第 7 章　SQL Server 2005 基础

信息管理技术和信息管理应用系统的发展，为数据库理论和数据库系统的发展提供了强大的推动力，不论是基于 C/S 模式还是基于 B/S 模式信息系统的开发都离不开数据库系统，不论是 Visual FoxPro 等本地数据库，还是 SQL Server 等远程数据库，都已经被广泛应用于人们的学习和工作中。而 SQL Server 2005 作为一个典型的数据库管理系统也受到了越来越多的软件开发人员和商业用户的青睐，从而被越来越多地使用。本章将介绍 SQL Server 2005 的基础知识。

7.1　SQL Server 2005 概述

要正确地选择和安装合适的 SQL Server 2005 以及使用 SQL Server 2005 所提供的基本功能就要对其不同版本以及软、硬件需求等信息有深刻的认识和理解。

7.1.1　SQL Server 2005 的版本

根据应用程序的需要，安装要求可能有很大不同。SQL Server 2005 的不同版本能够满足企业和个人独特的性能以及价格要求。下面介绍如何在 SQL Server 2005 的不同版本中作出最佳选择。

1）SQL Server 2005 Enterprise Edition（适用于 32 位和 64 位）

Enterprise Edition 支持超大型企业进行联机事务处理（OLTP）、具有高度复杂的数据分析、数据仓库系统，达到了网站所需的性能水平。Enterprise Edition 的全面商业智能和分析能力及其高可用性功能（如故障转移群集），使它可以处理大多数关键业务的企业工作。Enterprise Edition 的功能是所有版本中最全面的，是超大型企业的理想选择，能够满足复杂的要求。该版本还推出了一种适用于 32 位或 64 位平台的 120 天评估版。

2）SQL Server 2005 Standard Edition（适用于 32 位和 64 位）

Standard Edition 包括电子商务、数据仓库和业务流解决方案所需的基本功能。Standard Edition 的集成商业智能和高可用性可以为企业提供支持其运营所需的基本功能。Standard Edition 是需要全面的数据管理和分析平台的中小型企业的理想选择。

3）SQL Server 2005 Workgroup Edition（仅适用于 32 位）

对于那些对数据库的大小和用户数量没有限制的小型企业，Workgroup Edition 是理想的数据管理解决方案之一。Workgroup Edition 可以作为前端 Web 服务器，也可以用于部门或分支机构的运营。它包括 SQL Server 产品系列的核心数据库功能，并且可以轻松地升级至 Standard Edition 或 Enterprise Edition。Workgroup Edition 是理想的入门级数据库，具有可靠、功能强大且易于管理的特点。

4)SQL Server 2005 Developer Edition(适用于 32 位和 64 位)

Developer Edition 使开发人员可以在 SQL Server 上生成任何类型的应用程序。它包括 SQL Server 2005 Enterprise Edition 的所有功能,但有许可限制,只能用于开发和测试系统,而不能作为生产服务器。Developer Edition 是独立软件供应商(ISV)、咨询人员、系统集成商、解决方案供应商以及创建和测试应用程序的企业开发人员的理想选择。Developer Edition 可以根据生产需要升级至 Enterprise Edition,本书如果不加特殊说明则是基于开发版本的 SQL Server 2005。

5)SQL Server 2005 Express Edition(仅适用于 32 位)

Express Edition 是一个免费、易用且便于管理的数据库。它与 Microsoft Visual Studio 2005 集成在一起,可以轻松开发功能丰富、存储安全、可快速部署的数据驱动应用程序。SQL Server 2005 Express Edition 可以再分发(受制于协议),还可以起到客户端数据库以及基本服务器数据库的作用。SQL Server 2005 Express Edition 是低端 ISV、低端服务器用户、创建 Web 应用程序的非专业开发人员以及创建客户端应用程序的编程爱好者的理想选择。

7.1.2 SQL Server 2005 的环境需求

选择了合适的 SQL Server 2005 版本之后,就要进行环境需求分析以确定所选择的版本能否在当前的环境下进行安装。安装、运行 SQL Server 2005 的硬件和软件需求如下(由于在 32 位平台上运行 SQL Server 2005 的要求与 64 位平台上的要求不同,本书以 32 位机为例来说明)。

1.硬件需求

- 显示器:VGA 或分辨率至少在 1 024×768 像素之上的显示器。
- 定点设备:需要 Microsoft 鼠标或兼容的定点设备。
- 驱动器:CD 或者 DVD 驱动器。
- 处理器、内存:SQL Server 2005 不同的版本对处理器类型、速度及内存的需求是不同的,如表 7-1 所示。
- 硬盘空间:实际的硬盘空间要求取决于系统的配置以及安装时所选择的 SQL Server 2005 服务和组件,具体要求如表 7-2 所示。

表 7-1 SQL Server 2005 不同的版本对处理器和内存的要求

SQL Server 2005 版本	处理器类型①	处理器速度②	内 存③
SQL Server 2005 Enterprise Edition④ SQL Server 2005 Developer Edition SQL Server 2005 Standard Edition SQL Server 2005 Workgroup Edition	Pentium Ⅲ或更高速度的处理器	最低:600 MHz 建议:1 GHz 或更高	最小:512 MB 建议:1 GB 或更大
SQL Server 2005 Express Edition	Pentium Ⅲ或更高速度的处理器	最低:500 MHz 建议:1 GHz 或更高	最小:192 MB 建议:512 MB 或更大

注:

① 如果不满足处理器类型的要求,系统配置检查器(SCC)将阻止安装程序运行。

② 如果不满足最低或建议的处理器速度要求，SCC 将向用户发出警告但不会阻止安装程序运行。多处理器计算机上将不会出现警告。

③ 如果不满足最低或建议的内存要求，SCC 将向用户发出警告但不会阻止安装程序运行。内存要求仅针对此版本，它不反映操作系统的其他内存要求。SCC 将在安装开始时确认内存是否可用。

④ SQL Server 2005 Evaluation Edition 支持与 SQL Server 2005 Enterprise Edition 相同的功能集。

表 7-2　SQL Server 2005 的各组件对硬盘空间的要求

组　　件	硬盘空间要求
数据库引擎和数据文件复制以及全文搜索	150 MB
Analysis Services 和数据文件	35 KB
Reporting Services 和报表管理器	40 MB
Notification Services 引擎组件、客户端组件和规则组件	5 MB
Integration Services	9 MB
客户端组件	12 MB
管理工具	70 MB
开发工具	20 MB
SQL Server 联机丛书和 SQL Server Mobile 联机丛书	15 MB
示例和示例数据库	390 MB

2. 软件需求

- 浏览器软件：所有 SQL Server 2005 的安装都需要 Microsoft Internet Explorer 6.0 SP1 或更高版本的支持，因为 Microsoft 管理控制台(MMC)和 HTML 帮助需要它。Internet Explorer 的最小安装即可满足要求，且不要求 Internet Explorer 是默认浏览器。然而，如果只安装客户端组件且不需要连接到要求加密的服务器，则 Internet Explorer 4.01(Service Pack 2)即可满足要求。
- Internet 信息服务(IIS)：安装 Microsoft SQL Server 2005 Reporting Services (SSRS，报表服务)需要 IIS 5.0 或更高版本。
- ASP.NET 2.0：当安装报表服务时，SQL Server 2005 安装程序会检查 ASP.NET 是否已安装到本机上。
- 还需要安装.NET Framework 2.0、Microsoft SQL Server 本机客户端、Microsoft SQL Server 安装程序支持文件。

表 7-3 列出了常见的操作系统所支持运行的 SQL Server 2005 版本。

表 7-3　不同操作系统所支持的 SQL Server 2005 版本

版　本 / 操作系统	Enterprise Edition	Developer Edition	Standard Edition	Workgroup Edition	Express Edition
Windows 2000	否	否	否	否	否
Windows 2000 Professional Edition SP4	否	是	是	是	是
Windows 2000 Server SP4	是	是	是	是	是

续表

操作系统 \ 版本	Enterprise Edition	Developer Edition	Standard Edition	Workgroup Edition	Express Edition
Windows 2000 Advanced Server SP4	是	是	是	是	是
Windows XP Home Edition SP2	否	是	否	否	是
Windows XP Professional Edition SP2	否	是	是	是	是
Windows Server 2003 SP1	是	是	是	是	是
Windows Server 2003 Enterprise Edition SP1	是	是	是	是	是

7.2 SQL Server 2005 的安装与配置

在选择好合适的 SQL Server 2005 版本以及运行环境后就可以进行软件的安装了，下面介绍安装 SQL Server 2005 的安全准备以及具体的安装过程。

7.2.1 SQL Server 2005 安装前的安全准备

安全很重要，这不只是对 Microsoft SQL Server 和 Microsoft 而言的，而是对每个产品和每家企业都很重要。遵循最佳的做法，大多数安全漏洞都可以避免。而要保证 SQL Server 2005 的安全性，则在设置服务器环境时应遵循以下最佳做法。

1)增强物理安全性

物理和逻辑隔离是构成 SQL Server 安全的基础。可通过下列方式增强 SQL Server 安装的物理安全性：

- 将服务器置于专门的房间，未经授权的人员不得入内。
- 将数据库的宿主计算机置于受物理保护的场所，最好是上锁的机房，房中配备水灾检测和火灾检测监视系统或灭火系统。
- 将数据库安装在公司 Intranet 的安全区域中，任何时候都不要直接连接到 Internet。
- 定期备份所有数据，并将副本存储在远离工作现场的安全位置。

2)使用防火墙

防火墙是保护 SQL Server 安装所不可或缺的。若要使防火墙发挥最佳效用，应遵循以下原则：

- 在服务器和 Internet 之间放置防火墙。
- 将网络分成若干安全区域，区域之间用防火墙分隔。先阻塞所有通信流量，然后有选择地只接受所需的通信。
- 在边界防火墙上，始终阻塞发往 TCP 端口 1433(由默认实例监视)的数据包和发往 UDP 端口 1434(由计算机上的实例之一监视)的数据包。如果命名实例正在侦听其他端口，则也阻塞那些端口。
- 在多层环境中，使用多个防火墙创建屏蔽子网。

- 如果在 Windows 域内部安装服务器，则将内部防火墙配置为允许 Windows 身份验证。
- 打开 Kerberos 或 NTLM 身份验证所使用的端口。
- 如果应用程序使用分布式事务处理，必须要将防火墙配置为允许 Microsoft 分布式事务处理协调器(MS DTC)在不同的 MS DTC 实例之间以及在 MS DTC 和资源管理器(如 SQL Server)之间进行通信。

3)隔离服务

隔离服务可以降低风险，防止已受到危害的服务危及其他服务。若要隔离服务，应遵循以下原则：

- 任何时候都不要在域控制器上安装 SQL Server。
- 在不同的 Windows 账户下运行各自的 SQL Server 服务。
- 在多层环境中，在不同的计算机上运行 Web 逻辑和业务逻辑。

4)创建具有最低特权的服务账户

SQL Server 安装程序自动向一个或多个服务账户授予对与 SQL Server 相关的文件的完全控制权限，以及对本地管理员组的完全控制权限。但是，通过创建具有运行 SQL Server 服务所需的最低特权的 Microsoft Windows 账户，可以将未经授权的访问降至最低。

5)配置安全的文件系统

使用恰当的文件系统可以增加安全性，对于所安装的 SQL Server 软件，应注意以下两点：

(1)使用 NTFS 文件系统。NTFS 是安装 SQL Server 软件的首选文件系统，因为该系统比 FAT 文件系统更稳定，可恢复性更强，并可以启用安全选项(如文件和目录访问控制列表及加密文件系统的文件加密)。有些 SQL Server 数据库引擎功能依赖于 NTFS，其中包括数据库快照和联机数据库一致性检查。请注意，基于 FAT 的系统有 4 GB 的文件大小限制。在安装过程中，如果 SQL Server 检测到 NTFS，则它将设置相应的文件和目录访问控制列表。不应对这些权限进行任何更改。

(2)如果使用加密文件系统，则数据库文件将以运行 SQL Server 的账户的身份加密。只有此账户才能解密这些文件。如果必须更改运行 SQL Server 的账户，则应首先在旧账户下解密这些文件，然后再在新账户下重新加密这些文件。

6)禁用 NetBIOS 和服务器消息块

边界网络中的服务器应禁用所有不必要的协议，包括 NetBIOS 和服务器消息块。其中 NetBIOS 使用的端口有 UDP/137(NetBIOS 名称服务)、UDP/138(NetBIOS 数据报服务)、TCP/139(NetBIOS 会话服务)，服务器消息块使用的端口有 TCP/139 和 TCP/445(文件和打印共享服务)。

7.2.2　SQL Server 2005 的安装

在进行安全准备之后就可以进行 SQL Server 2005 的安装了，SQL Server 2005 的安装过程与其他 Microsoft Windows 系列产品类似。其安装过程如下：

(1)将 SQL Server 2005 的安装光盘放入光驱中，双击 servers 目录下的 setup，打开如图 7-1 所示的“最终用户许可协议”对话框，选中“我接受许可条款和条件”复选框，单击“下一步”按钮，打开“安装必备组件”对话框，显示需要安装的组件，如图 7-2 所示。

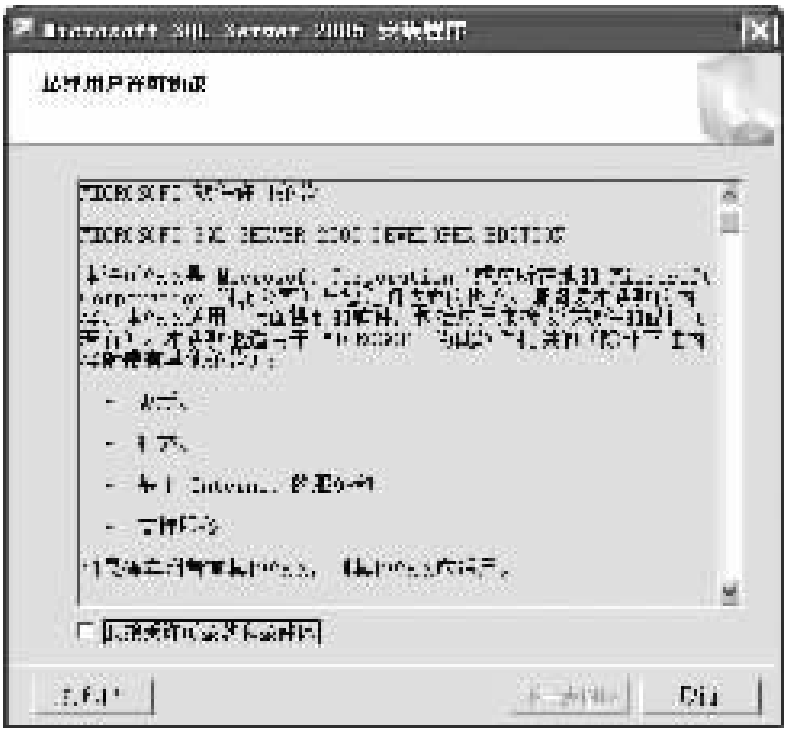

图 7-1 “最终用户许可协议”对话框

图 7-2 “安装必备组件”对话框

(2)单击“安装”按钮自动开始安装,安装完成后单击“下一步”按钮,开始扫描计算机配置,如图 7-3 所示,扫描完计算机配置后自动进入“欢迎使用 Microsoft SQL Server 安装向导”,如图 7-4 所示。

(3)单击“下一步”按钮,打开“系统配置检查”对话框,检查系统中是否有潜在的安装问题。系统检查完毕后,“报告”按钮变为可用,且当所有检查结果成功或失败的检查项不严重时,“下一步”按钮也变为可用,如图 7-5 所示。单击“下一步”按钮,系统运行几秒后打开“注册信息”对话框,如图 7-6 所示,输入姓名、公司信息,单击“下一步”按钮。

注意:如果在系统配置检查时,某些项出现“警告”或者“错误”提示,可以通过对应操作

后的“消息”一栏中所给出的提示信息进行处理。

图 7-3　扫描计算机配置

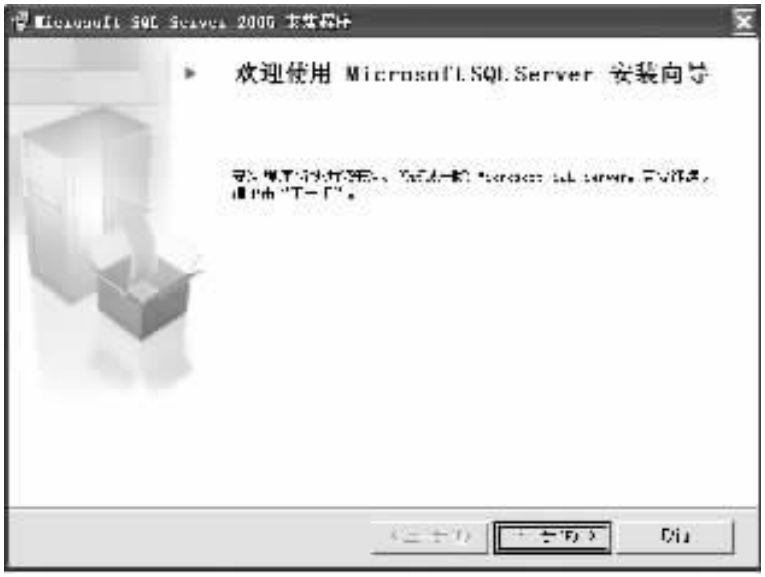

图 7-4　欢迎使用 Microsoft SQL Server 安装向导

图 7-5　“系统配置检查”对话框

图 7-6 “注册信息”对话框

(4)在打开的“要安装的组件”对话框中,可以选择系统要安装或升级的组件选项,如图 7-7 所示,其中共 6 个组件选项,各组件的功能如表 7-4 所示。

表 7-4 SQL Server 2005 各组件的功能

SQL Server 2005 组件	功能说明
SQL Server Database Services	包括数据库引擎(用于存储、处理和保护数据的核心服务)复制、全文搜索以及用于管理关系数据和 XML 数据的工具
Analysis Services	包括用于创建和管理联机分析处理(OLAP)以及数据挖掘应用程序的工具
Reporting Services	包括用于创建、管理和部署表格报表、矩阵报表、图形报表以及自由格式报表的服务器和客户端组件。Reporting Services 还是一个可用于开发报表应用程序的可扩展平台
Notification Services	是一个平台,用于开发和部署将个性化即时信息发送给各种设备上的用户的应用程序
Integration Services	是一组图形工具和可编程对象,用于移动、复制和转换数据
工作站组件、联机丛书和开发工具	工作站组件主要指连接组件和管理工具。其中连接组件包括DB-Library、OLEDB for OLAP、ODBC、ADODB 和 ADOMD+的网络库。管理工具包括 SQL Server Management Studio、SQL Server 配置管理器、SQL Server Profiler 和复制监视器。SQL Server 联机丛书是 SQL Server 2005 的核心文档。开发工具是用于 Analysis Services、Reporting Services 和 Integration Services 解决方案的集成开发环境

选择好相应的组件后,单击“下一步”按钮(没有选择项时该按钮不可用)进入“实例名”对话框,如图 7-8 所示,为安装的软件选择默认实例或命名实例。此处选择“默认实例”单选

按钮，单击“下一步”按钮。

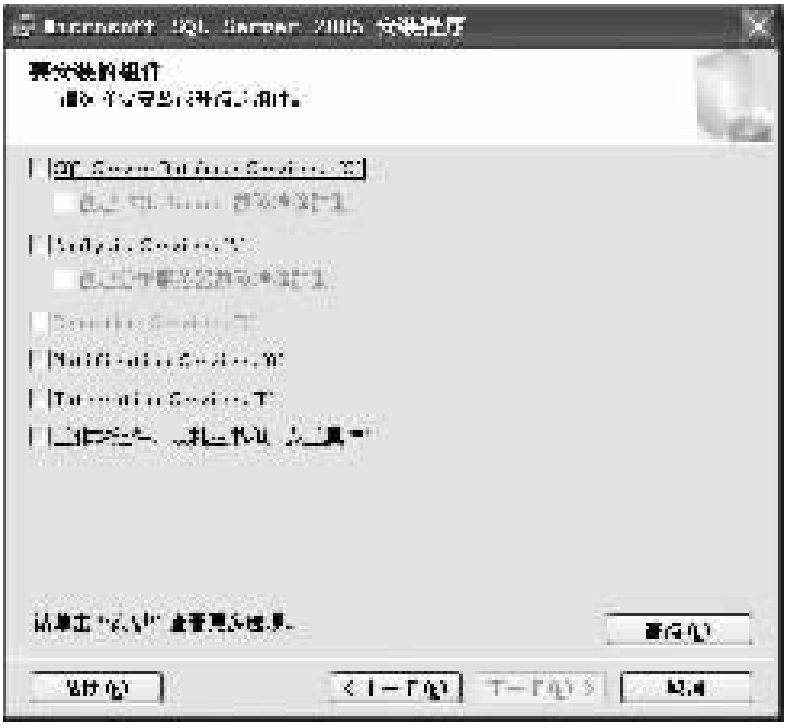

图 7-7　“要安装的组件”对话框

图 7-8　“实例名”对话框

(5)在打开的“服务账户”对话框中进行设置，如图 7-9 所示。该对话框中可以设置为对所有服务使用一个域账户，也可以根据需要为每个服务指定单独的账户，还可以设置“使用内置系统账户”。其中，“使用内置系统账户”指定一个不需要密码的本地系统账户，以连接到同一台计算机上的 SQL Server；“使用域用户账户”指定一个使用 Windows 身份验证的域用户账户，以设置并连接到 SQL Server。设置完毕后单击“下一步”按钮，打开“身份验证模式”对话框，如图 7-10 所示，设置身份验证模式，单击“下一步”按钮。

图 7-9 “服务账户”对话框

图 7-10 “身份验证模式”对话框

(6)在打开的“排序规则设置”对话框中指定 SQL Server 实例的排序规则,如图 7-11 所示。单击“下一步”按钮,打开“错误和使用情况报告设置”对话框,如图 7-12 所示。在该对话框中设置错误处理方式,单击“下一步”按钮,打开“准备安装”对话框,如图 7-13 所示。单击“安装”按钮,打开“安装进度”对话框,如图 7-14 所示。安装完毕后单击 “下一步”按钮,打开“完成 Microsoft SQL Server 2005 安装”对话框,如图 7-15 所示。单击“完成”按钮,完

成 SQL Server 2005 的安装。

图 7-11　“排序规则设置”对话框

图 7-12　“错误和使用情况报告设置”对话框

图 7-13　“准备安装”对话框

图 7-14 “安装进度”对话框

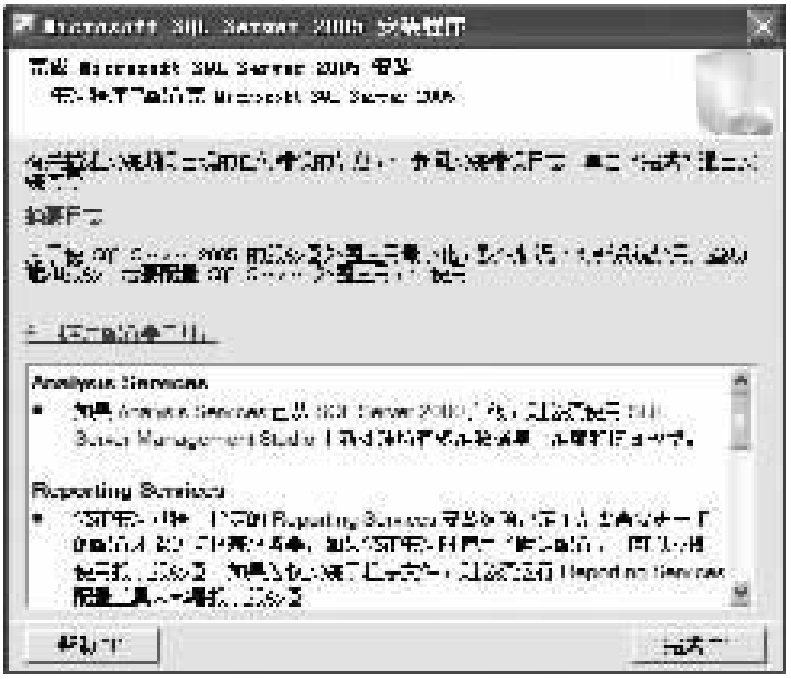

图 7-15 “完成 Microsoft SQL Server 2005 安装”对话框

7.2.3 SQL Server 2005 安装后的安全工作

安装完成后,若要增强所安装的 SQL Server 2005 软件的安全性,应遵循以下有关账户和身份验证模式的最佳做法。

1)服务账户

- 以可能的最低特权运行 SQL Server 服务。
- 将 SQL Server 服务与 Windows 账户相关联。

2)身份验证模式

连接 SQL Server 时要求 Windows 身份验证。

3)强密码

- 即使使用了 Windows 身份验证,也始终为 sa 账户指派强密码。
- 始终对所有 SQL Server 账户使用强密码。

7.2.4　SQL Server 2005 系统数据库简介

SQL Server 2005 有 5 个系统数据库,分别是 master、model、msdb、resource 和 tempdb,其功能和主要用途分别如下。

1. master

master 数据库记录 SQL Server 系统的所有系统级信息,包括实例范围的元数据(如登录账户)、端点、链接服务器和系统配置设置。master 数据库还记录所有其他数据库是否存在以及这些数据库文件的位置。另外,它还记录 SQL Server 的初始化信息。因此,如果 master 数据库不可用,则 SQL Server 无法启动。在 SQL Server 2005 中,系统对象不再存储在 master 数据库中,而是存储在 resource 数据库中。

2. model

model 数据库用于在 SQL Server 实例上创建的所有数据库的模板,因为每次启动 SQL Server 时都会创建 tempdb,所以 model 数据库必须始终存在于 SQL Server 系统中。当发出 CREATE DATABASE(创建数据库)命令时,将通过复制 model 数据库中的内容来创建数据库的第一部分,然后用空页填充新数据库的剩余部分,如果修改 model 数据库,之后创建的所有数据库都将继承这些修改。

3. msdb

msdb 数据库是代理服务数据库,为报警、任务调度和记录操作员的操作提供存储空间。

4. resource

resource 资源数据库是只读数据库,它包含了 SQL Server 2005 中的所有系统对象。SQL Server 系统对象(如 sys. objects)在物理上持续存在于 resource 数据库中,但在逻辑上,它们出现在每个数据库的 sys 架构中。resource 为一个隐藏数据库,无法使用 SQL 命令来看到它。

5. tempdb

tempdb 系统数据库是连接到 SQL Server 实例的所有用户都可用的全局资源,它保存所有临时表和临时存储过程。另外,它还用来满足所有其他临时存储要求,例如,存储 SQL Server 生成的工作表。每次启动 SQL Server 时,都要重新创建 tempdb,以使系统启动时,该数据库总是空的。在断开连接时会自动删除临时表和存储过程,并且在系统关闭后没有活动连接。因此,tempdb 中不会有任何内容从一个 SQL Server 会话保存到另一个会话。

7.3　SQL Server 2005 工具和实用程序

在 Windows 环境下完成 SQL Server 2005 安装后产生的程序组如图 7-16 所示,该程序组中包含 6 个程序项,下面对它们的功能进行简要介绍。

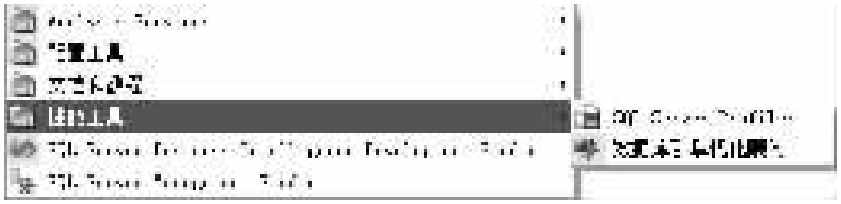

图 7-16 安装完成后的程序项

7.3.1 Analysis Services

Microsoft SQL Server 2005 Analysis Services(SSAS)为商业智能应用程序提供了联机分析处理(OLAP)和数据挖掘功能。Analysis Services 允许设计、创建和管理多维结构,使其包含从其他数据源(如关系数据库)聚合的数据,并通过这种方式来支持 OLAP。对于数据挖掘应用程序,Analysis Services 允许使用多种行业标准的数据挖掘算法来设计、创建和可视化从其他数据源构造的数据挖掘模型。

7.3.2 配置工具

在配置工具中,最常用的是 SQL Server Configuration Manager,即 SQL Server 配置管理器,如图 7-17 所示。SQL Server 配置管理器是一种工具,用于管理与 SQL Server 相关联的服务、配置 SQL Server 使用的网络协议以及从 SQL Server 客户端计算机管理网络连接配置。SQL Server 配置管理器是一个 Microsoft 管理控制台管理单元,可以从“开始”菜单进行访问,也可以将其添加到其他任何 Microsoft 管理控制台中。Microsoft 管理控制台(mmc.exe)使用 Windows system32 文件夹中的 SQLServerManager.msc 文件打开 SQL Server 配置管理器。SQL Server 配置管理器集成了 SQL Server 2000 的服务器网络实用工具、客户端网络实用工具和服务管理器工具。

图 7-17 配置管理器界面

7.3.3 文档和教程

SQL Server 2005 提供了强大的联机帮助文档和示例教程,它与微软其他应用系统的帮助文档类似,支持索引和全文搜索功能,可根据关键词来快速查找所需信息。当用户安装 SQL Server 2005 时,可以安装 SQL Server 文档(称为“SQL Server 2005 联机丛书”)、示例应用程序和教程。联机丛书、示例和教程涵盖了有效使用 SQL Server 所需的概念和过程。

联机丛书还包含使用 SQL Server 存储、检索、报告和修改数据的语言和编程接口的参考材料。如果使用 SQL Server 2005 安装程序安装文档和教程，则可以通过以下两种方式访问文档和教程：

(1)通过 Microsoft SQL Server 2005 程序组的“文档和教程”子菜单。

(2)通过 SQL Server 工具和实用工具中的 F1 帮助和动态帮助。

SQL Server 2005 联机丛书还可以从网站获得(可以以网页的形式浏览，也可下载到本地计算机浏览)，使用户可以在没有安装 SQL Server 2005 的计算机上访问文档。联机丛书如图 7-18 所示。

图 7-18　“联机丛书”窗口

7.3.4　性能工具

性能工具包括数据库引擎优化顾问和 SQL Server Profiler，下面分别介绍。

1. 数据库引擎优化顾问

企业数据库系统的性能取决于组成这些系统的数据库中物理设计结构的有效配置。这些物理设计结构包括索引、聚集索引、索引视图和分区，其目的在于提高数据库的性能和可管理性。Microsoft SQL Server 2005 提供了数据库引擎优化顾问，这是分析一个或多个数据库工作负荷的性能效果的工具。分析数据库的工作负荷效果后，数据库引擎优化顾问会提供在 Microsoft SQL Server 2005 数据库中添加、删除或修改物理设计结构的建议，用户可以根据这些建议对对应的物理结构进行相应的调整，实现这些调整后，数据库引擎优化顾问就使得系统能够用最短的时间执行工作负荷任务，从而提高数据库引擎服务应用系统的效率。

数据库引擎优化顾问提供了两种界面：

(1)图形界面方式：一种用于优化数据库、查看优化建议和报告的工具。执行“开始”→“所有程序”→Microsoft SQL Server 2005→“性能工具”→“数据库引擎优化顾问”命令，打开

“连接到服务器”对话框，在该对话框中，查看默认设置并单击“连接”按钮，打开 Database Enging Tuning Advisor 窗口，即数据库引擎优化顾问配置窗口，如图 7-19 所示。第一次打开时，显示两个主窗格。左窗格包含会话监视器，其中列出已对此 Microsoft SQL Server 实例执行的所有优化会话；右窗格包含“常规”和“优化选项”选项卡，单击“优化选项”选项卡（如图 7-20 所示），可设置优化时间、数据库中使用的物理设计结构、使用的分区策略、数据库中保留的物理设计结构等。

图 7-19　数据库引擎优化顾问配置窗口

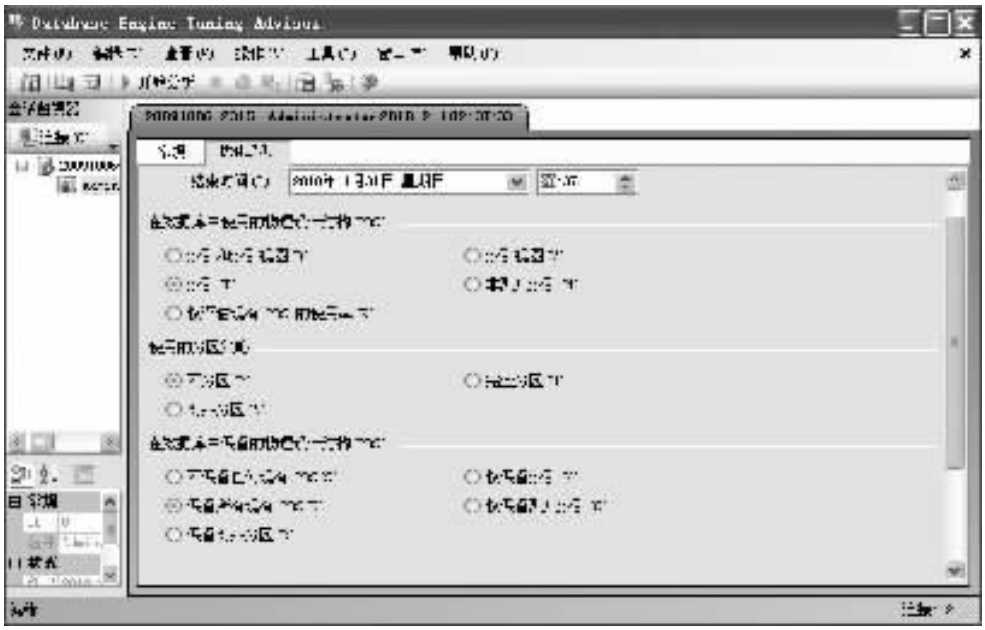

图 7-20　数据库引擎优化顾问优化选项界面

(2)dta 命令实用工具：用于实现数据库引擎优化顾问在软件程序和脚本方面的功能。在早期版本的 SQL Server 中，数据库引擎优化顾问的一些功能是由索引优化向导提供的，现在，数据库引擎优化顾问评估更多类型的事件和结构并提供更高质量的建议。

2. SQL Server Profiler

SQL Server Profiler（SQL 分析器）是一个功能丰富的工具，用于创建和管理跟踪，并分

析和重播跟踪结果。对系统管理员来说，它是一个连续实时地捕获用户活动情况的间谍。

使用 SQL Server Profiler 主要可以实现下述功能：

- 监视 SQL Server Database Engine、分析服务器或 Integration Services 的实例(在它们发生后)的性能。
- 调试 Transact-SQL 语句(7.5 节将详细介绍)和存储过程。
- 通过标识低速执行的查询来分析性能。
- 通过重播跟踪来执行负载测试和质量保证。
- 重播一个或多个用户的跟踪。
- 通过保存显示计划的结果来执行查询分析。
- 在项目开发阶段，通过单步执行语句来测试 Transact-SQL 语句和存储过程，以确保代码按预期方式运行。
- 通过捕获生产系统中的事件并在测试系统中重播这些事件来解决 SQL Server 中的问题。这对测试和调试很有用，并使得用户可以不受干扰地继续使用生产系统。
- 审核和检查在 SQL Server 实例中发生的活动。这使得安全管理员可以检查任何审核事件，包括登录尝试的成功与失败，以及访问语句和对象的权限的成功与失败。
- 将性能计数器与跟踪关联以诊断性能问题。
- 配置可用于以后跟踪的跟踪模板。
- 聚合跟踪结果以允许对相似事件类进行分组和分析。这些结果基于单个列分组提供计数。
- 允许非管理员用户创建跟踪。

启动 SQL Server Profiler 可以通过“开始”菜单，或者通过 SQL Server Management Studio 中的“工具”菜单，或者通过数据库引擎优化顾问中的“工具”菜单。启动 SQL Server Profiler 后，单击“文件”→“新建跟踪”选项，打开如图 7-21 所示的“连接到服务器”对话框，选择对应的服务器类型、服务器名称、身份验证方式以及用户名和密码后，单击“连接”按钮，打开“跟踪属性”对话框，在“常规”选项卡中设置跟踪名称、跟踪提供程序名称、跟踪提供程序类型、使用模板、保存的位置、是否启用跟踪停止时间等，如图 7-22 所示。在“事件选择”选项卡中设置跟踪的事件和事件列，如图 7-23 所示。

图 7-21　“连接到服务器”对话框

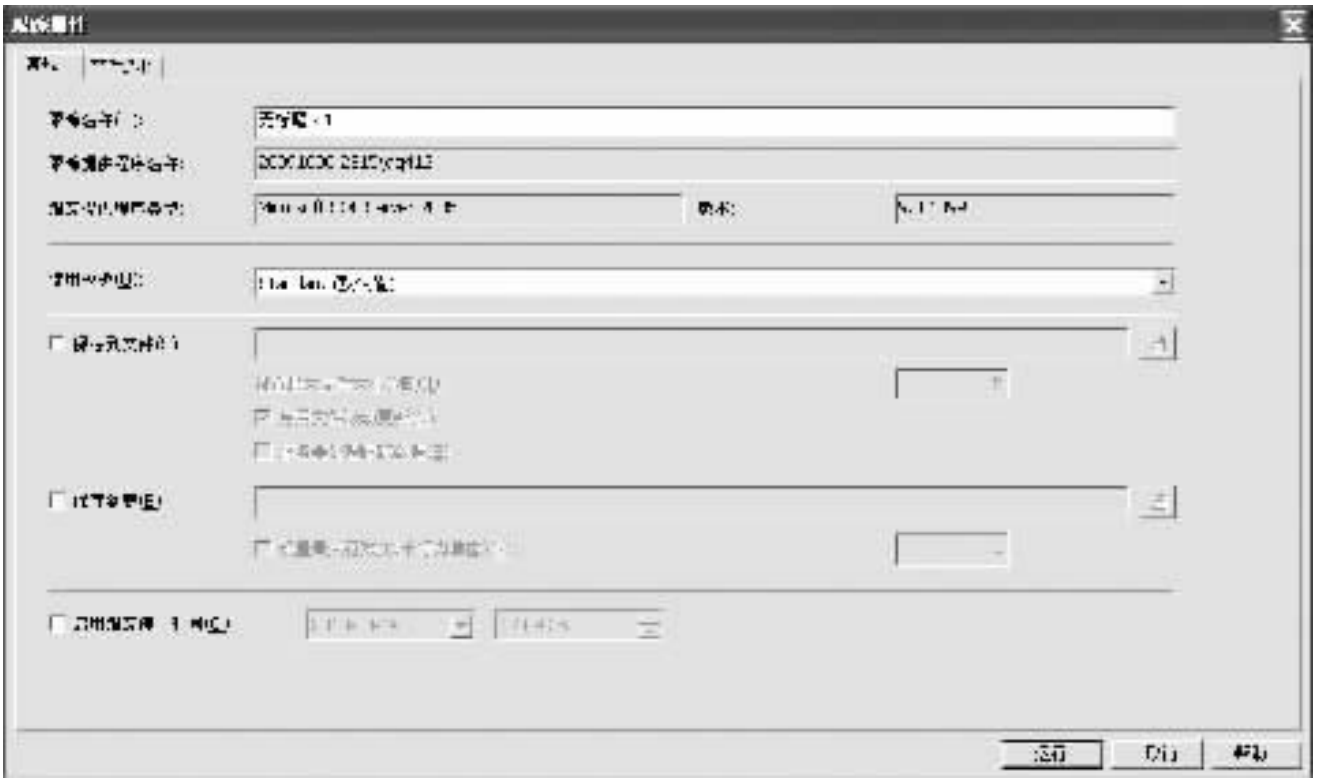

图 7-22 “常规”选项卡

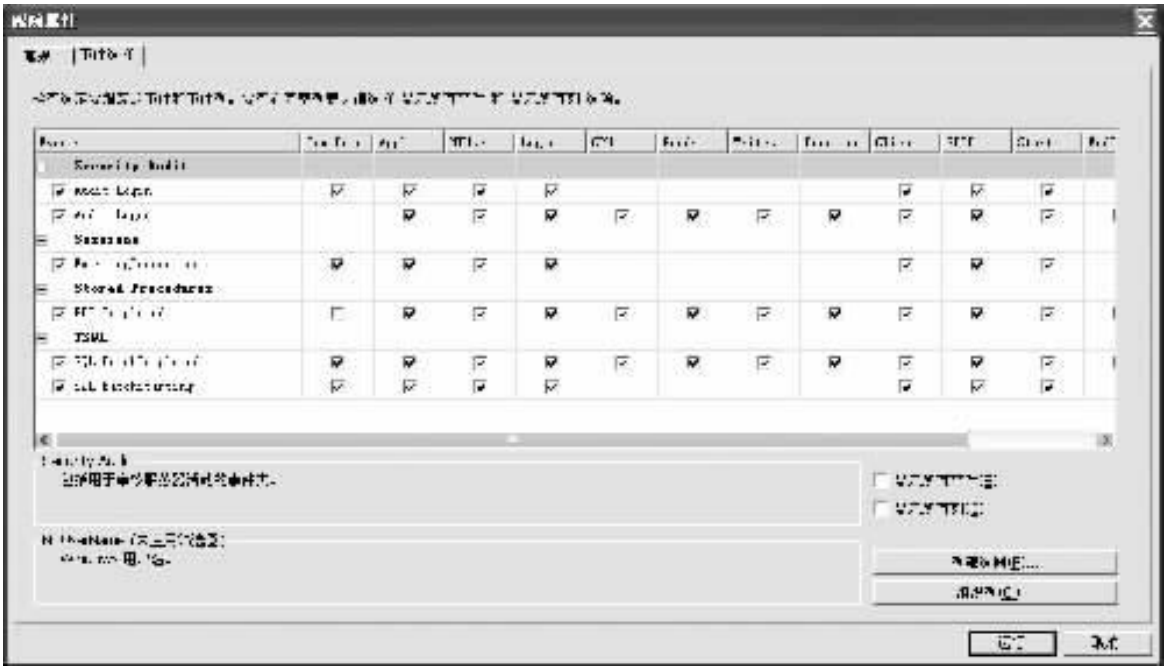

图 7-23 “事件选择”选项卡

7.3.5 SQL Server Business Intelligence Development Studio

Business Intelligence Development Studio(商业智能开发平台)是一个集成的环境,用于开发商业智能构造(如多维数据集、数据源、报告和 Integration Services 软件包)。其界面如图 7-24 所示。Business Intelligence Development Studio 包含一些项目模板,这些模板可以提供开发特定构造的上下文。例如,如果要创建一个包含多维数据集、维数或挖掘模型的 Analysis Services 数据库,则可以选择一个 Analysis Services 项目。

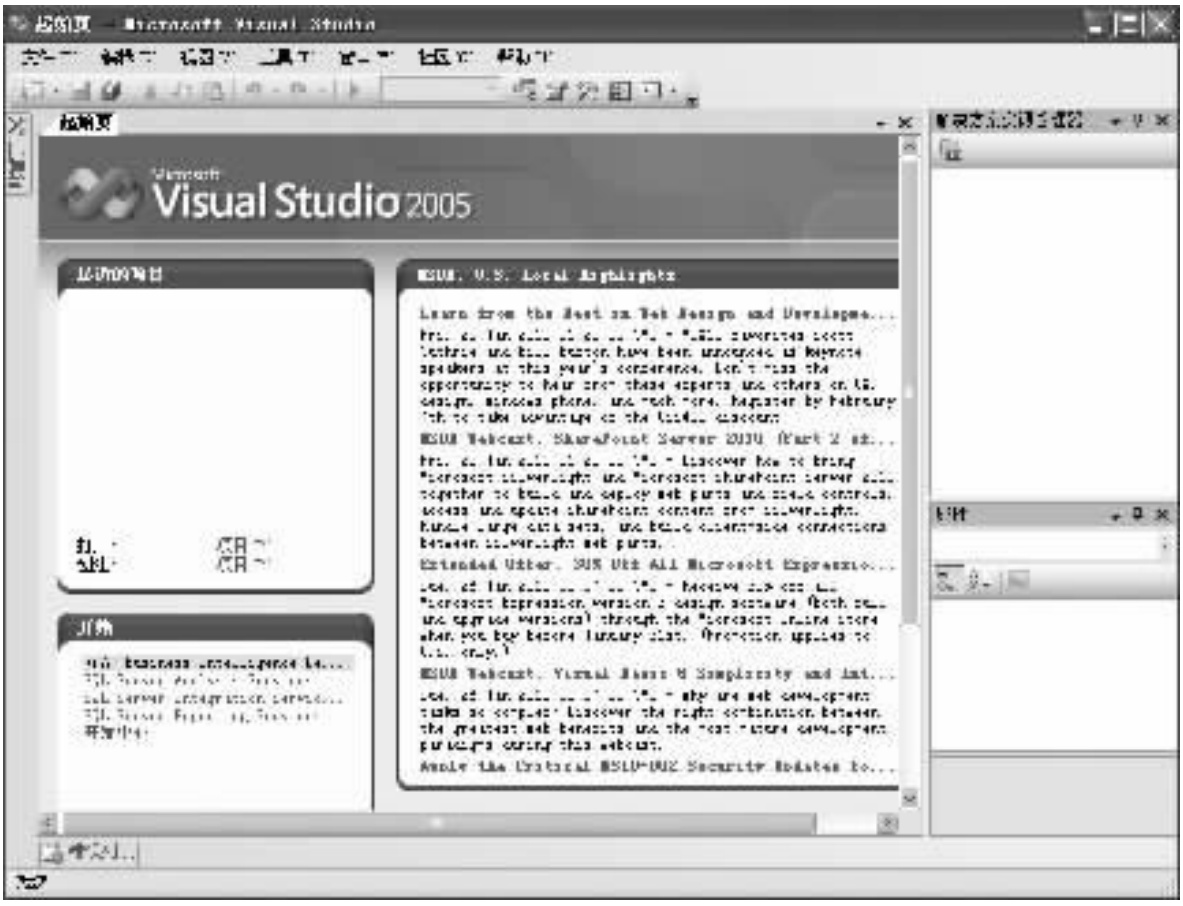

图 7-24　商业智能开发平台界面

在 Business Intelligence Development Studio 中开发商业项目时，可以将其作为某个解决方案的一部分进行开发，而该解决方案独立于具体的服务器。例如，可以在同一个解决方案中包括 Analysis Services 项目、Integration Services 项目和 Reporting Services 项目。在开发过程中，可以将对象部署到测试服务器中进行测试，然后可以将项目的输出结果部署到一个或多个临时服务器或生产服务器。

7.3.6　SQL Server Management Studio

SQL Server Management Studio(SQL Server 管理平台)是一个集成的环境，用于访问、配置和管理所有 SQL Server 组件。SQL Server Management Studio 组合了大量图形工具和丰富的脚本编辑器，使各种技术水平的开发人员和管理人员都能访问 SQL Server。

SQL Server Management Studio 将以前版本 SQL Server 中包括的企业管理器和查询分析器的各种功能，组合到一个单一环境中。此外，SQL Server Management Studio 还提供了用于管理 Analysis Services、Integration Services、Reporting Services 和 XQuery 的环境。此环境为开发者提供了一个熟悉的体验，为数据库管理人员提供了一个单一的实用工具，使他们能够通过易用的图形工具和丰富的脚本完成任务。

利用 SQL Server 管理平台，主要可以实现以下功能，今后各章节中所涉及的操作也主要通过该管理平台来实现。

- 注册服务器。
- 管理数据库。
- 创建对象，如数据库、表、数据库用户和登录名等。

- 管理安全性。
- 配置复制。
- 管理索引等。

要打开 SQL Server 2005 管理平台，可以执行“开始”→“所有程序”→Microsoft SQL Server 2005→SQL Server Management Studio 命令，启动后，界面如图 7-25 所示。

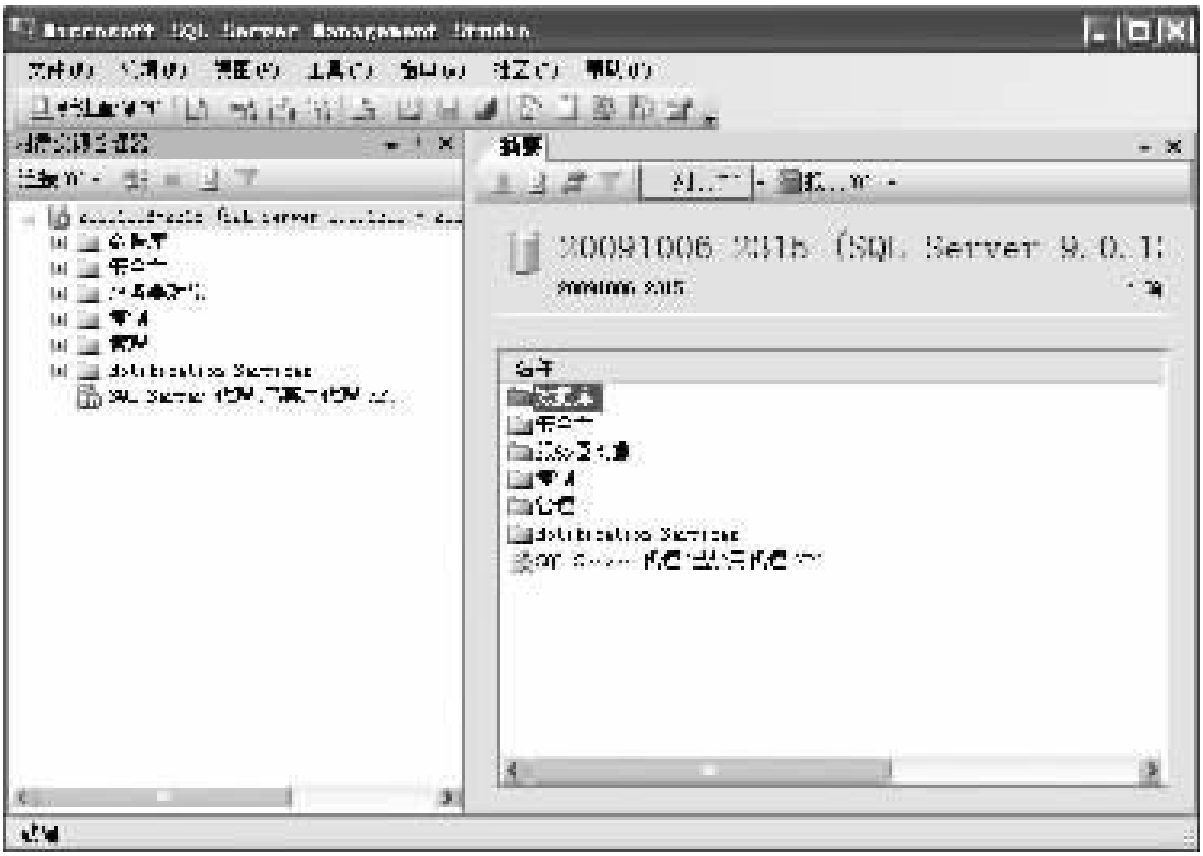

图 7-25　SQL Server 管理平台界面

默认情况下，SQL Server 管理平台中将显示两个组件窗口，并可以通过执行菜单栏的“视图”→“已注册的服务器”命令，打开“已注册的服务器”窗口，下面分别介绍这 3 个窗口：

(1)“已注册的服务器”窗口列出了经常管理的服务器。可以在此列表中添加和删除服务器。如果计算机以前安装了 SQL Server 2000 企业管理器，则系统将提示用户导入已注册服务器的列表；否则，列出的服务器中仅包含运行 Management Studio 的计算机上的 SQL Server 实例。如果未显示所需的服务器，请右击“数据库引擎”，在弹出的快捷菜单中选择“更新本地服务器注册”命令。

(2)对象资源管理器包括与其连接的所有服务器的信息，是服务器中所有数据库对象的树视图。此树视图可以包括 SQL Server Database Engine、Analysis Services、Reporting Services、Integration Services 和 SQL Server Mobile 的数据库。打开 SQL Server 管理平台时，系统会提示将对象资源管理器连接到上次使用的设置。可以在“已注册的服务器”组件中双击任意服务器进行连接，但无须注册要连接的服务器。

(3)文档窗口是 Management Studio 中的最大部分。文档窗口可能包含查询编辑器和浏览器窗口。默认情况下，将显示已与当前计算机上的数据库引擎实例连接的“摘要”页。

注意：SQL Server Management Studio 可用于开发和管理数据库对象，以及用于管理和配置现有 Analysis Services 对象。Business Intelligence Development Studio 可用于开发商

业智能应用程序。如果要实现使用 SQL Server 数据库服务的解决方案，或者要管理使用 SQL Server、Analysis Services，Integration Services 或 Reporting Services 的现有解决方案，则应当使用 SQL Server Management Studio。如果要开发使用 Analysis Services、Integration Services 或者 Reporting Services 的方案，则应当使用 Business Intelligence Development Studio。

7.4　SQL Server 2005 卸载

SQL Server 2005 安装完毕后就可以利用 SQL Server 2005 提供的强大数据管理功能来使用和管理数据库、数据表等信息。在使用完成后如果需要卸载或删除 SQL Server 2005 中的特定的组件或者完全卸载整个 SQL Server 2005 系统，则可以通过再次运行安装程序或者通过以下步骤来实现：

(1)执行“开始”→“控制面板”，打开“控制面板”窗口，双击“添加或删除程序”图标。

(2)在“添加或删除程序”窗口中选择要卸载的 SQL Server 2005 组件，如图 7-26 所示，单击“删除”按钮，启动 SQL Server 2005 安装向导，按照安装向导的提示进行操作。如果要删除特定的组件，则单击“更改”按钮。

图 7-26　“添加或删除程序”窗口

(3)单击“更改”按钮后打开“选择组件”对话框，如图 7-27 所示。选择相应组件后，“下一步”按钮变为可用，单击“下一步”按钮打开 SQL Server 2005 的安装向导，再单击“下一步”按钮，则可进行相应组件的卸载操作。

(4)单击“选择组件”对话框的“报告”按钮，打开“安装报告”对话框，可以查看计算机上安装的 SQL Server 2005 各组件及其功能的列表。如图 7-28 所示。

图 7-27 “选择组件”对话框

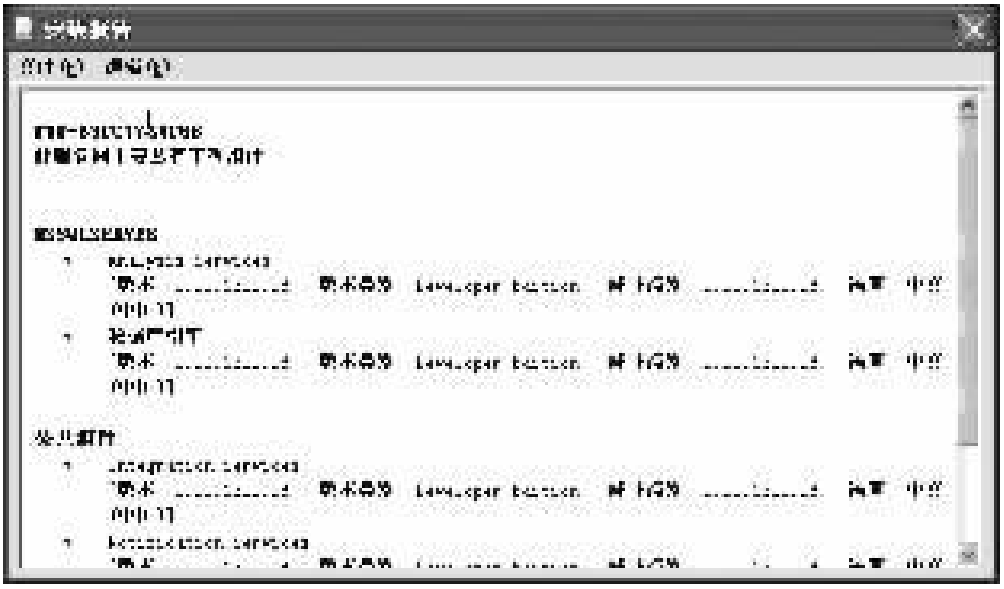

图 7-28 “安装报告”对话框

7.5 Transact-SQL 简介

Transact-SQL 语言(简称 T-SQL 语言)是 SQL Server 对标准 SQL 语言的扩充,如引入了程序设计的思想,增强了程序的流程语句等。因此,在 Transact-SQL 语言中,标准的 SQL 语句使用起来畅通无阻。Transact-SQL 语言最主要的用途是设计服务器的能够在后台执行的程序块,如存储过程、触发器等。

7.5.1　变量

在 Transact-SQL 中可以使用两种变量：局部变量和全局变量。

1. 局部变量

局部变量是用户可以自定义的变量，它的作用范围仅在程序内部。局部变量在程序中通常用来存储从表中查询到的数据，或当做程序执行过程中的暂存变量。局部变量必须以“@”开头，而且必须先用 DECLARE 命令说明后才可使用。其语法形式如下：

```
DECLARE @变量名 变量类型[,@变量名 变量类型…]
```

其中，变量类型可以是 SQL Server 2005 支持的所有数据类型。

在 Transact-SQL 中不能像在一般的程序语言中那样，使用“变量＝变量值”来给变量赋值，必须使用 SELECT 或 SET 命令来设定变量的值。其语法如下：

```
SELECT @局部变量＝变量值
SET @局部变量＝变量值
```

例如，声明两个变量并赋值，语句如下：

```
USE student
GO
DECLARE @a char(20),@b int(3)
SET @a='s03'
SET @b=23
```

2. 全局变量

全局变量是 SQL Server 2005 系统内部使用的变量，其作用范围并不局限于某一程序，而是任何程序均可随时调用。全局变量通常存储一些 SQL Server 2005 的配置设定值和效能统计数据。用户可在程序中用全局变量来测试系统的设定值或 Transact-SQL 命令执行后的状态值。全局变量不是由用户的程序定义的，而是由系统定义和维护的，且只能使用预先说明及定义的全局变量。引用全局变量时必须以“@@”开头。SQL Server 提供的全局变量共有 33 个，但并不是每个都可用。而且局部变量与全局变量不能同名，否则会在应用时出错。

7.5.2　注释符

在 Transact-SQL 中可使用两类注释符：

(1)ANSI 标准的注释符“－－”用于单行注释。

(2)与 C 语言相同的程序注释符号，即“/＊……＊/”，“/＊”用于程序注释开头，“＊/”用于程序注释结尾，可以将程序中多行文字标示为注释。

7.5.3　流程控制语句及命令

1. BEGIN…END

BEGIN…END 语句用于将多个 Transact-SQL 语句组合为一个逻辑块。当控制语句执行一个包含两条或两条以上 Transact-SQL 语句的语句块时，可以使用 BEGIN 和 END 语

句。其语法格式如下：

```
BEGIN
  ＜命令行或程序块＞
END
```

BEGIN 和 END 语句必须成对使用。任何一条语句均不能单独使用。BEGIN 语句行后为 Transact-SQL 语句块。最后，END 语句行指示语句块结束。

BEGIN 和 END 语句用于下列情况：

- WHILE 循环需要包含语句块。
- CASE 函数的元素需要包含语句块。
- IF 或 ELSE 子句需要包含语句块。

BEGIN…END 语句块允许嵌套。

【例 7-1】 找出得分在 80 分以上的同学，并将这些同学的学号打印输出。

```
USE student
GO
DECLARE @message varchar(20)
IF EXISTS(SELECT * FROM study WHERE Score＞80)
  BEGIN
    SET @message='下列同学分数在 80 分以上，成绩优良'
    PRINT @message
  END
END IF
```

在本例中，BEGIN 和 END 定义一系列一起执行的 Transact-SQL 语句。如果没有包括 BEGIN…END 块，IF 条件仅使 SET 赋值语句执行，而不返回打印信息。

2. IF…ELSE

IF…ELSE 的语法格式如下：

```
IF＜条件表达式＞
  ＜命令行或程序块＞
ELSE＜条件表达式＞
  ＜命令行或程序块＞
```

IF 语句用于条件的测试。结果流的控制取决于是否指定了可选的 ELSE 语句。

- 指定 IF 而无 ELSE：IF 语句取值为 TRUE 时，执行 IF 语句后的语句或语句块，IF 语句取值为 FALSE 时，跳过 IF 语句后的语句或语句块。
- 指定 IF 并有 ELSE：IF 语句取值为 TRUE 时，执行 IF 语句后的语句或语句块，然后跳过 ELSE 语句及其之后的语句或语句块继续执行。IF 语句取值为 FALSE 时，跳过 IF 语句后的语句或语句块，而执行 ELSE 语句后的语句或语句块。

下面的示例中用到了带有语句块的 IF 条件。

【例 7-2】 如果数据库学分不低于 3 分，那么就显示“数据库学分过高!”；否则显示“数据库学科的学分正常”。

```
USE student
IF(SELECT Score FROM Course WHERE Cname="DataBase")＞=3
```

```
  BEGIN
    PRINT "数据库学分过高!"
  END
ELSE
  PRINT "数据库学科的学分正常"
```

3. CASE

计算条件列表并返回多个可能结果表达式之一。

CASE 具有两种格式：

- 简单 CASE 函数将某个表达式与一组简单表达式进行比较以确定结果。
- CASE 搜索函数计算一组布尔表达式以确定结果。

两种格式都支持可选的 ELSE 参数。CASE 的语法格式为：

格式 1：

```
CASE <条件表达式>
  WHEN <条件表达式> THEN <表达式>
  …
  WHEN <条件表达式> THEN <表达式>
  [ELSE <表达式>]
END
```

该语句的执行过程：将 CASE 后面表达式的值与各 WHEN 子句中的表达式的值进行比较，如果两者相等，则返回 THEN 后的表达式的值，然后跳出 CASE 语句，否则返回 ELSE 子句中的表达式的值。ELSE 子句是可选项。当 CASE 语句中不包含 ELSE 子句时，如果所有比较失败，CASE 语句将返回 NULL。

【例 7-3】 从学生表 Stu 中，选取 Sno 和 Ssex，如果 Ssex 为“男”则输出“M”，如果为“女”则输出“F”。

```
SELECT Sno,Ssex=
  CASE Ssex
    WHEN '男' THEN 'M'
    WHEN '女' THEN 'F'
  END
FROM Stu
```

格式 2：

```
CASE
  WHEN <条件表达式> THEN <表达式>
  …
  WHEN <条件表达式> THEN <表达式>
  [ELSE <表达式>]
END
```

该语句的执行过程：首先测试 WHEN 后的表达式的值，如果其值为真，则返回 THEN 后面的表达式的值，否则测试下一个 WHEN 子句中的表达式的值。如果所有 WHEN 子句后的表达式的值都为假，则返回 ELSE 后表达式的值。如果在 CASE 语句中没有 ELSE 子

句,则 CASE 表达式返回 NULL。

注意:CASE 命令可以嵌入到 SQL 命令中。

【例 7-4】 从 study 表中查询所有同学选课成绩情况,凡成绩小于 60 分输出“不及格”,60～69 分输出“及格”,70～89 分输出“良好”,大于或等于 90 分输出“优秀”。

```
SELECT Sno,Cno,score=
  CASE
    WHEN Score<60 THEN '不及格'
    WHEN Score BETWEEN 60 AND 69 THEN '及格'
    WHEN Score BETWEEN 70 AND 89 THEN '良好'
    WHEN Score>=90 THEN '优秀'
  END
FROM study
```

4. WHILE

只要指定的条件为真,则 WHILE 语句重复执行语句或语句块。BREAK 语句退出最内层 WHILE 循环,CONTINUE 语句重新开始 WHILE 循环。如果没有其他语句可以执行,则程序执行 BREAK 语句。如果要继续执行代码,则执行 CONTINUE 语句。

其语法格式如下:

```
WHILE <条件表达式>
BEGIN
  <命令行或程序块>
[BREAK]
[CONTINUE]
END
```

5. WAITFOR

WAITFOR 命令用来暂时停止程序执行,直到所设定的等待时间已过或所设定的时间已到才继续往下执行。其中,“时间”必须为 datetime 类型的数据,但不能包括日期。

其语法格式如下:

```
WAITFOR {DELAY 'time'|TIME 'time'}
```

各关键字含义如下:

- DELAY:用来设定等待的时间,最多可达 24 小时。
- TIME:用来设定等待结束的时间点。

6. RETURN

RETURN 命令用于结束当前程序的执行,返回到上一个调用它的程序或其他程序。在括号内可指定一个返回值。如果没有指定返回值,SQL Server 系统会根据程序执行的结果返回一个内定值,如:

0　程序执行成功

－1　找不到对象

－2　数据类型错误

－3　死锁

－4　违反权限原则
－5　语法错误
－6　用户造成的一般错误
－7　资源错误
－8　非致命的内部错误
－9　已达到系统的极限
－10 和－11　致命的内部不一致错误
－12　表或指针破坏
－13　数据库破坏
－14　硬件错误
如果运行过程产生了多个错误，则返回绝对值最大的数值。

7.5.4　其他语句及命令

1. PRINT

PRINT 语句用一个字符或 unicode 字符串表达式作为参数。它把这个字符串作为一个消息返回给应用程序。

PRINT 的语法格式如下：

```
PRINT 'any ASCII text'|@local_variable|@@FUNCTION|string_expression
```

2. BACKUP

BACKUP 命令用于将数据库内容或其他事务处理日志备份到存储介质（如软盘、硬盘、磁带等）上。

3. CHECKPOINT

CHECKPOINT 命令用于将当前工作的数据库中被更改过的数据页或日志页从数据库缓冲器中强制写入硬盘。

4. DBCC

数据库一致性检查程序命令。用于验证数据库完整性、查找错误、分析系统使用情况等。DBCC 命令后必须加上子命令，系统才知道要做什么。例如，DBCC CHECKALLOC 命令用来检查数据库内所有数据页的分配和使用情况。

5. EXECUTE

EXECUTE 命令用来执行存储过程。

6. KILL

KILL 命令用于终止某一过程的执行。默认情况下，sysadmin 和 processadmin 固定数据库角色的成员具有 KILL 的默认权限，KILL 权限不可转让。

7. RAISERROR

RAISERROR 命令用于在 SQL Server 系统返回错误信息时，同时返回用户指定的信息。

8. RESTORE

RESTORE 命令用来将数据库或其他事务处理日志备份文件由存储介质回存到 SQL

Server 系统中。

本章小结

本章主要介绍了 SQL Server 2005 的版本、环境需求、安装与配置的具体步骤、工具和实用程序以及 SQL Server 2005 的卸载。通过学习了解 SQL Server 2005 的版本和环境需求，掌握 SQL Server 2005 的安装、配置、卸载，熟练运用 SQL Server 2005 的工具和实用程序，它们在今后各章节中均有一定的应用。最后介绍了 Transact-SQL 语言，作为后续章节的理论基础。

习 题 7

1. 简述安装 SQL Server 2005 的硬件需求。
2. 简述 SQL Server 2005 各组件的功能。
3. 简述 SQL Server 2005 中 master 数据库的功能。
4. 简述 SQL Server 2005 管理平台的功能。
5. 简述卸载 SQL Server 2005 的方法。

第 10 章　存储过程与触发器

存储过程和触发器是 SQL Server 数据库系统重要的数据库对象，以 SQL Server 2005 为后台数据库所创建的应用具有重要的应用价值。本章介绍存储过程和触发器的概念、作用和基本操作。

10.1　存储过程概述

10.1.1　存储过程的概念

存储过程(store procedure)是为了实现某个特定任务，以一个存储单元的形式存储在服务器上的一组 Transact-SQL 语句的集合。也可以把存储过程看成是以数据库对象形式存储在 SQL Server 中的一段程序或函数。存储过程既可以是一些简单的 Transact-SQL 语句，也可以由一系列用来对数据库实现复杂任务规则的 Transact-SQL 语句或控制流语句所组成。

存储过程同其他编程语言中的过程相似，有以下特点：

(1)接收输入参数并以输出参数的形式将多个值返回调用过程或批处理。

(2)包含执行数据库操作(包括调用其他过程)的编程语句。

(3)向调用过程或批处理返回状态值，以表明成功或失败(及失败原因)。

使用存储过程，有以下优点：

(1)增强安全机制。SQL Server 可以只给用户访问存储过程的权限，而不授予用户访问存储过程引用的对象(表或视图)的权限。这样，用户可以通过存储过程操作数据库中的数据，而不能直接访问与存储过程相关的表，保证了数据的安全性。

(2)提高执行速度。用户可以多次使用存储过程的名称调用存储过程。存储过程在第一次执行时进行编译，然后将编译好的代码保存在高速缓存中，当用户再次执行该存储过程时，调用的是高速缓存中的编译代码。因此，其执行速度比执行相同的 Transact-SQL 语句快得多。

(3)减小网络流量。存储过程中包含大量的 Transact-SQL 语句，但它以一个独立的单元存放在服务器上。调用执行过程中，只需传递执行存储过程的调用命令即可执行结果，返回调用过程或批处理，从而减小了网络上数据的传输量。

10.1.2　存储过程的类型

在 SQL Server 2005 中，存储过程可以分为 3 类：用户自定义的存储过程、系统存储过程和扩展存储过程。

(1)用户自定义的存储过程。用户自定义的存储过程是用户根据自身需要,为完成某一特定功能,在自己的普通数据库中创建的存储过程。

(2)系统存储过程。系统存储过程以 sp_为前缀,主要用于从系统表中获取信息,为系统管理员管理 SQL Server 提供帮助,为用户查看数据库对象提供方便。例如,系统存储过程 sp_help 用来查看数据库对象信息。从物理意义上讲,系统存储过程存储在资源数据库中。从逻辑意义上讲,系统存储过程出现在每个系统定义数据和用户自定义数据库的 sys 架构中。

(3)扩展存储过程。扩展存储过程以 xp_为前缀,它是关系数据库引擎的开放式数据服务层的一部分,可以使用户在动态链接库(DLL)文件所包含的函数中实现逻辑,从而扩展了 Transact-SQL 语句的功能,并且可以像调用 Transact-SQL 语句一样调用这些函数。

10.2 存储过程的操作

10.2.1 创建存储过程

在 SQL Server 2005 中通常可以使用两种方法创建存储过程:一种是使用 Transact-SQL 语句创建存储过程,另一种是使用 SQL Server 管理平台创建存储过程。创建存储过程时,需要注意下列事项:

(1)只能在当前数据库中创建存储过程。

(2)创建者不但要具有数据库的 CREATE PROCEDURE 权限,还必须具有对架构(在其下创建过程)的 ALTER 权限。

(3)存储过程是数据库对象,其名称必须遵守标识符命名规则。

(4)不能将 CREATE PROCEDURE 语句与其他 Transact-SQL 语句组合到一个批处理中。

(5)创建存储过程时,应指定:输入参数和向调用过程或批处理返回的输出参数;执行数据库操作编程语句;返回调用过程或批处理以表明成功或失败的状态值。

1. 使用 Transact-SQL 语句创建存储过程

使用 Transact-SQL 语句创建存储过程的语法格式如下:

```
CREATE PROC[EDURE]procedure_name[;number]
[{@parameter data_type}[VARYING][=default][OUTPUT]][,…n]
[WITH{RECOMPILE|ENCRYPTION|RECOMPILE,ENCRYPTION}]
[FOR REPLICATION]
AS
sql_statement[…n]
```

参数说明如下:

- procedure_name:新建存储过程的名称,其名称必须符合标识符命名规则,且对于数据库及其所有者必须唯一。
- number:可选的整数,用来对同名的过程分组,以便用一条 DROP PROCEDURE 语句将同组的过程一起删除。例如,对于存储过程 orderproc1、orderproc2 等,可以使用 DROP PROCEDURE orderproc 语句将其全部删除。如果名称中包含定界标识

符，则数字不应包含在标识符中，只在存储过程名称前后使用适当的定界符。

- parameter：存储过程中的输入或输出参数。
- data_type：参数的数据类型。
- VARYING：用于指定作为输出参数支持的结果集（由存储过程动态构造，内容可能发生变化）。该选项仅用于游标参数。
- default：指参数的默认值，必须是常量或 NULL。如果定义了默认值，则不必指定该参数的值即可执行过程。
- OUTPUT：指示参数是输出参数。此选项的值可以返回给调用 EXECUTE 的语句。使用 OUTPUT 参数将值返回给过程的调用方。除非是 CLR 过程，否则 text、ntext 和 imaget 参数不能作为 OUTPUT 参数。使用 OUTPUT 关键字的输出参数可以为游标占位符，但 CLR 过程除外。
- RECOMPILE：表示 SQL Server 不保存存储过程的计划，该过程将在运行时重新编译。在使用非典型值或临时值而不希望覆盖缓存在内存中的执行计划时，最好使用 RECOMPILE 选项。
- ENCRYPTION：表示加密存储过程文本。
- FOR REPLICATION：用于指定不能在订阅服务器上执行为复制创建的存储过程。使用该选项创建的存储过程可用于存储过程筛选，且只能在复制过程中执行。该选项不能和 WITH RECOMPILE 选项一起使用。
- sql_statement：指存储过程中的任意数目和类型的 Transact-SQL 语句。

在创建存储过程时，一般情况下先编写实现存储过程功能的 SQL 语句，然后进行调试，当得到预期的结果后，再按照存储过程的语法创建存储过程。如果存储过程带有参数，可以先用一个实参来代替调试。

【例 10-1】 在 student 数据库中，创建一个查询存储过程 st_jsjbj，要求该存储过程列出计算机系的班级名称。创建过程如下：

（1）在创建本例存储过程时，可以先在编辑器中编写实现存储过程功能的Transact-SQL语句。代码如下：

```
USE student
GO
SELECT 班级名称 FROM 班级
WHERE 系部代码=(SELECT 系部代码 FROM 系部
WHERE 系部名称='计算机系')
```

（2）调试该语句正确后，再创建存储过程。在编辑器中输入其完整的代码如下：

```
USE student
GO
CREATE PROC dbo.st_jsjbj
AS
SELECT 班级名称 FROM 班级
WHERE 系部代码=(SELECT 系部代码 FROM 系部
WHERE 系部名称='计算机系')
GO
```

(3)单击工具栏上的“分析”按钮,进行语法检查;语法无误后,单击工具栏上的“执行”按钮,创建该存储过程。

2. 使用 SQL Server 管理平台创建存储过程

【例 10-2】 在 student 数据库中,创建一个名称为 st_jsjxsxx 的存储过程,要求该存储过程列出计算机系学生的姓名、性别和年龄信息。

创建步骤如下:

(1)启动 Microsoft SQL Server Management Studio,在“对象资源管理器”窗口中,依次展开“数据库”→student→“可编程性”。

(2)右击“存储过程”,在弹出的快捷菜单中单击“新建存储过程”选项,打开编辑器,该界面提供了一些与创建存储过程相关的命令语句及提示信息,如图 10-1 所示。

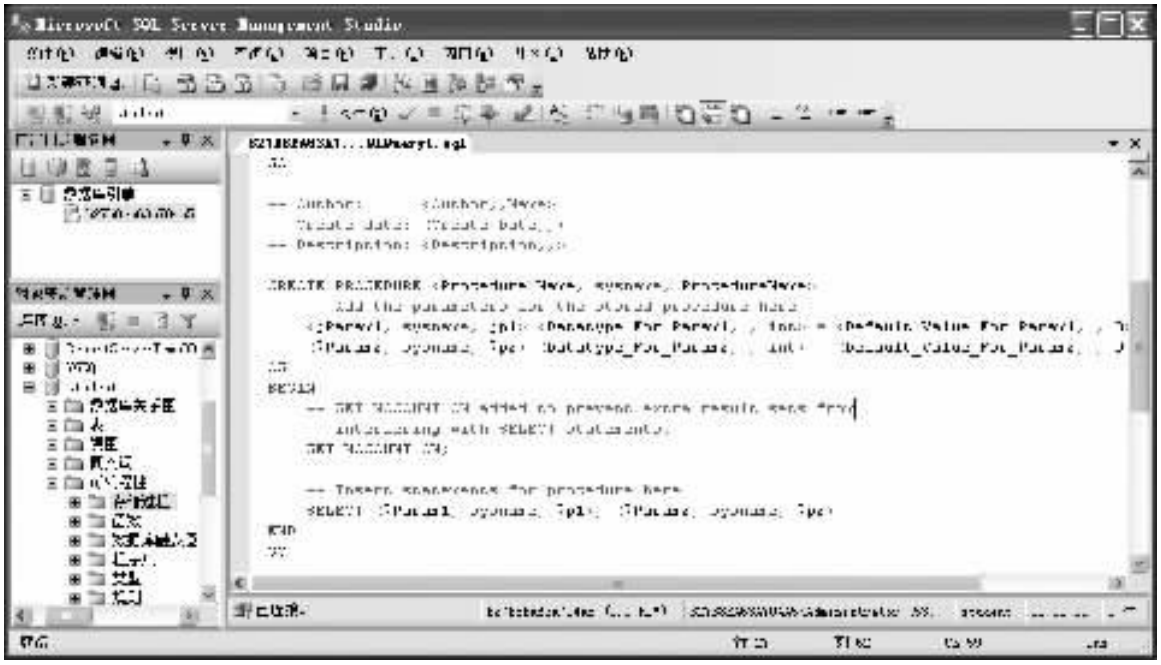

图 10-1 打开编辑器后的窗口

(3)在编辑器中根据相应提示输入存储过程名称和 Transact-SQL 语句,由于此存储过程不需要参数,删除参数部分即可。输入完成后的 st_jsjxsxx 存储过程如图 10-2 所示。

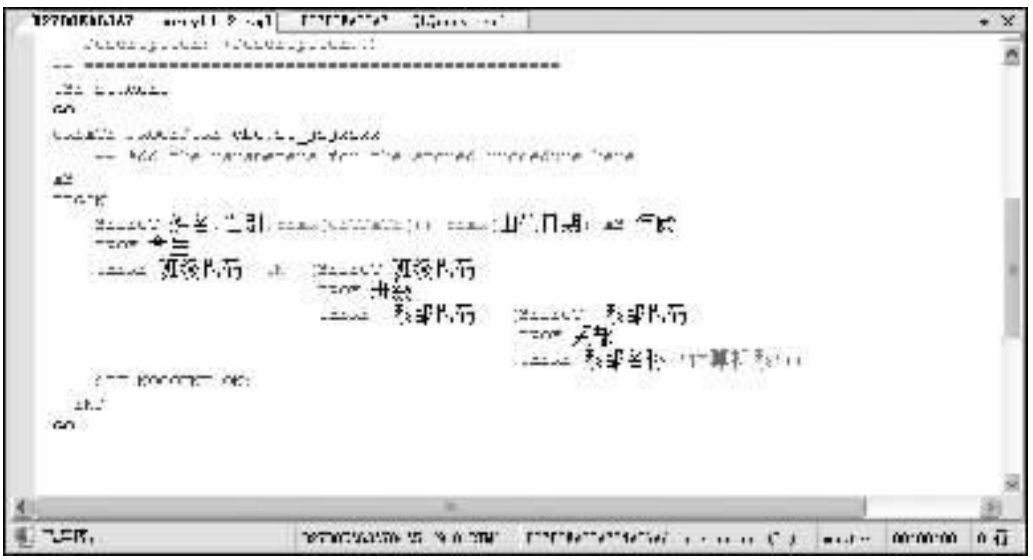

图 10-2 输入完成后的 st_jsjxsxx 存储过程

(4)单击工具栏上的“执行”按钮,完成存储过程的创建。如果需要保存存储过程代码,可单击工具栏上的“保存”按钮。

10.2.2　执行存储过程

对于储存在服务器上的存储过程,可以使用 EXECUTE 命令来执行它。使用 EXECUTE 命令执行存储过程的语法格式如下:

```
[[EXEC[UTE]]
{
    [@return_status=]
        {procedure_name[;number]|@procedure_name_var}
}
[[@parameter=]{value|@variable[OUTPUT]|[DEFAULT]]}
[,:n]
[WITH RECOMPILE]
```

如果存储过程是批处理中的第一句,EXECUTE 命令可以省略,可以使用存储过程的名字执行该存储过程;@return_status 是一个可选的整型变量,用来保存存储过程的返回状态值;@procedure_name_var 是局部定义变量名,用来代表存储过程的名称。其他参数与创建存储过程命令中的参数意义相同。

【例 10-3】　执行存储过程 st_jsjbj。

新建查询,用来执行存储过程 st_jsjbj,代码如下:

```
USE student
GO
EXECUTE st_jsjbj
GO
```

在编辑器中执行以上代码,结果如图 10-3 所示。

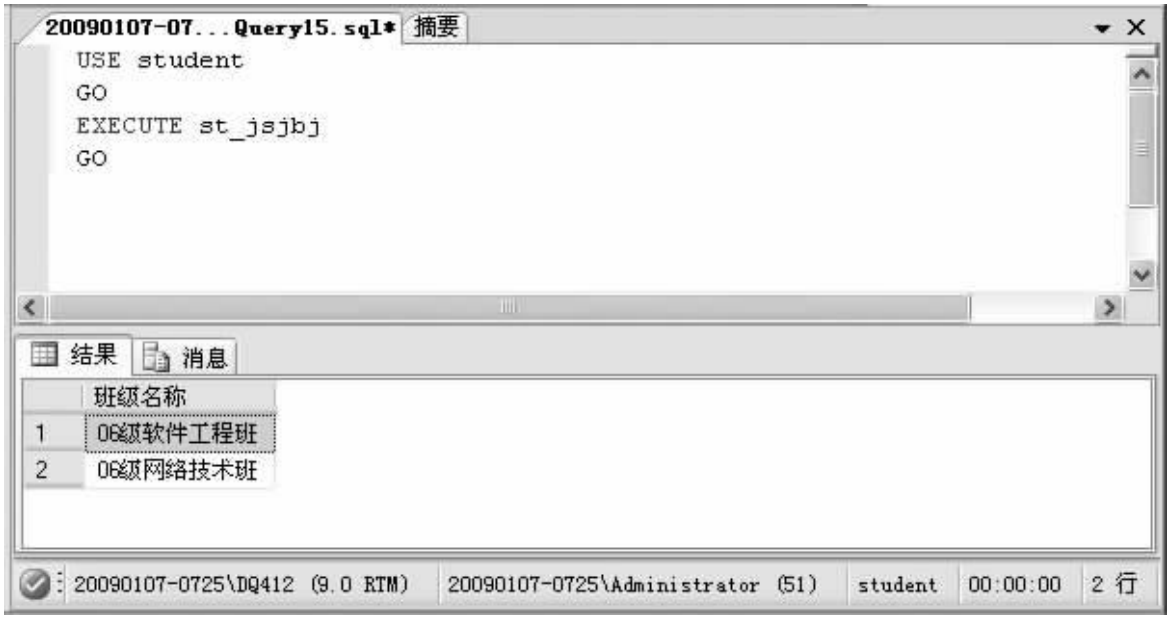

图 10-3　执行存储过程

10.2.3 查看存储过程

存储过程创建好后，其名称保存在系统表 sysobjects 中，其源代码保存在 syscomments 中，再通过 ID 字段进行关联。如果需要查看存储过程的相关信息，可以直接使用系统表，也可以使用系统存储过程，还可以使用 SQL Server 管理平台。

1. 使用系统表查看存储过程信息

【例 10-4】 使用系统表查看 student 数据库中名称为 st_jsjbj 的存储过程的定义信息。代码如下：

```
USE student
GO
SELECT TEXT FROM SYSCOMMENTS
WHERE ID IN (SELECT ID FROM SYSOBJECTS
             WHERE NAME='st_jsjbj' AND XTYPE='P')
GO
```

在编辑器中执行上述代码，结果如图 10-4 所示。

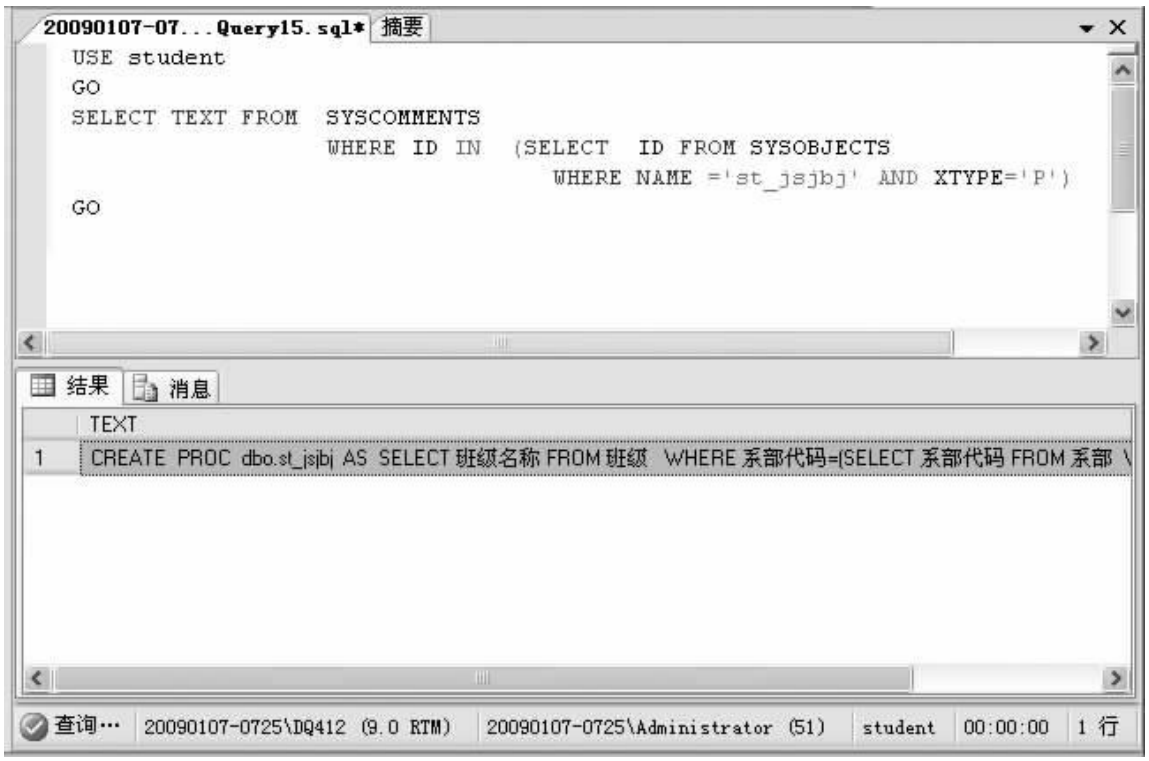

图 10-4 使用系统表查看存储过程信息

2. 使用系统存储过程查看存储过程信息

对用户建立的存储过程，可以使用 SQL Server 2005 提供的系统存储过程来查看其相关信息。下面列举几个常用的系统存储过程：

(1)使用 sp_help 查看存储过程的一般信息，包含存储过程的名称、拥有者、类型和创建时间，其语法格式为：

sp_help 存储过程名

(2)使用 sp_helptext 查看存储过程的定义信息，其语法格式为：

sp_helptext 存储过程名

(3)使用 sp_depends 查看存储过程的相关性,其语法格式为:

sp_depends 存储过程名

【例 10-5】 使用相关的系统存储过程查看 student 数据库中名为 st_jsjbj 的存储过程的定义、相关性及一般信息。

代码如下:

```
USE student
GO
EXEC sp_help st_jsjbj
EXEC sp_helptext st_jsjbj
EXEC sp_depends st_jsjbj
GO
```

执行上述代码返回的结果如图 10-5 所示。

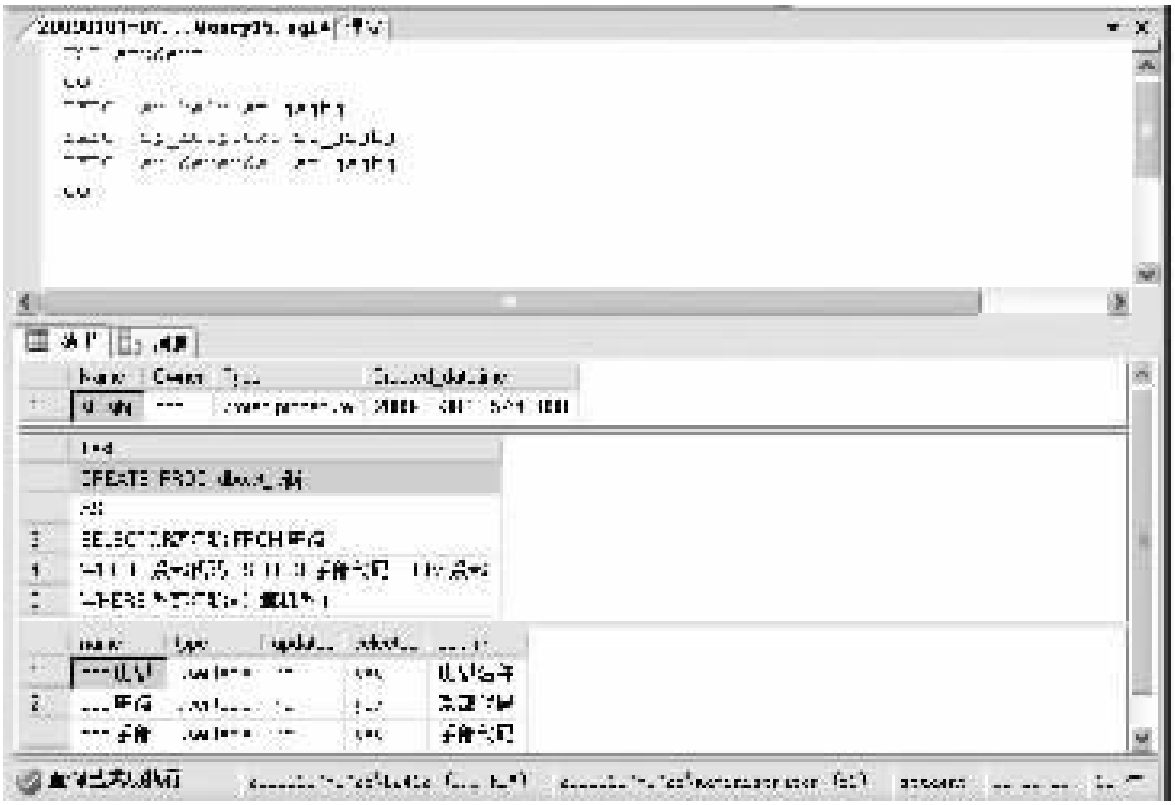

图 10-5　使用系统存储过程查看存储过程信息

3. 使用 SQL Server 管理平台查看存储过程信息

使用 SQL Server 管理平台查看存储过程的相关信息的操作步骤如下:

(1)启动 Microsoft SQL Server Management Studio,在“对象资源管理器”窗口中,依次展开“数据库”→student→“可编程性”→“存储过程”。

(2)在展开的“存储过程”中,右击需要查看的存储资源管理器,在弹出的快捷菜单中选择“属性”选项,打开“存储过程属性”窗口。

(3)选择“选择页”列表框中的“常规”选项,可以查看该存储过程的创建日期、所属的数据库和数据库用户等信息。

(4)选择“选择页”列表框中的“权限”选项,可以查看该存储过程名称,并且可以为该存

储过程添加用户、授予其权限。

(5)选择“选择页”列表框中的“扩展属性”选项,可以向数据库对象添加自定义属性。

10.2.4 修改存储过程

当存储过程所依赖的基本表发生变化或用户有需要时,可以对存储过程的定义或参数进行修改,修改通过执行 CREATE PROCEDURE 语句创建的过程,不会改变权限,也不影响相关的存储过程或触发器。

1.使用 Transact-SQL 语句修改存储过程

修改存储过程的 Transact-SQL 语句是 ALTER PROCEDURE,语法格式如下:

```
ALTER PROC[EDURE]procedure_name [;number]
[{@parameter data_type}[VARYING][=default][=default][OUTPUT]][,…n]
[WITH{RECOMPILE|ENCRYPTION|RECOMPILE,ENCRYPTION}]
[FOR REPLICATION]
AS
sql_statement […n]
```

其中各个参数与创建存储过程中参数的意义相同。

【例 10-6】 修改存储过程 st_jsjbj,使用该存储过程列出经济管理系的班级名称。

代码如下:

```
USE student
GO
ALTER PROC dbo.st_jsjbj
AS
SELECT 班级名称 FROM 班级
WHERE 系部名称=(SELECT 系部代码 FROM 系部
WHERE 系部名称='经济管理系')
GO
```

2.使用 SQL Server 管理平台修改存储过程

使用 SQL Server 管理平台修改存储过程的操作步骤如下:

(1)启动 Microsoft SQL Server Management Studio,在“对象资源管理器”窗口中依次展开“数据库”→student→“可编程性”→“存储过程”。

(2)在“存储过程”中右击需要修改的存储过程,在弹出的快捷菜单中选择“修改”命令,在编辑器中打开该存储过程。

(3)根据需要,修改存储过程。

10.2.5 删除存储过程

当存储过程没有存在的意义时,可以使用 DROP PROCEDURE 语句或 SQL Server 管理平台将其删除。

1.使用 DROP PROCEDURE 语句删除存储过程

DROP PROCEDURE 语句可以一次从当前数据库中将一个或多个存储过程或过程组

删除，其语法格式如下：

```
DROP PROCEDURE 存储过程名称[,…n]
```

【例 10-7】 删除存储过程 st_jsjbj。

代码如下：

```
USE student
GO
DROP PROCEDURE st_jsjbj
GO
```

2. 使用 SQL Server 管理平台删除存储过程

在 SQL Server 管理平台中删除存储过程的操作步骤如下：

(1)启动 Microsoft SQL Server Management Studio，在“对象资源管理器”窗口中，依次展开“数据库”→student→“可编程性”→“存储过程”。

(2)在展开的“存储过程”中，右击需要删除的存储过程，在弹出的快捷菜单中选择“删除”命令。

(3)在弹出的“删除对象”对话框中，单击“确定”按钮，删除该存储过程。

10.3　创建和执行带参数的存储过程

带参数的存储过程可以扩展存储过程的功能。使用输入参数，可以将外部信息传入存储过程；使用输出参数，可以将存储过程的信息传到外部。创建带参数的存储过程时，参数可以是一个，也可以是多个，有多个参数时，参数之间用逗号分隔；所有类型的数据均可以作为存储过程的参数，一般情况下，参数的数据类型要与其相关字段的数据类型一致。

1. 使用输入参数

【例 10-8】 在 student 数据库中，创建一个查询存储过程 st_bjmc，要求该存储过程带一个输入参数，用于接收系部名称。

执行该存储过程时，将根据输入的系部名称列出该系部的班级名称。代码如下：

```
CREATE PROC st_bjmc @xbmc varchar(30)
AS
SELECT 班级名称 FROM 班级
WHERE 系部代码=(SELECT 系部代码 FROM 系部
                WHERE 系部名称=@xbmc)
```

本例中，“系部名称”的数据类型为 varchar，所以定义输入参数@xbmc 的数据类型为 varchar。

执行带参数的存储过程，可以采用以下两种方式：

(1)按位置传递。在调用存储过程时，直接给出参数值。如果多于一个参数，给出的参数值要与参数定义的顺序一致。例如，执行存储过程 st_bjmc，查看“商务技术系”的班级名称，代码如下：

```
EXEC st_bjmc '商务技术系'
```

(2)使用参数名称传递。在调用存储过程时,按“参数名=参数值”的形式给出参数值。采用此方式,参数如果多于一个,给出的参数顺序可以与参数定义的顺序不一致。例如,执行存储过程 st_bjmc 时,查看“商务技术系”的班级名称,代码如下:

```
EXEC st_bjmc @xbmc='商务技术系'
```

在执行存储过程 st_bjmc 时,如果不给出参数值,系统将提示错误。如果希望在不给出参数时显示计算机系的班级名称,可以通过设置参数默认值来实现。

下面对存储过程 st_bjmc 进行修改,实现默认显示计算机系班级的功能。代码如下:

```
ALTER PROC st_bjmc @xbmc varchar(30)='计算机系'
AS
SELECT 班级名称 FROM 班级
WHERE 系部代码=(SELECT 系部代码 FROM 系部
        WHERE 系部名称=@xbmc)
```

执行 st_bjmc,如果不带参数,则显示计算机系的班级名称;如果为其指定参数,则显示指定系部的班级名称。

2.使用输出参数

【例 10-9】 在 student 数据库中,创建一个查询存储过程 st_kcpjf,要求该存储过程带一个输出参数,用于返回 SQL Server 2005 课程的平均分数。

一般情况下,输出参数的数据类型要与它接收的确定值的类型一致。执行该存储过程时,将把 SQL Server 2005 课程的平均分数传递出来。代码如下:

```
USE student
GO
CREATE PROC dbo.st_kcpjf @pjf tinyint OUTPUT
AS
SELECT @pjf=AVG(成绩) FROM 课程注册
WHERE 课程号=(SELECT 课程号 FROM 课程
            WHERE 课程名称='SQL Server 2005')
GO
```

执行带有输出参数的存储过程时,需要声明变量来接收存储过程的返回值。在使用该变量时,还必须为它加上 OUTPUT 声明。一般情况下,声明的变量的数据类型要与存储过程的输出参数的数据类型一致。执行 st_kcpjf 的代码如下:

```
USE student
GO
DECLARE @pj tinyint
EXECUTE st_kcpjf @pj OUTPUT
PRINT 'SQL Server 2005 课程学生的平均分数为'+STR(@pj)
GO
```

执行上述代码,其结果如图 10-6 所示。

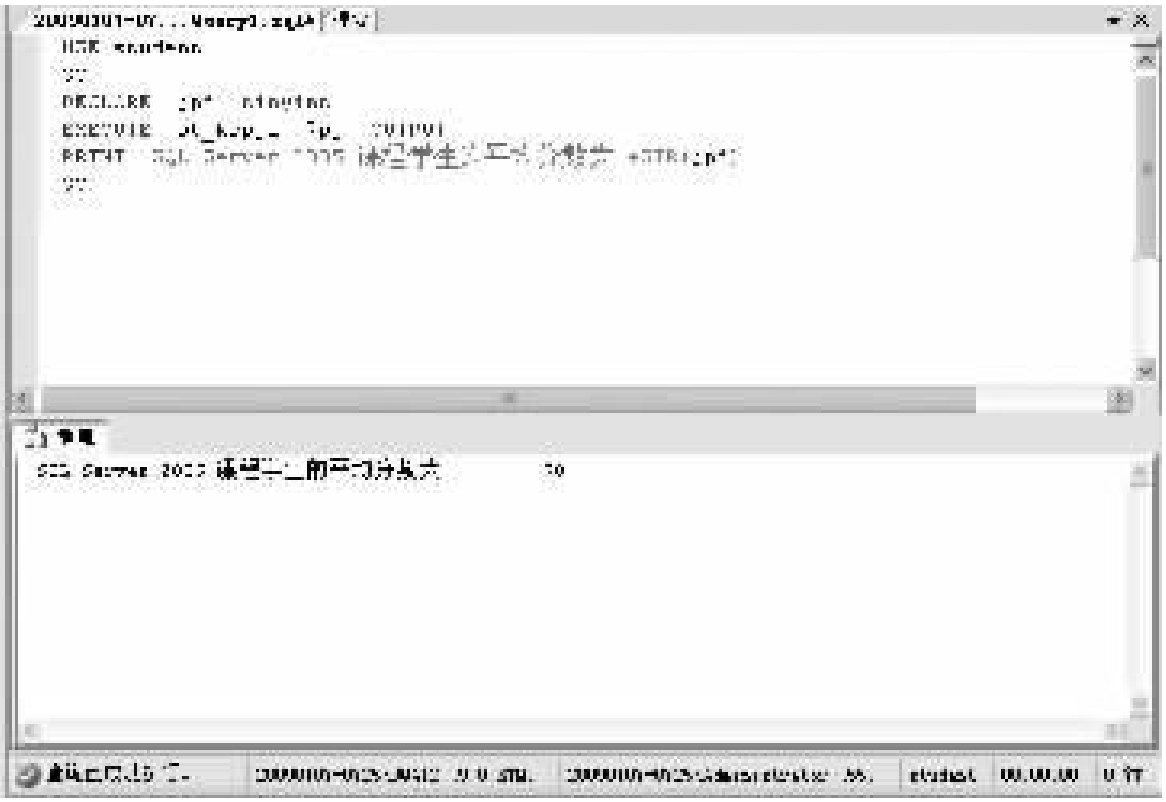

图 10-6　执行含有输出参数的存储过程

3. 使用多个参数

【例 10-10】 在 student 数据库中，创建一个判断存储过程 st_sfyz，要求该存储过程带两个输入参数和一个输出参数，用于判断输入的身份是否正确，并给出相关信息提示。

代码如下：

```
/* 创建存储过程 */
CREATE PROC dbo.sfyz @user varchar(12),@password varchar(12),@zt tinyint OUTPUT
AS
    DECLARE @yhm varchar(12),@mim varchar(12)
    DECLARE yongh CURSOR FOR SELECT 用户名,密码 FROM 管理员
    OPEN yongh
    SET @zt=2
FETCH next FROM yongh
    INTO @yhm,@mim
    IF(@user=@yhm and @password=@mim)
      SET @zt=1
    ELSE
WHILE @@fetch_status=0 and @zt=2
      BEGIN
        FETCH next FROM yongh
        INTO @yhm,@mim
        IF(@user=@yhm and @password=@mim)
        BEGIN
```

```
            SET @zt=1
          END
        END
    CLOSE yongh
    DEALLOCATE yongh
GO
/* 执行存储过程 */
DECLARE @aa tinyint
EXEC sfyz 'lyj','lyj2008',@aa output
IF @aa=1
PRINT '身份验证成功'
ELSE
PRINT '输入的身份不正确'
GO
```

10.4 存储过程的重新编译

存储过程第一次执行后，其被编译的代码将驻留在高速缓存中，当用户再次执行该存储过程时，SQL Server 将其从缓存中调出执行。在使用了存储过程后，有时可能会因为某些原因，必须向表中添加数据列或为表添加索引，从而改变数据库的逻辑结构，此时 SQL Server 不自动执行优化，直到下一次重新启动后，再运行该存储过程。这需要对存储过程进行重新编译，使其优化。SQL Server 提供 3 种重新编译存储过程的方法，分别介绍如下。

1. 在创建存储过程时设定重新编译

在创建存储过程时指定 WITH RECOMPILE 选项，使 SQL Server 在每次执行存储过程时都要重新编译。其语法格式如下：

```
CREATE PROCEDURE procedure_name
WITH RECOMPILE
AS sql_statement
```

当存储过程的参数值在每次执行时都有较大的差异，导致每次均需创建不同的执行计划时，可使用 WITH RECOMPILE 选项。

2. 在执行存储过程时设定重新编译

在执行存储过程时指定 WITH RECOMPILE 选项，可强制对存储过程进行重新编译。其语法格式如下：

```
EXECUTE procedure_name WITH RECOMPILE
```

3. 使用系统存储过程设定重新编译

使用系统存储过程 sp_recompile 强制在下次运行存储过程时进行重新编译。其语法格式如下：

```
EXEC sp_recompile OBJECT
```

其中，OBJECT 是当前数据库中的存储过程、触发器、表或视图的名称。如果 OBJECT 是存储过程或触发器的名称，那么该存储过程或触发器将在下次运行时重新编译；如果 OBJECT 是表或视图的名称，那么所有引用该表或视图的存储过程都将在下次运行时重新编译。

10.5　系统存储过程与扩展存储过程

在 SQL Server 中还有两类重要的存储过程：系统存储过程和扩展存储过程。这些存储过程为用户管理数据库、获取系统信息、查看系统对象提供了很大的帮助。下面简单介绍这两类存储过程。

10.5.1　系统存储过程

SQL Server 2005 提供了上百个系统存储过程，这些系统存储过程可以帮助用户很容易地管理 SQL Server 的数据库。

【例 10-11】　使用 sp_monitor 显示 CPU、I/O 的使用信息。

代码如下：

```
USE master
GO
EXEC sp_monitor
GO
```

返回的结果如图 10-7 所示，该结果显示了当时有关 SQL Server 繁忙程度的信息。

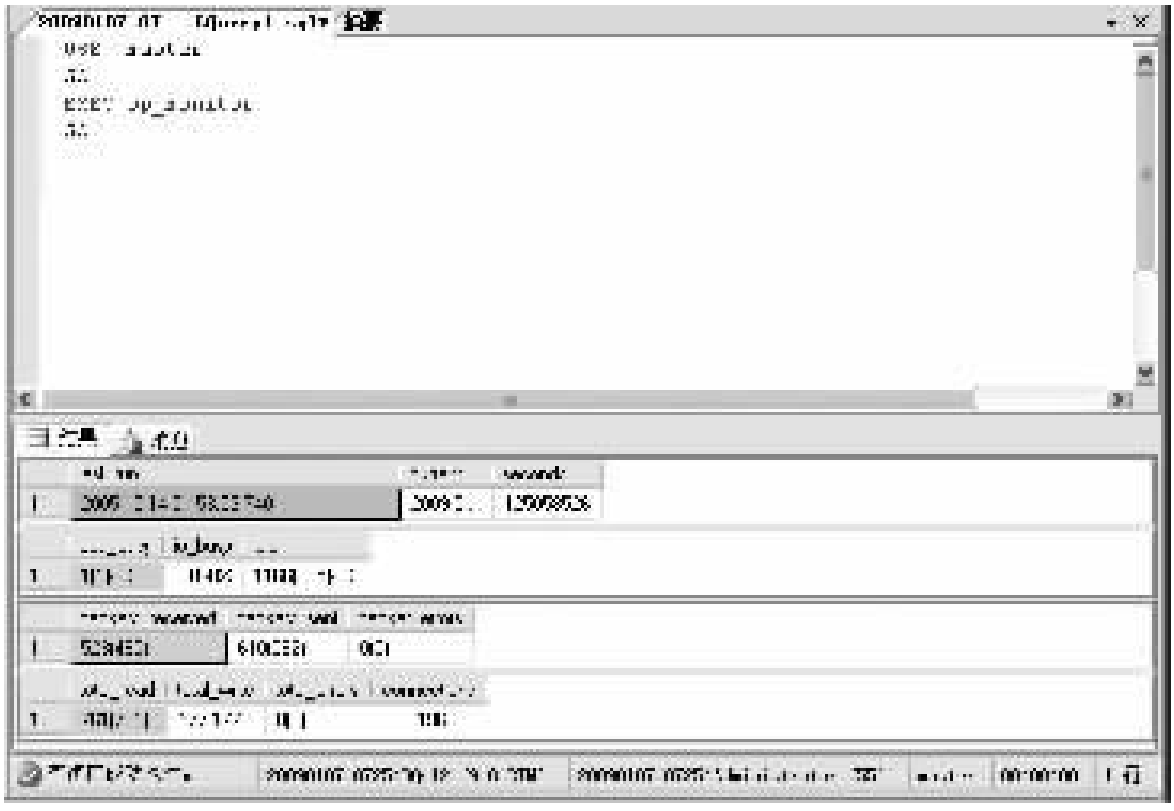

图 10-7　执行 sp_monitor 的结果

10.5.2 扩展存储过程

扩展存储过程是允许用户使用某种编程语言(如 C 语言)创建的应用程序,程序中使用 SQL Server 开放数据服务的 API 函数,它们可以直接在 SQL Server 地址空间中运行。用户可以像使用普通的存储过程一样使用它们,也可以将参数传给它们并返回结果和状态值。

扩展存储过程编写好后,可以由系统管理员在 SQL Server 中注册登记,然后将其执行权限授予其他用户。扩展存储过程只能存储在 master 数据库中。

10.6 案例中的存储过程

1.创建一个加密存储过程

在 student 数据库中,创建一个名称为 st_jiami 的加密存储过程,该过程用来查询一门课程也没有选修的学生的学号与姓名。创建完成后,执行该存储过程。

```
USE student
GO
/* 如果存储过程 st_jiami 存在,将其删除 */
IF EXISTS(SELECT name FROM SYSOBJECTS WHERE name='st_jiami' AND type='P')
   DROP PROCEDURE st_jiami
GO
/* 建立一个加密的存储过程 */
CREATE PROCEDURE st_jiami
/* 加密选项 */
WITH ENCRYPTION
AS
SELECT 学号,姓名 FROM 学生
WHERE 学号 NOT IN (SELECT 学号 FROM 课程注册)
GO
/* 执行 st_jiami */
EXEC st_jiami
GO
```

2.创建带输入参数的存储过程

在 student 数据库中,创建一个带参数的存储过程 st_chengjichaxun。当在该存储过程中输入任意一个成绩时,将从 3 个表(学生表、课程注册表、课程表)中查询出大于或等于该成绩的学生的学号、姓名、课程名和课程成绩。创建完成后,执行该存储过程,查询获得学分且成绩大于或等于 60 的学生。

```
USE student
GO
/* 如果存储过程 st_chengjichaxun 存在,将其删除 */
```

```
IF EXISTS(SELECT name FROM SYSOBJECTS WHERE name='st_chengjichaxun' AND type='P')
DROP PROCEDURE st_chengjichaxun
GO
/* 创建一个带参数的存储过程 st_chengjichaxun */
CREATE PROCEDURE st_chengjichaxun @chengji tinyint
AS
SELECT A.学号,A.姓名,C.课程名称,B.成绩 FROM 学生 AS A
JOIN 课程注册 AS B ON A.学号=B.学号
JOIN 课程 AS C ON B.课程号=C.课程号
WHERE B.成绩>=@chengji
ORDER BY A.学号
GO
/* 执行 st_chengjichaxun,显示获得学分学生的学号、姓名、课程名和课程成绩 */
EXEC st_chengjichaxun 60
GO
```

3. 创建带输出参数的存储过程

在 student 数据库中，创建一个存储过程 st_dkcjfx，当任意输入一个存在的课程名称（本例为 SQL Server 2005）时，该存储过程将统计出该门课程的平均成绩、最高成绩和最低成绩。

```
USE student
GO
/* 如果存储过程 st_dkcjfx 存在,将其删除 */
IF EXISTS (SELECT name FROM SYSOBJECTS WHERE name='st_dkcjfx' AND type='P')
DROP PROCEDURE st_dkcjfx
GO
/* 创建存储过程 st_dkcjfx */
/* 定义 1 个输入参数 kechengming */
/* 定义 3 个输出参数 avgchengji,maxchengji 和 minchengji,用于接收平均成绩、最高成绩和最低成绩 */
CREATE PROCEDURE st_dkcjfx
    @kechengming varchar(20),@avgchengji tinyint OUTPUT,
    @maxchengji tinyint OUTPUT,@minchengji tinyint OUTPUT
AS
SELECT @avgchengji=AVG(成绩),@maxchengji=MAX(成绩),@minchengji=MIN(成绩)
FROM 课程注册 WHERE 课程号 IN
    (SELECT 课程号 FROM 课程 WHERE 课程名称=@kechengming)
GO
/* 执行存储过程 st_dkcjfx */
USE student
```

```
GO
/* 声明 4 个变量,用于保存输入和输出参数 */
DECLARE @kechengming1  varchar(20)
DECLARE @avgchengji1  tinyint
DECLARE @maxchengji1  tinyint
DECLARE @minchengji1  tinyint
/* 为输入参数赋值 */
SELECT @kechengming1='SQL Server 2005'
/* 执行存储过程 */
EXEC st_dkcjfx @kechengming1,@avgchengji1 OUTPUT,
    @maxchengji1 OUTPUT,@minchengji1 OUTPUT
/* 显示结果 */
SELECT @kechengming1 AS 课程名称,@avgchengji1 AS 平均成绩,
@maxchengji1 AS 最高成绩,@minchengji1 AS 最低成绩
GO
```

4.创建添加、删除和修改记录存储过程

在 student 数据库中,创建 P_StudentInfo_Add、P_StudentInfo_Del 和 P_StudentInfo_Update 3 个存储过程,分别用于为学生表添加记录、删除记录和修改记录。

```
/* 添加记录存储过程 */
CREATE PROCEDURE P_StudentInfo_Add
@stuID char(12),
@stuName varchar(8),
@stuSex char(2),
@stuBirthday datetime,
@stuEnterDay datetime,
@stuClassID char(9)
AS
INSERT INTO 学生(学号,姓名,性别,出生日期,入学时间,班级代码)
VALUES(@stuID,@stuName,@stuSex,@stuBirthday,@stuEnterDay,@stuClassID)
GO
/* 删除记录存储过程 */
CREATE PROCEDURE P_StudentInfo_Del
@stuID char(12)
AS
DELETE 学生 WHERE 学生.学号=@stuID
GO
/* 修改记录存储过程 */
CREATE PROCEDURE P_StudentInfo_Update
@stuID char(12),
@stuName varchar(8),
```

```
@stuSex char(2),
@stuBirthday datetime,
@stuEnterDay datetime,
@stuClassID char(9)
AS
UPDATE 学生 SET 学号＝@stuID,姓名＝@stuName,性别＝@stuSex,出生日期＝@stuBirthday,入学时间＝@stuEnterDay,班级代码＝@stuClassID WHERE 学号＝@stuID
GO
```

10.7　触发器概述

触发器(trigger)是一种特殊类型的存储过程。与其他存储过程类似,它也由Transact-SQL语句组成,可以实现一定的功能;不同的是,触发器的执行不能通过名称调用来完成,而是当用户对数据库发生事件(如添加、删除、修改数据)时,将会自动触发与该事件相关的触发器,使其自动执行。触发器不允许带参数,它的定义与表紧密相连,触发器可以作为表的一部分。

在SQL Server 2005中,触发器可分为两大类:DML触发器和DDL触发器。

1.DML触发器

DML触发器是当用户对表或视图执行了INSERT、UPDATE或DELETE操作而被激活的触发器,该类触发器有助于在表或视图中修改数据时强制业务规则、扩展数据完整性。

根据引起触发器激活的时机,DML触发器分为两种类型:AFTER触发器和INSTEAD OF触发器。

(1)AFTER触发器又称为后触发器。引起触发器执行的操作成功完成之后激发该类触发器。如果操作因错误(如违反约束或语法错误)而执行失败,触发器将不会执行。此触发器只能定义在表上,不能定义在视图上。可以为每个触发操作(INSERT、UPDATE和DELETE)创建多个AFTER触发器。如果表上有多个AFTER触发器,可使用sp_settriggerorder定义哪个AFTER触发器最先激发,哪个最后激发。除第一个和最后一个触发器,所有其他的AFTER触发器的激发顺序不确定,并且无法控制。

(2)INSTEAD OF触发器又称为代替触发器。该类触发器代替触发操作执行,即触发器在数据发生变动之前被激发,取代变动数据的操作,执行触发器定义的操作。该类触发器既可在表上定义,也可在视图上定义。对于每个触发操作只能定义一个INSTEAD OF触发器。

DML触发器包含复杂的处理逻辑,能够实现复杂的数据完整性约束。同其他约束相比,它主要有以下优点:

(1)触发器自动执行。系统内部机制可以侦测用户在数据库中的操作,并自动激活相应的触发器,实现相应的功能。

(2)触发器能够对数据库中的相关表实现级联操作。触发器是基于某个表创建的,但可以针对多个表进行操作,实现数据库中相关表的级联操作。

(3)触发器可以实现比CHECK约束更为复杂的数据完整性约束。在数据库中为了实

现数据完整性约束,可以使用 CHECK 约束或触发器。CHECK 约束不允许引用其他表中的列来完成检查工作,而触发器可以引用其他表中的列。例如,在 student 数据库中,向“学生”表中插入记录,当输入系部代码时,必须先检查“系部”表中是否存在该代码的系部。这可以通过触发器实现,而不能通过 CHECK 约束实现。

(4)触发器可以评估数据修改前后表的状态,并根据其差异采取对策。

(5)一个表中可以同时存在 3 种不同操作(INSERT、UPDATE 和 DELETE)的触发器,对于同一个修改语句可以有多个不同的对策响应。

2. DDL 触发器

与 DML 触发器一样,DDL 触发器将激发存储过程以响应事件。但与 DML 触发器不同的是,它们不会被响应针对表或视图的 UPDATE、INSERT 和 DELETE 语句激发,相反,它们会被响应多种数据定义语言(DDL)的语句激发。这些语句主要是以 CREATE、ALTER 和 DROP 开头的语句。DDL 触发器可用于管理任务,如审核和控制数据库操作。执行以下操作时,可以使用 DDL 触发器:

(1)记录数据库架构中的更改或事件。

(2)防止用户对数据库架构进行修改。

(3)希望数据库对数据库架构的更改作出响应。

由于 DDL 触发器和 DML 触发器可以使用相似的 SQL 语法进行创建、修改和删除,它们还具有其他相似的行为。所以下面只介绍 DML 触发器的创建与使用,以下涉及的触发器均是 DML 触发器。

10.8 触发器的创建执行

10.8.1 Inserted 表和 Deleted 表

在创建触发器前,需要了解两个与触发器密切相关的专用临时表:Inserted 表和 Deleted 表。系统为每个触发器创建专用临时表,其表结构与触发器作用的表结构相同。专用临时表被存放在内存中,由系统进行维护,用户可以对其进行查询,不能对其进行修改。触发器执行完成后,与该触发器相关的临时表被删除。

向表中插入数据时,执行 INSERT 操作,此时,系统将自动创建一个与触发器表具有相同表结构的 Inserted 临时表,用来保存新插入的记录。删除表中数据时,执行 DELETE 操作,此时,系统将自动创建一个与触发器表具有相同表结构的 Deleted 临时表,用来保存触发器表中被删除的记录,方便用户查找当前的删除数据。修改表中的数据时,执行 UPDATE 操作,相当于删除一条旧的记录,添加一条新的记录。其中,被删除的记录放在 Deleted 表中,添加的新记录放在 Inserted 表中。

10.8.2 创建触发器

创建触发器可以使用 Transact-SQL 语句,也可以使用 SQL Server 管理平台。在创建触发器前,必须注意以下几点:

(1)CREATE TRIGGER 必须是批处理中的第一条语句,并且只能应用到一个表中。

(2)触发器只能在当前的数据库中创建,但可以引用当前数据库的外部对象。

(3)表的所有者具有创建触发器的默认权限,且不能将该权限转给其他用户。

(4)不能在临时表或系统表上创建触发器,触发器可以引用临时表但不能引用系统表。

(5)如果指定了触发器架构名称来限定触发器,则将以相同的方式限定表名称。

(6)如果一个表的外键包含对定义的 DELETE/UPDATE 操作的级联,则不能在该表上定义 INSTEAD OF DELETE/UPDATE 触发器。

在创建触发器时,必须指明在哪一个表上定义触发器及触发器的名称、激活时机和激活触发器的语句(INSERT、UPDATE 或 DELETE)。

1. 使用 Transact-SQL 语句创建触发器

使用 Transact-SQL 语句创建触发器的语法格式如下:

```
CREATE TRIGGER trigger_name
ON{table|view}
[WITH ENCRYPTION]
{
  {{FOR|AFTER|INSTEAD OF}{[INSERT][,][DELETE][,][UPDATE]}
    [NOT FOR REPLICATION]
AS
[{IF UPDATE(column)
  [{AND|OR}UPDATE(column)]
    […n]
|IF(COLUMNS_UPDATED(){bitwise_operator}updated_bitmask)
    {comparison_operator}updated_bitmask[…n]
}]
sql_statement[…n]}
}
```

参数说明:

- trigger_name:触发器名称,必须符合标识符命名规则,并且在当前数据库中唯一。
- table|view:用于定义触发器的表或视图。
- WITH ENCRYPTION:对 CREATE TRIGGER 语句文本进行加密。
- AFTER:默认的触发器类型,后触发器。此类型触发器不能定义在视图上。
- INSTEAD OF:表示建立替代类型的触发器。
- NOT FOR REPLICATION:表示当复制进程更改触发器所涉及的表时,不执行该触发器。
- IF UPDATE:指定对表中字段进行添加或修改内容时起作用,不能用于删除操作。
- sql_statement:定义触发器被触发后,将执行的 Transact-SQL 语句。

【例 10-12】 在 student 数据库中,为“班级”表建立一个名为 del_banji 的 DELETE 触发器,其作用是当删除“班级”表中的记录时,检查“学生”表中是否存在该班级的学生。如果

存在,则提示不允许删除该班级的信息。

代码如下:

```
USE student
GO
IF EXISTS (SELECT name FROM sysobjects
           WHERE name='del_banji' AND type='TR')
DROP TRIGGER del_banji
GO
CREATE TRIGGER del_banji ON 班级
FOR DELETE
AS
DECLARE @banjidaima char(9)
SELECT @banjidaima=班级代码 FROM Deleted
IF EXISTS (SELECT * FROM 学生 WHERE 班级代码=@banjidaima)
BEGIN
PRINT '班级正在使用,不能被删除!'
ROLLBACK TRANSACTION
END
GO
```

【例 10-13】 删除“班级”表中的班级代码为“060101001”的“06 级软件工程 001 班”,观察触发器 del_banji 的作用。

代码如下:

```
USE student
GO
DELETE 班级 WHERE 班级代码='060101001'
GO
```

结果为:

```
班级正在使用,不能被删除!
```

2. 使用 SQL Server 管理平台创建触发器

在 SQL Server 管理平台中创建触发器的操作步骤如下:

(1)启动 Microsoft SQL Server Management Studio,在“对象资源管理器”窗口中,依次展开“数据库”→student→“表”。

(2)在“表”中,展开需要建立触发器的表(如班级),右击触发器,在弹出的快捷菜单中选择“新建触发器”选项。

(3)在打开的创建触发器模板中输入触发器创建文本。如图 10-8 所示。

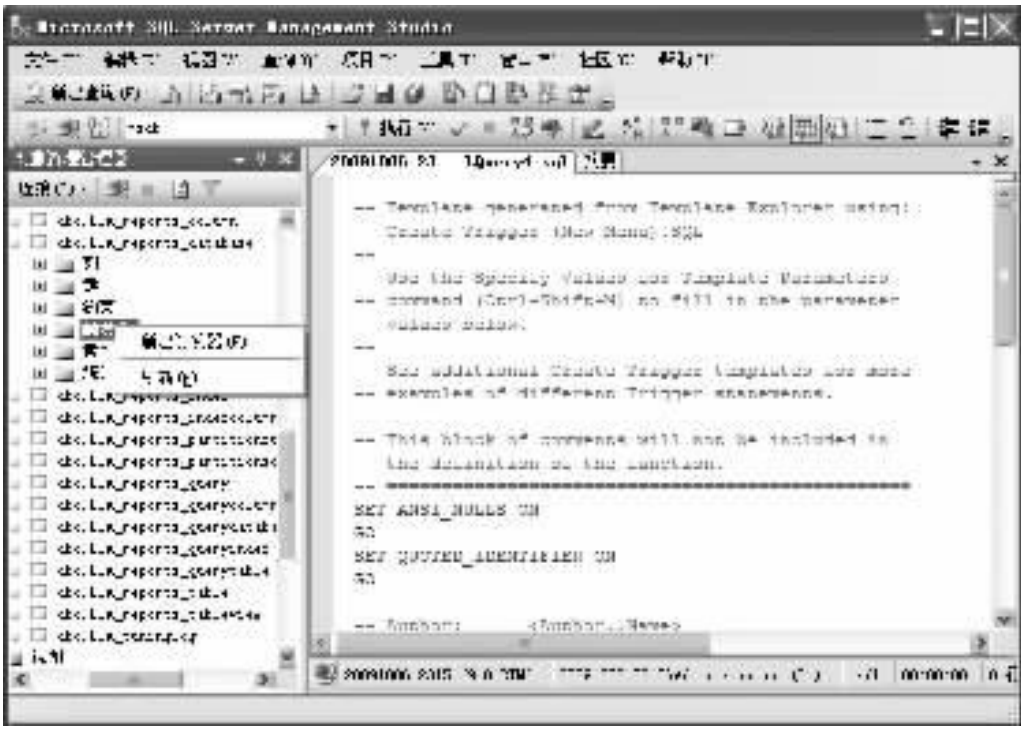

图 10-8　创建触发器模板

(4)单击工具栏上的“执行”按钮,完成触发器的创建。

10.8.3　查看触发器信息

触发器创建好后,其名称保存在系统表 sysobjects 中,其源代码保存在 syscomments 中。如果需要查看触发器信息,既可以使用系统存储过程,也可以使用 SQL Server Management Studio。

触发器是特殊的存储过程,能用于查看存储过程的系统存储过程都适用于触发器。可以使用 sp_help 查看触发器的一般信息,使用 sp_depends 查看触发器的依赖关系。

【例 10-14】 使用系统存储过程 sp_helptrigger 查看班级表中的触发器的信息。

代码如下:

```
USE student
GO
EXEC sp_helptrigger 班级
GO
```

结果如图 10-9 所示。

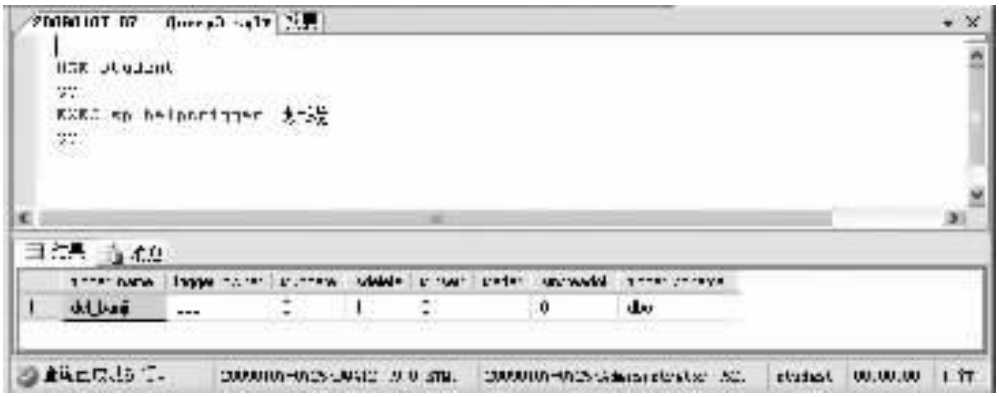

图 10-9　查看触发器信息

10.9 触发器的操作

10.9.1 修改触发器

对于建立好的触发器,可以根据需要对其名称及文本进行修改。通常使用系统存储过程对其进行更名操作,使用 SQL Server 管理平台或 Transact-SQL 语句修改其文本。

1.修改触发器名称

对触发器进行更名,可以使用系统存储过程 sp_rename 来完成,其语法格式如下:

[EXECUTE]sp_rename 触发器原名,触发器新名

2.修改触发器定义

修改触发器的定义,可以使用 ALTER TRIGGER 语句。ALTER TRIGGER 语句与 CREATE TRIGGER 语句的语法相似,只是语句的第一个关键字不同。

3.修改触发器文本

在 SQL Server 管理平台中修改触发器文本的操作步骤如下:

(1)启动 Microsoft SQL Server Management Studio,在“对象资源管理器”窗口中,依次展开“数据库”→student→“表”(如班级)→“触发器”。

(2)在“触发器”中,右击需要修改的触发器(如 del_banji),在弹出的快捷菜单中选择“修改”命令。

(3)根据需要,修改触发器。

10.9.2 禁止、启用和删除触发器

对于创建的触发器,如果暂时不用,可以禁止其执行。触发器被禁止执行后,对表进行数据操作时,不会激活与数据操作相关的触发器。当需要触发器时,可以再启动它。如果触发器没有存在的必要,则可以将其删除。对于触发器的这些操作,可以使用 Transact-SQL 语句实现,也可以使用 SQL Server 管理平台实现。

1.使用 Transact-SQL 语句实现

禁止触发器的语法格式有以下两种:

- DISABLE TRIGGER 触发器名称 ON 表名
- ALTER TABLE 表名 DISABLE TRIGGER 触发器名称

启用触发器的语法格式有以下两种:

- ENABLE TRIGGER 触发器名称 ON 表名
- ALTER TABLE 表名 ENABLE TRIGGER 触发器名称

删除触发器的语法格式如下:

DROP TRIGGER{触发器名称}[,…n]

2.使用 SQL Server 管理平台实现

(1)启动 Microsoft SQL Server Management Studio,在“对象资源管理器”窗口中,依次

展开“数据库”→student→“表”(如班级)→“触发器”。

(2)在“触发器”中,右击需要修改的触发器,在弹出的快捷菜单上选择相应的命令。

10.10　嵌套触发器

在触发器中可以包含影响另外一个表的 INSERT、UPDATE 或 DELETE 语句,这就是嵌套触发器。具体说就是,如果 A 上的触发器在执行时引发了表 B 上的触发器,而表 B 上的触发器又激活了表 C 上的触发器,表 C 上的触发器又激活了表 D 上的触发器……所有触发器依次触发。这些触发器不会形成无限循环,因为 SQL Server 规定触发器最多可嵌套至 32 层。如果允许使用嵌套触发器,且链中的一个触发器开始一个无限循环,如果超出嵌套级,触发器将被终止执行。正确地使用嵌套触发器,可以执行一些有用的操作,但嵌套触发器比较复杂,使用时要注意技巧。例如,由于触发器在事务中执行,如果在一系列嵌套触发器的任意层中发生错误,则整个事务都将被取消,且所有的数据修改都将回滚。一般情况下,在触发器中包含 PRINT 语句,用来确定错误发生的位置。

在默认情况下,系统允许嵌套。可以使用 sp_configure 系统存储过程修改是否允许嵌套。其语法格式如下:

```
EXEC sp_configure 'nested trigger',0|1
```

其中,如果设置为 0,则允许嵌套;设置为 1,则禁止嵌套。

10.11　案例中的触发器

1. 创建一个 INSERT 触发器

在 student 数据库中建立一个名为 insert_xibu 的 INSERT 触发器,存储在“专业”表中。向“专业”表中插入记录时,如果插入了在“系部”表中没有的系部代码,则表示不能插入记录,否则提示记录插入成功。

```
USE student
GO
IF EXISTS (SELECT name FROM sysobjects
          WHERE name='insert_xibu' AND type='TR')
    DROP TRIGGER insert_xibu
GO
CREATE TRIGGER insert_xibu ON [dbo].[专业]
FOR INSERT
AS
    DECLARE @XIBU CHAR(2)
    SELECT @XIBU=系部.系部代码
    FROM 系部,inserted
    WHERE 系部.系部代码=inserted.系部代码
```

```
IF @XIBU<> ''
    PRINT('记录插入成功')
ELSE
BEGIN
    PRINT('系部代码不存在系部表中,不能插入记录,插入将终止!')
END
GO
```

2. 创建一个 DELETE 触发器

在 student 数据库中建立一个名为 delete_zhye 的 DELETE 触发器,存储在“专业”表中。删除“专业”表中的记录时,如果“班级”表引用了此记录的专业代码,则提示用户不能删除记录,否则提示记录已删除。

```
USE student
GO
IF EXISTS(SELECT name FROM sysobjects
        WHERE name='delete_zhye' AND type='TR')
    DROP TRIGGER delete_zhye
GO
CREATE TRIGGER delete_zhye
ON 专业
FOR DELETE
AS
    IF(SELECT COUNT(*) FROM 班级 INNER JOIN DELETED
    ON 班级.专业代码=DELETED.专业代码)>0
BEGIN
    PRINT('该专业被班级表所引用,不可以删除此条记录,删除将终止')
    ROLLBACK TRANSACTION
END
ELSE
    PRINT('此条记录已删除')
GO
```

3. 创建一个 UPDATE 触发器

在 student 数据库中建立一个名为 update_zymc 的 UPDATE 触发器,存储在“专业”表中。更新“专业”表中的专业名称时,提示不能修改专业名称。

```
USE student
GO
IF EXISTS(SELECT name FROM sysobjects
        WHERE name='update_zymc' AND type='TR')
        DROP TRIGGER update_zymc
CREATE TRIGGER update_zymc
```

```
ON [dbo].[专业]
FOR UPDATE
AS
IF UPDATE(专业名称)
BEGIN
    PRINT('不能修改专业名称')
    ROLLBACK TRANSACTION
END
GO
```

本 章 小 结

本章重点介绍了存储过程和触发器的概念、创建和执行。存储过程是对数据库中的任意表或者多个表进行复杂的数据处理而设计的，需要用户指定它去运行，它才会执行里面的 SQL 代码。而触发器是针对某个表进行添加、修改等操作作出的响应，是不需要用户干预而自动运行的。使用触发器是利用 Transact-SQL 语言进行程序设计的较高技术和技巧，需要熟练掌握。

习　题　10

1. 什么是存储过程？存储过程有什么特点？
2. 什么是触发器？触发器有什么特点？
3. 使用触发器有哪些好处？
4. 触发器有哪几种类型？
5. 创建存储过程时，应该注意什么？
6. 创建触发器时，应该注意的事项有哪些？
7. 创建存储过程有哪些方法？执行存储过程的命令是什么？用哪个命令可以删除存储过程？
8. 查看存储过程和触发器信息的系统存储过程有哪些？