储存双精度数值数据
日期型(Date)	8	储存日期数据
日期时间型(DateTime)	8	储存日期和时间数据

续表 1.1

类 型	最大长度	备 注
逻辑型(Logical)	1	包括逻辑真 (.T.)或逻辑假(.F.)
字符型(Binary,二进制)	*254	可储存除 NULL 字符以外的任何字符
备注型(Binary,二进制)	4	可储存除 NULL 字符以外的任何字符
备注型(Memo)	4	任何字符,以及图形
通用型(General)	4	OLE 对象和图形
图形型(Picture)	4	图形, 适用于微软版
注: 表中*号表示该类型的最大长度, 其长度可小于该值。 无*的表示其长度是定值。		

(6) 记录指针(Pointer) ——通常数据表是由多个记录组成, 因此当前的记录需用指针随时标出, 如图 1.1 所示。

一系统管理数据库				
记录 1	字段 1	字段 2	字段 3	字段 4
记录 2	字段 1	字段 2	字段 3	字段 4 ←
记录 3	字段 1	字段 2	字段 3	字段 4
:	:	:	:	:
:	:	:	:	:
:	:	:	:	:
记录 n	字段 1	字段 2	字段 3	字段 4

图 1.1 数据表与记录

图 1.1 中, 当指针指向第二个记录时, 表示第二个记录是当前的记录, 所做的修改和显示都是针对第二个记录的。一个数据表只有一个指针, 也就是说记录指针在数据表中是唯一的, 如同时打开多个数据表, 则数据表 A 中关于记录指针的任何操作都不会在数据表 B 中引起任何后果。而当指针移动到第一个记录的前面, 则表示到了文件的开头; 当指针移动到最后一个记录的后面, 则表示到了文件的末尾。

1.5 VFP 数据库中的文件

在 Visual FoxPro 数据库中, 常用以下几种文件:

- (1) 数据库文件(DBC) ——这是一个表结构的文件。
- (2) 数据库索引文件(DCX) ——这种文件是通过索引的建立而建立起来的, 与 VFP 复合式结构索引文件类型相似。当打开数据库文件后, 通过 Data Session 工具查看时, 可以发现, 在该数据库文件中存在着识别索引。
- (3) 数据库备注字段文件(DCT) ——当 DBC 文件中具有许多备注字段类型时, VFP 一般将备注字段的内容存放到备注文件(Memo File ,FPT)中。当启动 DBC 中的备注字段浏览其内容时, 浏览的内容来源于 DCT 文件。

第2章 数据库的建立

2.1 表的建立

Visual FoxPro 5.0 与以往 FoxPro 数据库语言的一个重要区别在于数据库的概念发生了根本的变化，它不再仅仅是一个简单的二维表，它可以是封装在一起的几个二维表。可以把现在的表文件理解成原来的数据库文件，它是数据库的基础。

2.1.1 表的生成

下面以一个学生成绩表为例说明 VFP 中的表文件的建立过程。

首先进入 VFP 的菜单。如图 2.1 所示，在文件菜单选取“新建”项，屏幕中会显示如图 2.2 所示的“新建”对话框，对话框中各选项左侧有一些单选的按钮。系统默认为建立一个“项目”。

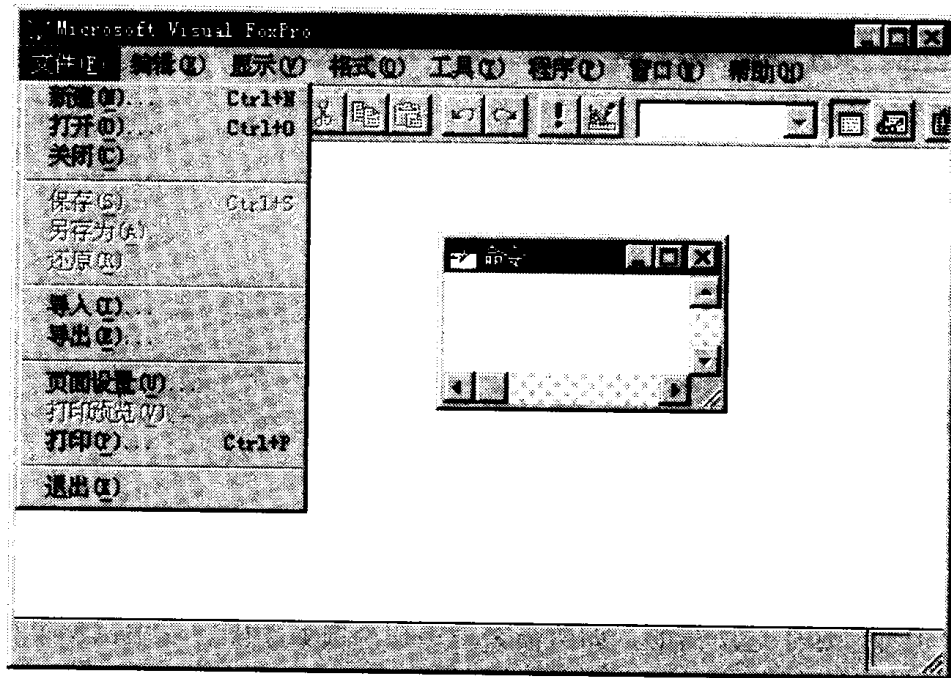


图 2.1 打开文件菜单

在选择“表”文件后，用鼠标点下“新建文件(N)”的图标，屏幕中将会弹出如图 2.3 所示的“创建”对话框，选择自己需要的目录和文件名，这儿选的是 d:\vfp50\student.dbf，在确认文件名称和目录后，屏幕将显示如图 2.4 所示的“表设计器”对话框，以建立所需要的数据库的字段。

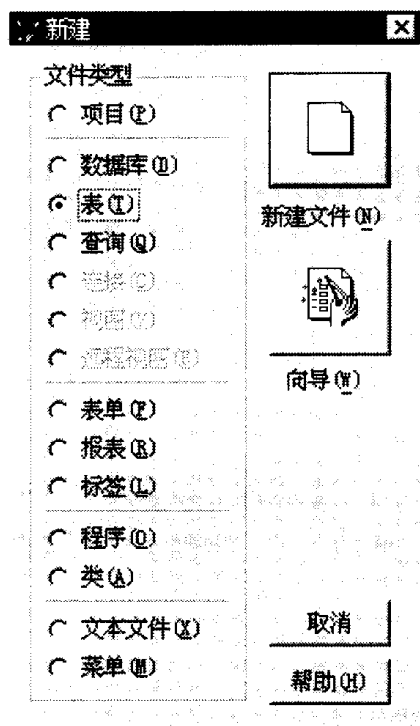


图 2.2 新建表

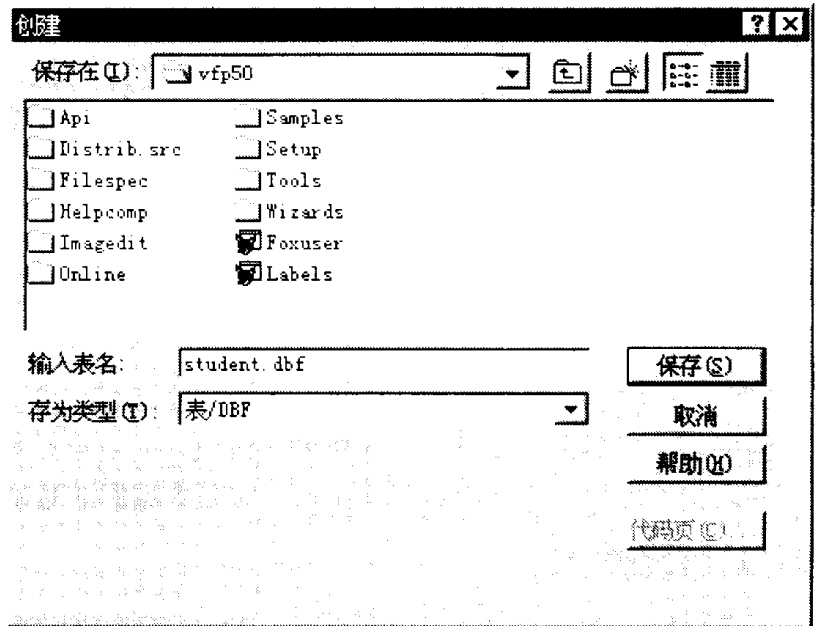


图 2.3 新表文件的建立

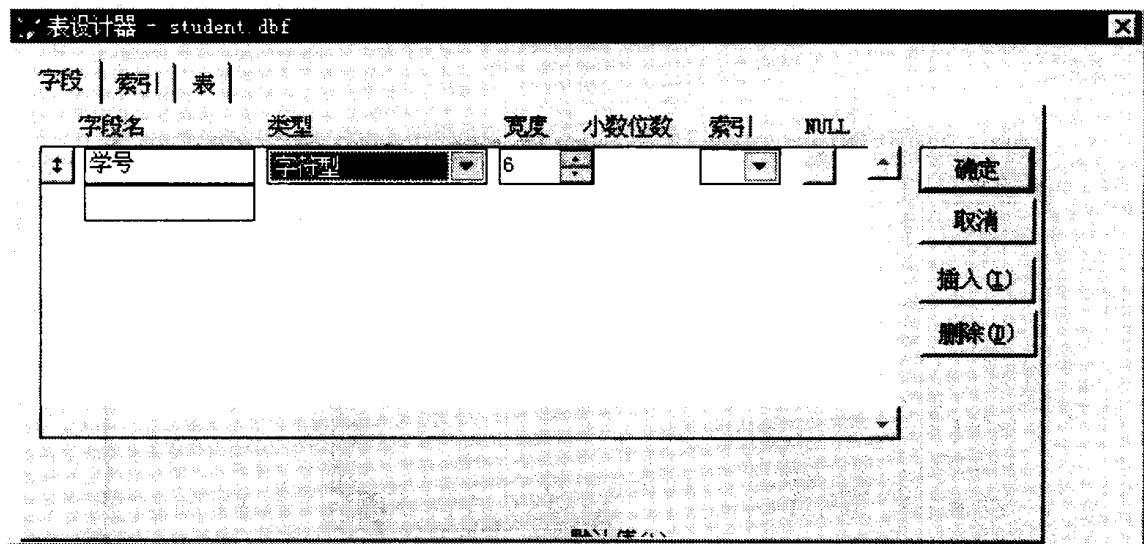


图 2.4 表中字段的设置

从图 2.4 中可以看出，其操作如下：

- (1) 在“字段名”域键入“学号”。
- (2) 在“类型”域中选所需的，如所需类型不在当前显示中，可用鼠标单击域右的下拉按钮选择所需的类型，这里选“字符型”。
- (3) 在“宽度”域中键入宽度，也可用鼠标点域右的扩大按钮和缩小按钮达到调整宽度的目的，这里选宽度为 6。
- (4) 在“索引”域中点下拉按钮，有三种选择：上升，下降以及不选。这里可根据需要来选，为作示范选择上升。

- (5) “NULL”域是以往的数据库所没有的，它用来决定是否允许键入无效的内容，系统默认为不允许，这里取默认值。
- (6) 重复操作（1）～（5）完成数据库的构造，它包括如图 2.5 所示的学号、姓名等九个字段。

设定完后，单击对话框中的确定键。系统将询问是否输入数据，请选择“是(Y)”。

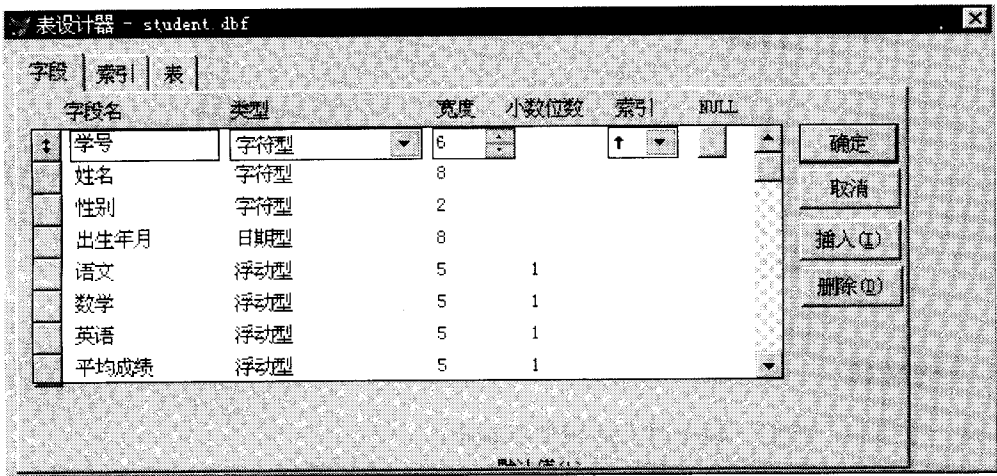


图 2.5 数据库的建立

随后弹出如图 2.6 所示的数据输入框，在框中输入数据，注意“总评”字段，如要输入信息，用鼠标双击“总评”字段域，屏幕中将弹出总评输入界面，输完想要的信息后，键入组合键“CTRL+W”存盘后退回数据输入界面。当该字段域中有信息时，原来的“memo”字样将会变成“Memo”。

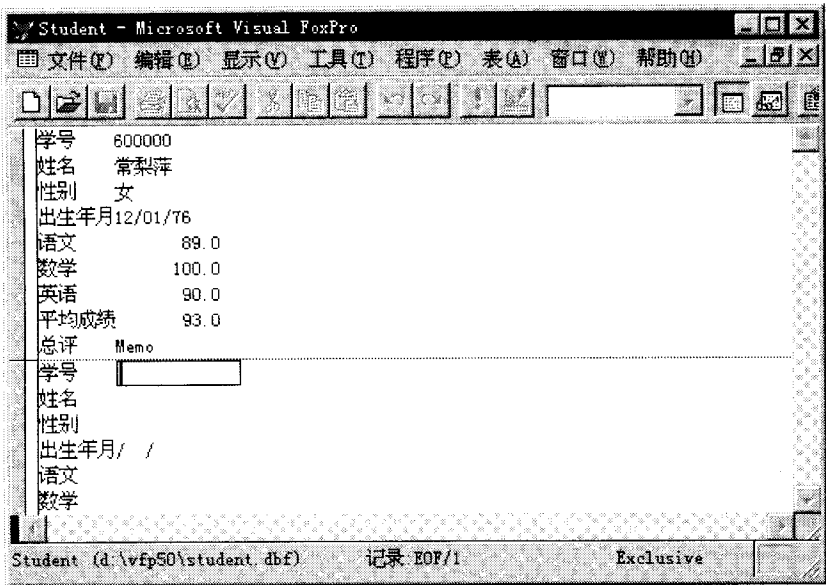


图 2.6 数据的输入

2.1.2 表的浏览

首先，在 VFP 的主菜单的“文件”选项上单点鼠标左键，然后选取其中的“打开(O)...”项，在打开文件的窗口选取 student.dbf。进入主菜单后，点下显示栏中的“浏览(B)”选项，得到如图 2.7 的表的显示图样。

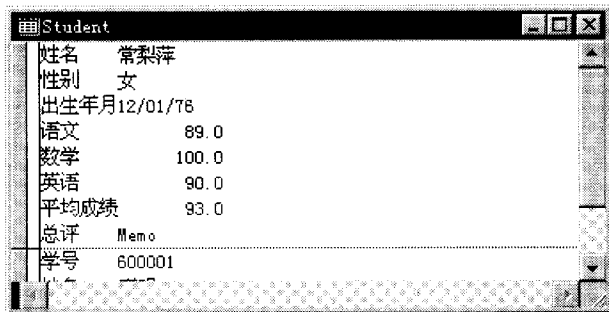


图 2.7 student.dbf 的浏览显示

2.1.3 表的编辑

在这一小节中，我们一起学习表的增加和编辑。

首先如上所述的那样打开文件，再打开浏览窗口。无论要修改哪个记录的内容，只要用鼠标点一下，使光标移动到目的项，就可以键入想修改的内容。如果想修改的记录不在当前窗口，可以通过点按滚动条找到此记录。另外还可以用此时出现在主菜单上的表项来移动记录的指针，如图 2.8 所示，移动到所需位置，然后就可添加新的记录，删除旧的记录，恢复已被修改的记录，改变字段的宽度以及移动字段的位置等等。

在这里要注意的是，对数据库结构的操作，如改变字段的长度，移动字段的位置等，只在浏览窗口中有效。例如，我们改变字段的长度，在浏览的窗口如我们所愿的改变了，但真正的数据库结构并没有改变，下次重新打开时会发现一切如故，但对记录的修改是同步的。当在浏览窗口修改了记录的内容，数据库中的内容也随之改变。

下面具体学习记录的删除，如图 2.8 所示将指针移到所要删除的记录位置，或者如图 2.9 所示采取指针直接定位的方法将指针指向想删除的记录，然后单击“切换删除标记”选项，则屏幕中显示的第三个记录前的长方条被涂黑，如图 2.10 所示以表示该记录将被删除。这里要注意的是，“切换删除标记”选项是一个开关式的选项，第一次点下，记录被加上删除标志，再点一下，记录就被恢复。当加上删除标志后，如确认删除，可单击“彻底删除”记录的选项，此时记录被删除后将不可再恢复。我们也可以利用系统提供的“删除记录”选项，“恢复记录”选项来达到目的。单击“删除”选项后，弹出如图 2.11 所示的“删除”的对话框，在作用范围域内选择记录，并指向想删除的记录号，然后点“删除”键，在浏览窗口我们会发现相应记录已被加上删除标志。如不想删除，单击“恢复记录”选项即可。