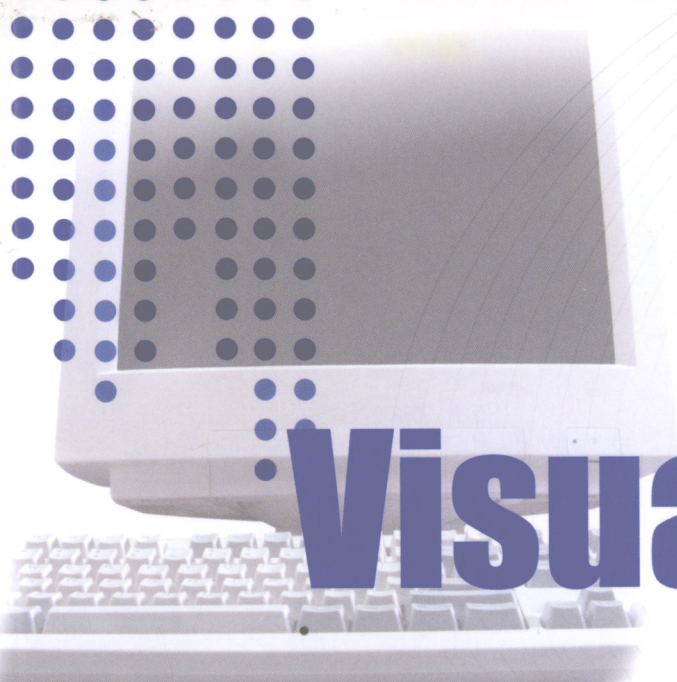




Visual Basic

电子商务

项目方案 精解



赠送

光盘中有每个实例制作过程的多媒体演示，并赠送了素材和模板

王健南 编著

航空工业出版社

程序开发项目方案精解丛书

Visual Basic 电子商务

项目方案精解

王健南 编著

航空工业出版社

内 容 提 要

Visual Basic 是数据库开发的有用工具,在企业信息管理系统开发中具有广泛的使用。本书以几个典型实例为导向,循序渐进地介绍了如何使用 Visual Basic 进行企业电子商务系统的开发工作。

本书注重实用性,所有程序代码编写全部手工完成,并可正常执行。本书内容繁多、结构紧凑,要求读者最好具有一定的 VB 编程和数据库操作基础和经验。本书对于正在从事或将要从事企业电子商务系统开发、使用或维护的相关人员具有更大的借鉴作用。

图书在版编目(CIP)数据

Visual Basic 电子商务项目方案精解 / 王健南编著.

北京:航空工业出版社,2006.10

ISBN 7-80183-820-3

I.V… II.王… III. ①企业管理—电子商务—系统设计②BASIC 语言—程序设计 IV.①F274—39②TP312

中国版本图书馆 CIP 数据核字(2006)第 111589 号

Visual Basic 电子商务项目方案精解 Visual Basic Dianzishangwu Xiangmu Fanganjingjie

航空工业出版社出版发行

(北京市安定门外小关东里 14 号 100029)

发行部电话: 010-64919539 010-64978486

北京航宇印刷厂印刷

全国各地新华书店经售

2006 年 10 月第 1 版

2006 年 10 月第 1 次印刷

开本: 787×1092

1/16

印张: 30.75

字数: 763 千字

印数: 1—6000

定价: 49.80 元

编审委员会名单

主任委员 肖治垣 王战航 崔亚量

副主任委员 皇甫满喜 王健南 王金岗

执行委员 崔慧勇 姜谷鹏 柏 松

委 员 (以姓氏笔画为序)

于淑芳	马洪儒	王卫华	王 宇	王 铁	王 萍
王 惠	王锦武	孔 娟	太洪春	邓毅夫	冯 颖
石 磊	石蔚云	司清亮	叶 勇	华 云	刘 冰
刘 去	刘 军	刘桂花	米西峰	先 云	先 勇
先 锋	闫广平	闫起亮	朱贵宪	邢素萍	陈春松
陈良琴	李东南	李安伏	李志川	李林义	李 娜
杜传宇	吴云花	吴允波	芦淑珍	张肖洁	张丽莉
张 鹏	林 锋	杨庆祥	杨端阳	郎建昭	治 国
柏仁能	段转平	柳志新	赵明生	赵拥军	郭东恩
柴方艳	聂爱丽	梁为民	梁玉萍	黄苏桥	韩翠英
谭中阳	谭 贤	翟秋菊	潘 瑾		

前言

从最早的打字机、复印机、传真机等办公设备和文字处理软件的使用，到现在利用计算机进行存储、排版、输出以及对人事、财务等企业方方面面进行管理，未来的企业商务将会是网络的广泛应用。当今企业，在当前的经济形式和新的竞争环境下要想赢得竞争优势，就必须变革企业现有的管理模式，必须进行企业流程再造，以求得在速度、质量、效率和柔性等各方面经营特性的巨大提高。

目前我国许多企业的信息化技术应用水平还偏低，处于企业信息管理的初级阶段。虽然计算机用于企业管理的比重不低，但应用水平并不高。这主要是因为管理技术水平落后，系统管理软件不完善，限制了信息技术作用的发挥，并缺乏合理的总体开发策略。企业可以通过自行编制电子商务系统软件，使企业的信息传递和办事效率大大提高。使员工之间、员工与管理层之间的沟通、资源共享等大为改善，也使日常工作安排、会议记录、车辆调度等事务通过系统顺利进行；同时，人事资源管理、企业采购管理、产品销售管理、物资库存管理、员工考勤管理以及统计报表查询等也逐步实现计算机管理。

本书各章典型实例都是考虑了行业的真实需求所选取和编写的。在项目功能和数据结构的设计上精心挑选和组织，并兼顾到普通需求和特殊需求，在程序设计上要求达到界面友好、操作简便、代码规范，并且根据企业实际应用情况，使用了多种数据库访问方式。

本书的各实例采用了符合实际设计的思想，按合理的顺序进行章节的编排。主要包括：

(1) 系统总体设计：分析此实例系统的应用背景、编写目的、模块功能划分。

(2) 数据库设计：在所有信息自动化系统的设计之初，在代码编写之前，必须完成对数据库信息的设计，保证数据库结构完整、建立数据库模型、添加所有所需的数据库中的数据表。

(3) 技术细节：对程序设计中可能遇到和使用到的相关重要技术，包括编程中或数据库管理上所需应用的技术方法等。

(4) 程序实现：从主窗体模块设计开始，逐步实现各个功能模块的界面和代码编写，最终完成一个满足企业需求的完整的系统实例。

(5) 附带光盘：随书附带的光盘中存有各个实例的源代码和经过打包后的可安装文件。对于书中无法全部列举出的详细代码，读者可以通过参考光盘中的源代码和代码中说明文字进行进一步学习。

本书由王健南主编，参加编写和修改的人员还有汉文、王强、周亚玲、黄峰、郎显源、罗颂、曹广薪、潘力、孙雄勇、余周军、陈艳华、赵凯、袁海波、赵艳锋、赵静一、秦鹏、苏治中、石伟玉、黄东、黄荣升、王豫、姚文浩、张建平、孙逊、叶顺源、韦韩等。本书覆盖知识面广泛，由于编写时间较为仓促，书中难免会有疏漏和不足之处，恳请广大读者提出宝贵意见。

读者朋友们，如果在使用过程中碰到不懂的地方，或发现书中存在错误及疑问之处的，请发送 E-mail 到 hangkongbooks@126.com，我们会尽快与您联系，共同解决问题。

编者

2006 年 9 月

目 录

第1章 电子商务系统开发基础

1.1 Visual Basic 进行数据库开发的特点	1
1.1.1 Visual Basic 开发数据库的优势	1
1.1.2 Visual Basic 提供的数据库访问技术	2
1.2 用 SQL 访问数据库	2
1.2.1 SQL 语言介绍	2
1.2.2 SQL 语句创建数据库	3
1.2.3 SQL 语句选取数据库	6
1.2.4 SQL 语句更新数据库	13
1.2.5 Visual Basic 中调试 SQL 语句	15
1.3 ActiveX 数据对象访问 (ADO) 技术	17
1.3.1 ADO 概述	17
1.3.2 ADO 的对象	18
1.3.3 在 Visual Basic 中使用 ADO	20
1.3.4 到数据源的连接	20
1.3.5 使用记录集	24
1.3.6 使用命令	31
1.3.7 使用字段信息	33
1.3.8 使用 ADO 错误处理	35

第2章 企业员工信息管理系统

2.1 系统总体设计	38
2.1.1 应用背景	38
2.1.2 项目目标	38
2.1.3 解决方案设计	39
2.1.4 系统模块功能分析	39
2.2 数据库设计	41
2.2.1 数据库需求分析	41
2.2.2 E-R 模型分析	42

2.2.3 数据表设计	43
2.3 技术细节	45
2.3.1 使用 ActiveX 数据对象 (ADO) 访问数据库	45
2.3.2 主要控件	45
2.4 主窗体模块	46
2.4.1 窗体界面设计	46
2.4.2 添加主窗体初始代码	48
2.5 登录窗体模块	52
2.5.1 界面设计	52
2.5.2 编写代码	54
2.6 员工档案信息模块	56
2.6.1 员工档案建立	56
2.6.2 员工档案修改	66
2.6.3 员工档案查询	71
2.6.4 员工档案删除	75
2.7 员工异动信息模块	78
2.7.1 员工异动操作	78
2.7.2 员工异动信息查询	84
2.7.3 员工异动信息修改	88
2.7.4 员工异动信息删除	93
2.8 人员统计分析模块	94
2.8.1 按聘用年月绘图	94
2.8.2 按学历绘图	102
2.8.3 按部门绘图	104
2.9 系统设置模块	107
2.9.1 增删管理员用户	107
2.9.2 修改登陆密码	111
2.10 扩充和提高	114
2.11 小结	114

第3章 财务管理系统

3.1 系统总体设计	115
3.1.1 应用背景	115

3.1.2	项目目标	115
3.1.3	解决方案设计	116
3.1.4	系统模块功能分析	117
3.2	数据库设计	118
3.2.1	数据库需求分析	118
3.2.2	E-R 模型分析	119
3.2.3	数据表设计	120
3.3	技术细节	122
3.3.1	使用数据源(ODBC)访问数据	122
3.3.2	控件和方法	124
3.4	主窗体模块	124
3.4.1	窗体界面设计	124
3.4.2	添加主窗体初始代码	126
3.5	登录与系统设置模块	130
3.6	工资管理模块	130
3.6.1	添加考勤记录	130
3.6.2	修改考勤记录	133
3.6.3	查询考勤记录	136
3.6.4	计算工资	140
3.6.5	添加工资记录	144
3.6.6	修改工资记录	147
3.6.7	工资查询	150
3.6.8	删除工资记录	154
3.6.9	月底结算	156
3.7	固定资产模块	159
3.7.1	添加固定资产记录	159
3.7.2	修改固定资产记录	162
3.7.3	固定资产查询	165
3.7.4	固定资产记录删除	169
3.8	流动资产模块	170
3.8.1	添加报销记录	170
3.8.2	查询报销记录	174
3.8.3	导出报销记录	178
3.9	扩充和提高	181
3.10	小结	182

第4章 供应商与客户管理系统

4.1	系统总体设计	183
-----	--------------	-----

4.1.1	应用背景	183
4.1.2	项目目标	183
4.1.3	解决方案设计	184
4.1.4	系统模块功能分析	184
4.2	数据库设计	186
4.2.1	数据库需求分析	186
4.2.2	E-R 模型分析	187
4.2.3	数据表设计	188
4.3	技术细节	191
4.3.1	常用错误处理	191
4.3.2	ADO Data 控件	192
4.4	主窗体模块	192
4.4.1	窗体界面设计	192
4.4.2	添加主窗体初始代码	193
4.5	登录与系统设置模块	197
4.6	供应商信息模块	198
4.6.1	供应商信息添加	198
4.6.2	供应商信息修改	203
4.6.3	添加删除供货类别	206
4.6.4	供应商信息查询	210
4.6.5	供应商信息删除	213
4.7	销售客户信息模块	216
4.7.1	客户信息添加	216
4.7.2	客户信息修改	222
4.7.4	客户活动记录	226
4.7.5	客户信息查询	231
4.7.6	客户信息删除	236
4.8	客户服务模块	238
4.8.1	问题信息录入	238
4.8.2	问题信息查询	241
4.8.3	问题信息更改	245
4.9	扩充和提高	248
4.10	小结	249

第5章 企业事务管理系统

5.1	系统总体设计	250
5.1.1	应用背景	250
5.1.2	项目目标	250
5.1.3	解决方案设计	250

5.1.4 系统模块功能分析	251
5.2 数据库设计	254
5.2.1 数据库需求分析	254
5.2.2 E-R 模型分析	255
5.2.3 数据表设计	255
5.3 技术细节	257
5.3.1 在一个窗体上实现多个功能 界面	257
5.3.2 引用、部件和使用的控件	258
5.4 主窗体模块	258
5.4.1 主窗体界面设计	258
5.4.2 添加主窗体初始代码	260
5.5 登录模块	262
5.6 会议管理模块	265
5.6.1 建立模块界面	265
5.6.2 增加会议记录	268
5.6.3 会议记录查询	271
5.6.4 删除会议记录	274
5.7 值班管理模块	275
5.7.1 建立模块界面	275
5.7.2 值班资料录入	277
5.7.3 值班记录查询	280
5.7.4 修改值班记录	283
5.7.5 值班记录删除	283
5.8 车辆管理模块	284
5.8.1 建立模块界面	284
5.8.2 填写购车记录	288
5.8.3 修改购车记录	290
5.8.4 删除购车记录	292
5.8.5 出车登记	292
5.8.6 修改出车记录	294
5.8.8 删除出车记录	296
5.8.9 填写车辆维修记录	296
5.8.10 修改车辆维修记录	298
5.8.11 删除车辆维修记录	300
5.9 扩充和提高	300
5.10 小结	301

第6章 采购管理系统

6.1 系统总体设计	302
6.1.1 应用背景	302
6.1.2 项目目标	302
6.1.3 解决方案设计	303
6.1.4 系统模块功能分析	303
6.2 数据库设计	306
6.2.1 数据库需求分析	306
6.2.2 E-R 模型分析	307
6.2.3 数据表设计	308
6.3 技术细节	311
6.3.1 使用 SQL Server 2000 建立 数据库	311
6.3.2 部件和引用	314
6.4 主窗体模块	314
6.4.1 窗体界面设计	314
6.4.2 添加主窗体初始代码	315
6.5 合同单录入模块	320
6.5.1 选择合同单窗体	320
6.5.2 合同单录入窗体	329
6.5.3 供应商列表窗体	338
6.5.4 物料编号查询	341
6.6 合同单明细查询模块	343
6.7 合同单状态维护模块	346
6.7.1 选择采购员窗体	346
6.7.2 合同单状态维护窗体	348
6.7.3 合同单状态维护窗体	353
6.8 合同单执行完毕维护模块	358
6.9 扩充和提高	365
6.10 小结	365

第7章 销售管理系统

7.1 系统总体设计	366
7.1.1 应用背景	366
7.1.2 项目目标	366
7.1.3 解决方案设计	367

7.1.4 系统模块功能分析	367
7.2 数据库设计	370
7.2.1 数据库需求分析	370
7.2.2 E-R 模型分析	371
7.2.3 数据表设计	371
7.3 技术细节	374
7.4 主窗体模块	375
7.4.1 窗体界面设计	375
7.4.2 添加主窗体初始代码	377
7.5 登录与系统设置模块	380
7.6 销售员定义模块	381
7.6.1 增加销售员	381
7.6.2 修改销售员	384
7.6.3 查询销售员	388
7.6.4 删除销售员	391
7.7 订单模块	392
7.7.1 添加订单	392
7.7.2 修改订单	395
7.7.3 订单删除	397
7.7.4 查询销售单	398
7.7.5 签订销售单	400
7.7.6 删除销售单	402
7.8 广告登记表模块	403
7.8.1 填写广告登记表	403
7.8.2 广告登记表查询	405
7.8.3 广告登记表修改	407
7.8.4 删除广告登记	409
7.9 销售业绩查询模块	410
7.9.1 视图查询方式	410
7.9.2 表格查询方式	412
7.10 扩充和提高	415
7.11 小结	415

第8章 库存管理系统

8.1 系统总体设计	417
8.1.1 应用背景	417
8.1.2 项目目标	417

8.1.3 解决方案设计	418
8.1.4 系统模块功能分析	419
8.2 数据库设计	420
8.2.1 数据库需求分析	420
8.2.2 E-R 模型分析	422
8.2.3 数据表设计	422
8.3 技术细节	424
8.3.1 使用 Microsoft 数据报表设计器 书写报表	424
8.3.2 DataReport 对象	425
8.3.3 类模块和数据识别类	425
8.3.4 部件和引用	426
8.4 主窗体模块	426
8.4.1 窗体界面设计	426
8.4.2 添加主窗体初始代码	427
8.5 登录模块	429
8.6 仓管员定义模块	429
8.6.1 仓管员定义	429
8.6.2 仓库定义	441
8.7 进库信息处理模块	443
8.7.1 进库单登记	443
8.7.2 进库明细管理	451
8.7.3 进库单查询	456
8.8 出库信息处理模块	460
8.8.1 出库单登记	460
8.8.2 出库明细管理	461
8.8.3 出库单查询	462
8.9 报表输出模块	463
8.10 扩充和提高	470
8.11 小结	471

第9章 企业管理系统打包与发布

9.1 打包和展开向导简介	472
9.2 使用打包和展开向导	472
9.3 安装和运行发布的系统	479
9.4 小结	481

第 1 章 电子商务系统开发基础

本书使用 Visual Basic 中文版, 在 Windows 2000 professional 版上进行开发, 使用的数据库有 Microsoft Access 2000, SQL Server 2000 等。在 VB 编程时主要使用 ADO 数据访问技术以及 SQL 语句进行。在设计电子商务系统时, 本书采用以下结构:

(1) 系统总体设计

系统总体设计是本书每个系统设计的第一步, 主要包括四个方面: 应用背景介绍, 说明了一个企业电子商务系统的应用来历、原因、重要性和工程背景等。项目目标, 指出本书在设计此系统时所要求达到的要求, 一般是对社会中企业的实际需求进行分析后得来的。解决方案设计, 就是对需求、目标的实现方法, 本书采用图示等方法说明一个系统实现的主要几大块内容。系统模块功能分析, 即前面说的功能模块划分, 这里确定主要的功能模块, 并按照分出的几大功能模块作为以后的窗体设计依据。

(2) 数据库设计

本书采用 Visual Basic 作为编程开发工具, 而 VB 强大的数据库开发功能使得相应的可应用的数据库种类繁多、复杂程度各有不同, 所以做好数据库的设计是本书中的一个重点。而且对于任何一个基于数据库的企业办公自动化系统而言, 数据库设计也是必不可少的。本书在数据库设计中主要考虑一个系统对数据库的需求, 绘制数据流程图、数据库的 E-R 实体关系图, 在具体设计数据库中的每张数据表, 确定表中字段、属性、主键、表键关系等。

(3) 技术细节说明

这一部分主要说明在 Visual Basic 进行系统开发时候可能用到的数据库技术、控件技术, 以及其他第三方支持技术, 或者是其他一些注意事项。

(4) 模块具体设计

最后是按照前面划分的功能模块组织各个窗体, 进行具体的系统程序设计工作。包括: 编写代码、连接数据库、调试通过、发布程序、扩充与提高等。

1.1 Visual Basic 进行数据库开发的特点

1.1.1 Visual Basic 开发数据库的优势

3.0 版以后, 微软公司把许多更新的数据访问功能加到了 Microsoft Visual Basic 中来, 使之具有强大的数据库连接和数据处理功能。与普通的数据库系统相比, 用 Visual Basic 作为数据库开发平台有以下优点:

- 简便容易。Visual Basic 提供了数据控件, 利用该控件, 用户只要编写少量的代码甚至不编写任何代码就可以访问数据库, 对数据库进行浏览。
- 灵活多样。Visual Basic 具有强大的数据库开发语言, 可以根据编程者的需要随意编写代码对数据库进行的各种操作, 远远超出了一般的数据库管理软件 (如 Access) 局限于特定结构只能进行特定操作的方式。另外, Visual Basic 可以和多种数据库连接, 也就是说, 目前较为流行的大多数数据库都可以与 Visual Basic 连接, 因而可以用 Visual Basic 开发相应的应

用程序。

- 不断发展完善。Visual Basic 同时又是一种可以扩充的语言，特别是在数据库应用方面的扩充。在 Visual Basic 中，可以使用 ActiveX 控件（也有称为 VBX 或 OLE 控件），这些控件可以由 Microsoft 公司提供，也可以由第三方开发者提供。有了这些控件，可以很容易地在 Visual Basic 中增加新功能，扩充 Visual Basic 数据存取控制的指令系统。在许多其他数据库开发平台中，还不具备这样功能的 ActiveX 控件。

Visual Basic 通过多种方式与各种数据库进行连接，主要有三种方式：使用 Access 数据库引擎连接数据、Microsoft ODBC 数据连接以及第三方 ODBC 数据连接。

用 Access 数据库引擎连接数据：是 VB 程序员常用的数据库连接方式。Access 是 Visual Basic 的“内部数据库”，即在 Visual Basic 中可以直接建立 Access 数据库。同时，通过 Access 数据库引擎，还可以使用 Btrieve、dBASE、FoxPro 和 Paradox 等数据库。

Microsoft ODBC 数据连接：也是我们常用的数据库连接方式，它可以让我们连接到很多常用的数据库中的数据上去。通过 Microsoft ODBC 驱动程序，我们可以使用 Microsoft SQL Server、Oracle、Sybase、Excel、Text 等的数据库。

第三方 ODBC 数据连接：通过第三方 ODBC 驱动程序，我们能够连接到更多更新的数据库，虽然有些数据库并不常见，但这是 Visual Basic 继续发展的一个窗口。

1.1.2 Visual Basic 提供的数据库访问技术

在 Visual Basic 中进行数据访问使用三种方法：ActiveX 数据对象(ADO)、远程数据对象(RDO) 和数据访问对象(DAO)。为什么在 Visual Basic 中有三种数据访问接口呢？因为数据访问技术总是在不断进步，而这三种接口的每一种都分别代表了该技术的不同发展阶段。最新的是 ADO，它是比 RDO 和 DAO 更简单，更加灵活的对象模型。对于新工程，应该使用 ADO 作为数据访问接口。

那么，为什么使用 ADO？ADO 是为 Microsoft 最新和最强大的数据访问范例 OLE DB 而设计的，是一个便于使用的应用程序层接口。OLE DB 为任何数据源提供了高性能的访问，这些数据源包括关系和非关系数据库、电子邮件和文件系统、文本和图形、自定义业务对象等。ADO 提供了轻量、高性能的接口，同时还使用了与 DAO 和 RDO 相似的约定和特性，简化的语言使它更易于学习。总之，ADO 的功能异常的强大，有关 ADO 的详细信息，后面会有详细介绍。另外，为了向后兼容性，对于现存的工程，Visual Basic 将继续支持 DAO 和 RDO。

1.2 用 SQL 访问数据库

1.2.1 SQL 语言介绍

SQL 代表 Structured Query Language（结构化查询语言）。它于 20 世纪 70 年代由 IBM 开发，当时是给计算机用户提供的一种从不同数据格式中选取数据的标准方法。其意图是建立一种不依赖于目前任何编程语言、但又可为所有编程语言用来更新和查询数据库信息的语言。

SQL 是一种强大的操作语言，它在 Visual Basic 和 Microsoft Access Jet 数据库引擎中被作为在数据库中存取数据的基本方法。SQL 语句可分为两大类：数据操作语言语句（DML）和数据定义语言语句（DDL）。DDL 语句用于定义数据表、索引和数据库关系。而 DML 语句则被用于对数据表中存储的信息进行选取、排列、求和和计算。

注意：SQL 的发音是“ess-que-ell”而非“sequel”，这种混淆来自于数据库语言的起源。SQL 语言是 60 年代以来 IBM 开发的一种叫 Sequel 语言的继续。因此，很多人（尤其是那些熟知 IBM Sequel 语言的人）继续用该发音，误读这种新数据库语言的名称。

SQL 语句仅仅是一种语句。每一语句对一个或多个数据库对象（表、列、索引等）进行操作。大多数 SQL 语句返回结果的形式是数据记录集合，通常又被称为“视图”（view）。SQL 并不是一种非常友好的语言，很多程序用“指向并单击”对话框、按例查询网格和其他对用户友好的界面隐藏了这些 SQL 语句，然而只要存取的数据存放在关系数据库中，就要用到 SQL 语句，不管是有意的还是无意的。

SQL 的语法是由美国国家标准协会（ANSI）下属的一个委员会制定的。ANSI-SQL 委员会的成员是从事建立和改进用于促进计算机软件产业快速发展的标准的信息系统专家。尽管每一种编程语言和数据库界面都有它自己独特的 SQL 版本，但几乎所有人都同意遵守 ANSI-SQL 委员会制定的基本标准。最为广泛使用的 SQL 标准是 SQL-89。这一标准是在 1989 年首次发表的。三年后又发表了一个新的标准：SQL-92。想进一步了解 ANSI-SQL 标准的读者可以在其他书中找到更多资料。请相信，在这里所介绍的概念和技巧可以用于所遇到的几乎所有基于 SQL 的编程和查询工具。

1.2.2 SQL 语句创建数据库

这一小节是用 SQL 的一类关键字进行与创建数据库有关的操作，这些关键字可用于创建新数据表、改变原有表结构以及从数据库中删除原有表等。

（1）创建、删除表

使用 CREATE TABLE 语句创建自己的表，指定字段列名、关键字字段以及类型等。CREATE TABLE 语句的基本形式包括三部分：CREATE TABLE 子句，表名，新表中每一列的列名、列类型和列长度的列表。下面的例子创建了一个简单的名为 PERS 的表。

```
CREATE TABLE PERS ( Field1 Text (30), Field2 INTEGER )
```

这一 SQL 语句创建了一个包含两列的名为 PERS 的数据表。Field1 列是 20 字节长的 TEXT 列，Field2 是 Integer 列，注意没有指明 INTEGER 列的长度。Microsoft Access Jet SQL 语句只接受 TEXT 列的长度值，其他列的长度都已经被预定已过了。这方面的内容可参考相关书籍。

注意：关键字字段是可用来标识或存取特定行的一组列。由不止一列组成的关键字成为组合关键字。在具有组合关键字的表中，组合关键字中各列的排序不一定与这些列在表中的排序相对应。有时关键字字段又被称为“索引字段”或“索引”。

在创建数据表示也可以使用 PRIMARY KEY 命令。做法是在主关键字字段名后面跟一个 CONSTRAINT 子句。例如：

```
CREATE TABLE PERS (Field1 TEXT(30) CONSTRAINT PKPERS PRIMARY KEY,
                    Field2 INTEGER)
```

这样Field1字段就作为PERS表的主关键字字段创建出来了,稍后会介绍CONSTRAINT子句。现在,只需几个可以在一个CREATE TABLE语句中同时创建主关键字字段和外部关键字字段即可。

注意:一个数据表只能有一个主关键字,但外部关键字字段在其所在数据表中不是唯一。主关键字用来控制数据显示的顺序,外部关键字将本地字段跟其他(外部)表中的字段相连接。主关键字字段决定了数据表中记录的唯一性,数据表中不能有主关键字字段相同的两条记录。在一个数据表中,外部关键字字段可以有多个也可以没有。

删除表使用DROP TABLE语句。他常用于删除临时表,也可以作为表和表之间或数据库和数据库之间的数据复制过程的一部分。如下例,删除一个表TABLE2。

```
DROP TABLE TABLE2
```

(2) 添加、删除表中字段

使用ALTER TABLE语句可以向表中加入或删除字段。包括两种格式:ADD COLUMN格式,用于在一个现有的表中加入新列,同时不丢失原有列的任何数据。DROP COLUMN格式用于从现有表中删除列,这也同时不会丢失未受影响的列中的数据。如下例:

```
ALTER TABLE TABLE1 ADD COLUMN FIELD1 DATE
ALTER TABLE TABLE1 ADD COLUMN FIELD2 CURRENCY
ALTER TABLE TABLE2 DROP COLUMN FIELD3
```

以上前两条语句共向同一个表TABLE1中增加了两列字段。第一列字段名为FIELD1,类形是DATE,第二列字段名为FIELD2,类型是CURRENCY。ALTER TABLE语句每次只能处理一列信息。上面的第三条语句就是从表TABLE2中删除字段FIELD3。

ADD COLUMN子句总是在表中左边的列后面加入新列字段,可以用SELECT_FORM语句控制列的显示顺序,但表中字段的物理顺序,必须用CREATE_TABLE语句中设置字段来确定。

(3) 管理关键字字段

前面指出,关键字字段就是索引。现在可以用CREATE INDEX语句来在现有表中创建索引。基本形式如下例:

```
CREATE INDEX INDEX1 ON TABLE1 (FIELD1)
```

这样就在表TABLE1中创建了一个关键字字段,名为INDEX1。CREATE INDEX语句创建关键字字段用于加强数据表的数据完整性,比如进行排序等。其一些变体形式如下:

```
CREATE INDEX INDEX1 ON TABLE1 (FIELD1) WITH PRIMARY
```

创建一个主关键字字段,保证表中每一行在该字段中都有唯一一个数据值。关键字字段是不允许有NULL值(空数据)。

```
CREATE UNIQUE INDEX INDEX2 ON TABLE2 (FIELD2)
```

在指定字段上创建一个唯一关键字字段。在这个例子里,不得有任何两列的值相同,但NULL只是允许的。

```
CREATE INDEX INDEX3 ON TABLE3 (FIELD3) WITH IGNORE NULL
```

创建一个不唯一的关键字字段,在字段列中允许有NULL记录。

```
CREATE INDEX INDEX1 ON TABLE1 (FIELD1) WITH DISALLOW NULL
```

创建一个不唯一的关键字字段,但不允许出现NULL值。

在编程时我们常常使用一种关键字命名约定,这一约定被SQL程序员广泛使用。就是

在命名关键字字段的时候,所有主关键字都用字母 PK 开头(如 PKusername),所有的为排序创建的关键字字段都用字母 SK 开头(SKpaixu),所有请求唯一值的关键字字段名都用字母 UK 开头,所有定义了外部关键字关系的关键字都用字母 FK 开头,其他关键字可以用字母 IK 开头。养成良好的命名习惯和定期定段的检查错误的习惯,对于任何编程人员来说都是至关重要的。

(4) 排序语句

使用 ASC (升序) 和 DESC (降序) 的 SQL 关键字可以控制语句达到给字段内容排序的作用。例如在建索引时:

```
CREATE INDEX SKindex ON TABLE1 (FIELD1 DESC)
```

这样,在表中建立了一个名为 SKindex 的关键字字段,并且按照字段 FIELD1 中的数据的降序进行排序。如果不使用排序 SQL 关键字,SQL 默认的是用升序排列。如上面语句如果去掉 DESC,则会按照 FIELD1 字段数据的升序给 SKindex 字段排序。如果想控制多列关键字字段的索引顺序,可以使用下面语句:

```
CREATE INDEX SKusername ON TABLE1 (FIELD1 ASC, TABLE2 DESC)
```

(5) 使用 CONSTRAINT 管理关系

从 SQL 语句的立场来看,CONSTRAINT 字句跟关键字完全一样。CONSTRAINT 用于创建增强数据库的逐句完整性的关键字,必须跟 CREATE TABLE 或 ALTER TABLE 语句一起使用。有以下几种形式:

注意: Microsoft Access 的 SQL 语法不允许在 CONSTRAINT 字句内使用 IGNORE NULL 或 DISALLOW NULL 关键字,所以,如果想创建包含这些关键字的数据完整性索引时,只能用 CREATE INDEX 来定义。

● PRIMARY KEY (主关键字)

这是较长用的语句,用于定义包含了表中主关键字的字段或字段集。如下例创建了一个数据库表,其中包含有一个主关键字字段 PKindex1 和一个 INTEGER 类型字段 name。

```
CREATE TABLE TABLE1 (index1 TEXT (255) CONSTRAINT PKindex1 PRIMARY KEY,  
name INTEGER)
```

● UNIQUE (唯一关键字)

在默认的情况下,用 UNIQUE 字句创建的关键字字段允许在相应的列中输入 NULL。不过将数据输入列时,它必须是唯一的值,否则数据库引擎将返回一个错误消息。这同在 CREATE INDEX 语句中使用 IGNORE NULL 关键字的情形一样,在缺省情况下,用 UNIQUE 创建的所有关键字都是 IGNORE NULL 关键字。

```
CREATE TABLE TABLE1 (index1 TEXT (255) CONSTRAINT UKindex1 UNIQUE,  
name INTEGER)
```

可以在多个字段中使用 UNIQUE 字句,这在决定数据表中多个字段的唯一性时非常方便。以下保证了 index1、index2、index3 三个字段的组合总是唯一的。

```
CREATE TABLE TABLE1 (index1 TEXT (255), index2 TEXT(20), index3 TEXT(100),  
CNSTRAINT UKindex1 UNIQUE (index1, index2, index3))
```

注意: 上例中的 UKindex1 只是代表的三个关键字字段的组合,并不是真正出现在数据表中的字段名。

● FOREIGN (外部关键字)

这是 CONSTRAINT 最强大的格式,用在表间建立关系。通常一个 FOREIGN 关系建立在一个含有有效列输入值列表的小型表(又称输入确认表)和另外一个表之间。第二个表

通常有一列被赋予了跟输入确认表中的主关键字列相同的名称。通过在上述两个文件之间建立外部关键字关系，能够实行一项数据规则：某一表中的唯一有效输入必须存在于输入确认表的主关键字列中的值。如下例：

```
CREATE TABLE PERS ( Field1 Text (30) CONSTRAINT PKField1 PRIMARY KEY,
    Field2 INTEGER );
CREATE TABLE TEMP ( TEST1 TEXT(30),
    Field1 TEXT(30) CONSTRAINT FKField1 REFERENCES PERS(Field1))
```

只要所参照的列定义了被参照表的 **PRIMARY** 字段，就可以省去 **CONSTRAINT** 子句中 **REFERENCES** 的字段名（如上面的 Field1），但更好的编程习惯是加入字段名使语句更明了。对于多个外部关键字关系的建立关系如下：

```
CREATE TABLE PERS ( FIELD1 TEXT(30), FIELD2 TEXT(20), FIELD3 TEXT(10),
    CONSTRAINT PKFIELD1 PRIMARY KEY(FIELD1, FIELD2, FIELD3), XYZ INTEGER );
CREATE TABLE TEMP ( TEST1 TEXT(30), FIELD1 TEXT(30), FIELD2 TEXT(20),
    FIELD3 TEXT(10), CONSTRAINT PKFIELD2 FOREIGN KEY(FIELD1, FIELD2, FIELD3))
```

增加多列外部关键字关系的语法跟建立单列外部关键字关系的语法是不同的。在创建多列外部关键字关系时，必须使用 **FOREIGN** 关键字。另外，括号中字段的顺序必须跟参照表中的参照关键字顺序一致。

注意：可以使用 **ALTER TABLE** 语句增加或删除现有的数据表中的约束关系，但要十分谨慎。因为，尽管 **SQL** 允许这样做，但如果目标表中一有数据，这样做常会导致数据完整性错误。建议只在用 **CREATE TABLE** 语句中创建表时建立约束。

1.2.3 SQL 语句选取数据库

本节介绍如何编写在现有数据表中选取数据的基本 **SQL** 语句。这是对数据库中数据的基本操作方法，在各种编程语言调用 **SQL** 语句进行编程时都大量使用到了。

（1）选取数据记录

最常用的 **SQL** 语句莫过于 **SELECT_FROM** 记录选取语句了。**SELECT_FROM** 语句用于从数据库的一个或多个表中选取记录，返回的结果是一个视图，该视图是原数据的一个子集。在 **Visual Basic** 中，视图可以以记录集、表、动态集或瞬像的形式返回，这些名词的具体解释可参看相关书籍。

最简单的 **SELECT_FROM** 语句包括两部分：

- 选取表的一个或多个列
- 选取所请求列的一个或多个表的清单

SQL 语句的基本形式如下：

```
SELECT 字段名 FROM 表名
```

SELECT 语句执行结果会返回一个包含了指定表中的指定字段名的数据对象。不论表中有多少条记录，都可以只用一条 **SELECT_FROM** 语句得到完整的结果集。这相当方便，但也相当危险，如果 **SELECT_FROM** 语句的结果集中含有太多记录，有可能使网络变慢、耗尽本地工作站内存、最后死机。在本节中会讲到使用 **WHERE** 子句将视图大小限制为只包含所需的记录。

为了返回表中所有列，可以在 **SELECT** 语句中列出每一列的名称，这在表中只有少数列时才管用。如果表中有很多列，这一工作将变得十分枯燥。这时，可以用星号（*）来代

替列名在结果集中自动列出表中的所有列。这在需要显示一个数据表而不知道其中所有列的名称时非常有用。如下例：

```
SELECT * FROM TABLE1
```

可见只要知道了表名就可以用上面的方法显示出整个表来。另外，当列出所要显示的字段名的时候，在 `SELECT_FROM` 语句中出现的各字段的顺序决定了它们在结果集中的显示顺序。例如：

```
SELECT FIELD1, FIELD2 FROM TABLE1
```

(2) 排序选择

在使用 `SELECT_FROM` 语句的时候，结果集中的记录是根据他们在基础表中被找到的顺序返回的。但如果用 `ORDER BY` 子句，可以按一定的顺序来显示选择的结果。这里，我们可以使用到上面提到的 `ASC`（升序）和 `DESC`（降序）关键字。例如：

```
SELECT * FROM TABLE1 ORDER BY FIELD1 ASC
```

这一语句执行结果会选取出 `TABLE1` 中的所有字段，但在选取时会按照 `FIELD1` 字段中数据值按升序来给所有记录排序。另外，可以在 `ORDER BY` 子句中输入一个以上子段，`SQL` 降序创建一个返回 `ORDER BY` 子句综合顺序的结果集。多个字段间用逗号隔开。

```
SELECT FIELD1, FIELD2, FIELD3 FROM TABLE1 ORDER BY FIELD1 ASC, FIELD2 DESC
```

上句执行结果会从表 `TABLE1` 中选出 `FIELD1`、`FIELD2`、`FIELD3` 三个子段，对选出的所有记录先按照 `FIELD1` 中记录的升序进行大排序，遇到 `FIELD1` 中有相同数据的时候，就要对这几个相同数据的记录按照 `FIELD2` 字段中的数据进行降序小排列，从而完成最后结果集的选取。

(3) 条件选择

`SELECT_FROM` 语句最常用的条件选择子句就是 `WHERE` 关键字，`WHERE` 关键字与 `SELECT` 语句一起使用能大大限制结果集的内容。它有两种作用：

- 限制单个结果集的内容。
- 将两个或多个表连接到一个结果集。

WHERE 关键字基本用法：

```
SELECT 字段名 FROM 表名 WHERE 查询条件
```

其中的查询条件最简单的形式如下例：

```
WHERE COLUMN = VALUE
```

其中 `COLUMN` 代表了所请求数据表中的列的名称，`VALUE` 代表了一个数值常量，它应该是属于 `COLUMN` 字段中的数据类型。例如：

```
SELECT NAME, FIELD2 FROM TABLE1 WHERE NAME='ZJH'
```

注意：这里用单引号（'）在 `SQL` 语句中将数值常量括起来，但 `Visual Basic SQL` 语句中也接受双引号。但有些数据类型，例如 `TEXT` 型数据在 `VB` 编程中应使用单引号括起来。

上例中返回两个字段的记录集内容，但不包括它们所有的记录行，只包含有符合 `NAME` 字段内容为 `ZJH` 的记录。另外可以用 `AND` 和 `OR` 操作符来将合法的逻辑比较连接成一个 `WHERE` 子句，也可以用 `=`、`<>`、`>`、`<`、`>=`、`<=` 等进行逻辑比较，如：

```
SELECT NAME, NUMBER, TABLE1 FROM TABLE1  
WHERE NAME = 'ZJH' AND NUMBER <> 150
```

此外还有 `BETWEEN_AND`、`IN` 和 `LIKE` 几种比较方法。如下例，结果中只包含了 `NUMBER` 字段值在 20 和 149 之间的所有记录行，且选择三个字段形成记录集。

```
SELECT NUMBER, FIELD1, FIELD2 FROM TABLE2
WHERE NUMBER BETWEEN 20 AND 149
```

如果只想选择出 **NUMBER** 值为 10、20、30 的关于三个字段的记录时，就可以使用 **IN** 关键字后跟想要的值，这些值之间用逗号隔开，外面再用括号或起来。如：

```
SELECT NUMBER, FIELD1, FIELD2 FROM TABLE2
WHERE NUMBER IN (10, 20, 30)
```

还可以使用 **LIKE** 函数来返回列内容跟传送函数的常量相似的所有行，**LIKE** 函数是一个非常强大的工具。例如，想返回 **NAME** 字段中所有含有字母 A（不管在什么位置）的行，可用以下语句：

```
SELECT NAME, TABLE1, TABLE2 FROM TABLE1
WHERE NAME LIKE ('*A*')
```

另外，我们还可以用 **WHERE** 子句来比较来自不同表的字段。为此，可以建立把两个或多个表格连接到一个结果集中的标准。这种 **WHERE** 子句的语法是：

```
SELECT TABLE1.COLUMN1, TABLE2.COLUMN1 FROM TABLE1, TABLE2
WHERE TABLE1.COLUMN1 = TABLE2.COLUMN1
```

TABLE1 和 **TABLE2** 是同一数据库中不同的数据表。**COLUMN1** 代表了每一个表中的一个列。扩展一下，还可以用不同表中不同的字段进行选择。也就是说，只要某列存在于基础表中，就可以将其用于语句的 **WHERE** 部分，而不一定要把它放入语句的 **SELECT** 部分。

```
SELECT TABLE1.COLUMN1, TABLE2.COLUMN2 FROM TABLE1, TABLE2
WHERE TABLE1.COLUMN3 = TABLE2.COLUMN4
```

注意：这里列出的是两个不同表中的字段，在从多个表中选取列时，最好是在每个字段名前面加上表名并用句点 (.) 连接。如果这一字段名在所选表的所有字段中是唯一的，**SQL** 并不要求使用“表名.字段名”这样的语法。但这样做是一个很好的习惯。

SQL 直接把列名赋予结果集，有时所赋名不易理解或不完整，这是可以用 **AS** 关键字来给结果集中的列重新命名。下面的 **SQL** 语句是如何在 **SELECT** 语句中用 **SD** 关键字来给结果集的字段名，这种重命名操作并不影响基础表的原始字段名。

```
SELECT TABLE1.OLDFIELD1 AS NEWFIELD1,
TABLE2.OLDFIELD2 AS NEWFIELD2,
TABLE3.OLDFIELD3 AS NEWFIELD3
WHERE TABLE1.OLDFIELD1 = TABLE2.OLDFIELD2
AND TABLE3.OLDFIELD3 = TABLE1.OLDFIELD1
```

(4) SQL 集合函数

SQL 标准定义了一组在所有符合 **SQL** 标准的系统中都必须提供的核心函数，这些函数就是集合函数 (aggregate function)。集合函数用来迅速返回列中的数字数据的计算结果。在 Microsoft Access Jet 数据库引擎中可使用的 **SQL** 集合函数有：

- **AVG**：返回某一列中所有数值的平均值。
- **COUNT**：返回列数并经常被用于获取某一视图中的总行数。**COUNT** 是唯一一个能用于非数值列的标准 **SQL** 集合函数。
- **SUM**：返回某一列中所有数值的总和。
- **MAX**：返回某一列中的最大数值。
- **MIN**：返回某一列中的最小数值。

下面语句说明的是上面五个 **SQL** 集合函数的用法。它们都是使用在 **SELECT** 语句的基