目　录

第一部分　基础型实验

基础型实验以 SQL Server 2008 的基础应用和 T-SQL 语言的应用为主，通过 8 个实验，使得学生能熟练掌握数据库的创建及管理、数据表的创建及管理、各种查询方法以及视图和索引的应用等，为数据库的高级应用开发打下坚实的基础。

每个实验由五个部分组成：实验目的、实验内容、相关知识、实验指导、扩展练习，实验 8 不包含扩展练习。

实验 1　SQL Server 2008 的安装及配置

一、实验目的

- 了解 SQL Server 2008 不同版本安装的硬件和软件要求；
- 熟悉 SQL Server 2008 的安装步骤；
- 了解 SQL Server 2008 的主要组件；
- 掌握启动、暂停和停止 SQL Server 2008 服务的方法；
- 掌握 SQL Server Management Studio 对象资源管理器的使用方法；
- 了解 SQL Server 2008 支持的身份验证模式；
- 掌握 sa 用户的开启和设置密码的方法。

二、实验内容

- 完成 SQL Server 2008 的安装；
- 启动、暂停和停止 SQL Server 2008 服务；
- 连接 SQL Server Management Studio，并了解每个部件的基本功能；
- 开启 sa 用户，并设置其密码。

三、相关知识

SQL Server 2008 是一个全面的数据库平台，是新一代的数据管理和分析解决方案。

它为企业用户提供了一个安全、可靠和高效的平台，用于企业数据管理和商业智能应用。

1. SQL Server 2008 的版本

根据数据库应用环境的不同，SQL Server 2008 分为以下 7 个版本。

(1)企业版(Enterprise Edition)

企业版是一个全面的数据管理和业务智能平台，为关键业务应用提供了企业级的可扩展性、数据仓库、安全、高级分析和报表支持。这一版本将为用户提供更加坚固的服务器和执行大规模在线事务处理。

(2)标准版(Standard Edition)

标准版是一个完整的数据管理和业务智能平台，为部门级应用提供了最佳的易用性和可管理特性。

(3)工作组版(Workgroup Edition)

工作组版是一个值得信赖的数据管理和报表平台，用以实现安全的发布、远程同步和对运行分支应用的管理能力。这一版本拥有核心的数据库特性，可以很容易地升级到标准版或企业版。

(4)Web 版(Web Edition)

Web 版是针对运行于 Windows 服务器中要求高可用并面向 Internet Web 服务的环境而设计的。这一版本为实现低成本、大规模、高可用性的 Web 应用或客户托管解决方案提供了必要的支持工具。

(5)精简版(Express Edition)

Express 版是 SQL Server 的一个免费版本，它拥有核心的数据库功能，其中包括了 SQL Server 2008 中最新的数据类型，但它是 SQL Server 的一个微型版本。这一版本是为了学习、创建桌面应用和小型服务器应用而发布的，也可供 ISV 再发行使用。

(6)开发版(Developer Edition)

开发版允许开发人员构建和测试基于 SQL Server 的任意类型应用。这一版本拥有所有企业版的特性，但只限于在开发、测试和演示中使用。基于这一版本开发的应用和数据库可以很容易地升级到企业版。

(7)压缩版(Compact Edition)

压缩版是针对开发人员而设计的免费嵌入式数据库，这一版本的意图是构建独立、仅有少量连接需求的移动设备、桌面和 Web 客户端应用。SQL Server Compact 可以运行于所有的微软 Windows 平台之上，包括 Windows XP 和 Windows Vista 操作系统以及 Pocket PC 和 Smart Phone 设备。

2. SQL Server 2008 的主要组件

(1)数据库引擎(SQL Server Database Engine，SSDE)

数据库引擎是 SQL Server 2008 的核心组件，负责完成业务数据的存储、处理、查询

和安全管理等操作。例如：创建数据库、创建表、执行各种数据查询、访问数据库等操作都是由数据库引擎完成的。在大多数情况下，使用数据库系统实际上就是使用数据库引擎。

(2)分析服务(SQL Server Analysis Server，SSAS)

分析服务提供了多维分析和数据挖掘功能，可以支持用户建立数据库和进行商业智能分析。另外，通过使用SSAS，用户可以完成数据挖掘模型的构造和应用，实现知识发现、知识表示、知识管理和知识共享。

(3)报表服务(SQL Server Reporting Services，SSRS)

报表服务为用户提供了支持Web的企业级的报表功能，用户可以方便地定义和发布满足自己需求的各种形式的报表。

(4)集成服务(SQL Server Integration Services，SSIS)

集成服务是一个数据集成和数据转换平台，可以完成有关数据的提取、转换、加载等。

四、实验指导

1. 检查软/硬件配置是否达到了SQL Server 2008的安装要求

为了正确安装和运行SQL Server 2008，计算机必须满足以下要求。

(1)硬件要求

处理器：Pentium兼容处理器或更高速处理器，处理器速度1.0 GHz以上，建议2.0 GHz或者更快。

内存：1 GB以上，建议2 GB或者更大。

硬盘：至少2 GB安装空间以及必要的数据预留空间。

(2)软件要求

软件要求是指在安装SQL Server 2008各种版本时必须安装的操作系统，应安装Windows XP及以上版本，此外安装时还需安装一系列组件，如NET Framework、SQL Server Native Client、SQL Server安装程序支持文件等。

2. 安装SQL Server 2008

在Win 7操作系统中，安装SQL Server 2008中文企业版(评估版)的步骤如下(其他版本的安装方法类似)。

第1步：首先从官网上下载SQL Server 2008的安装文件，并把下载好的原始文件解压出来，在解压的文件中双击setup.exe，进入“SQL Server安装中心”窗口，单击左边菜单栏中的“安装”选项卡，在窗口右边将列出可以进行的安装方式，如图1-1-1所示。单击“全新SQL Server独立安装或向现有安装添加功能”选项将安装全新的SQL Server 2008。

图 1-1-1 “SQL Server 安装中心”窗口

第 2 步：安装程序将检查 SQL Server 安装程序支持文件时可能发生的问题，并将检查信息显示在“安装程序支持规则”窗口中，如图 1-1-2 所示。如果有检查未通过的规则，必须进行更正，否则安装无法继续。

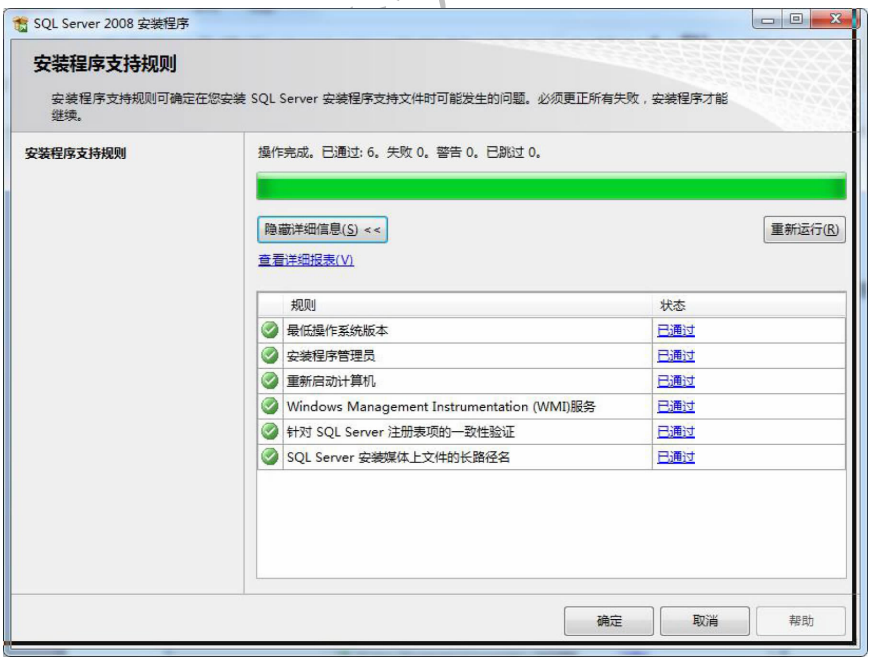

图 1-1-2 “安装程序支持规则”窗口

第 3 步：安装程序支持规则全部通过后单击“确定”按钮，进入“产品密钥”窗口，如图

1-1-3 所示。在“指定可用版本”选项中选择“Enterprise Evaluation”，在“输入产品密钥”选项中输入评估版的 25 位产品密钥，完成后，单击“下一步”按钮。

图 1-1-3　“产品密钥”窗口

第 4 步：进入“许可条款”窗口，阅读并接受许可条款，单击“下一步”按钮，进入“安装程序支持文件”窗口。单击“安装”按钮，安装 SQL Server 必备组件，安装完成后重新进入“安装程序支持规则”窗口，如图 1-1-4 所示。如果通过，则单击“下一步”按钮。

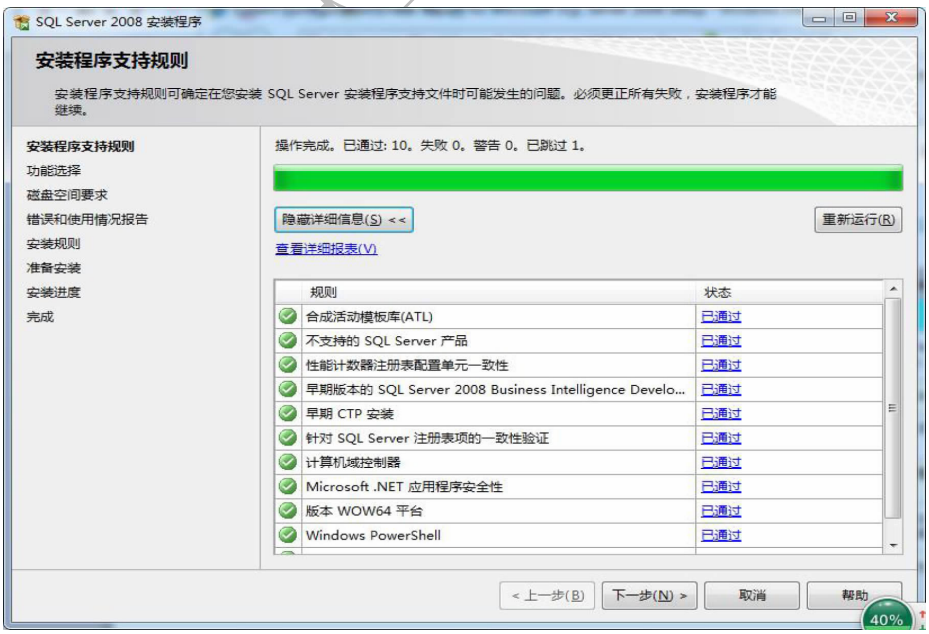

图 1-1-4　安装程序支持规则检查

第 5 步：进入“功能选择”窗口，如图 1-1-5 所示。用户可以根据自身需求来安装组件，初学者可默认全选，若要精简安装，选择数据库引擎服务、客户端工具连接、管理工具（基本和完整）即可，单击“下一步”按钮，进入“实例配置”窗口。

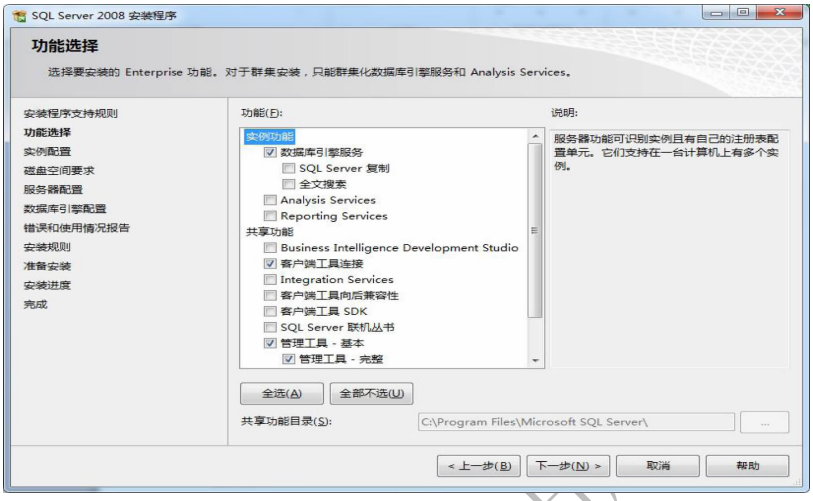

图 1-1-5 “功能选择”窗口

第 6 步：在“实例配置”窗口中进行实例配置。操作系统每一个登录账户下都可以安装一个全新的 SQL Server 数据库系统，每一次安装就是安装一个 SQL Server 的实例。如果是第一次安装，则既可以使用默认实例，也可以自行指定实例名称。如果当前服务器上已经安装了一个默认的实例，则再次安装时必须指定一个实例名称。自定义实例名的方法为，选择“命名实例”单选按钮，在后面的文本框中输入自定义的实例名称。如果选择“默认实例”，则实例名称默认为 MSSQLSERVER。这里选择“默认实例”，如图 1-1-6 所示。

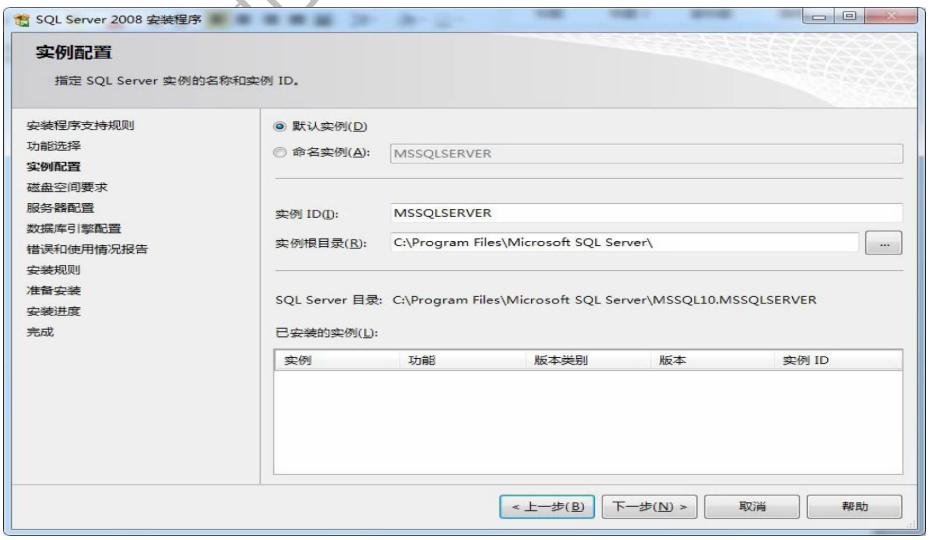

图 1-1-6 “实例配置”窗口

第 7 步：单击“下一步”按钮，进入“磁盘空间要求”窗口，窗口中将显示安装 SQL Server 2008 所需要的磁盘空间，单击“下一步”按钮，进入“服务器配置”窗口，指定哪些账户可以使用 SQL Server 数据库系统中的服务，并提供账户对应的密码。这些账户可以是数据库系统中自带的账户，也可以是操作系统中的账户，如果设置为操作系统中的账户，建议每一项服务都设置为同一个账户，以免不能同时使用多项服务(默认是设置为数据库系统中内置的一些账户，这种设置下，无论用户使用哪一个操作系统账户登录操作系统，均能使用这些服务)，如图 1-1-7 所示。

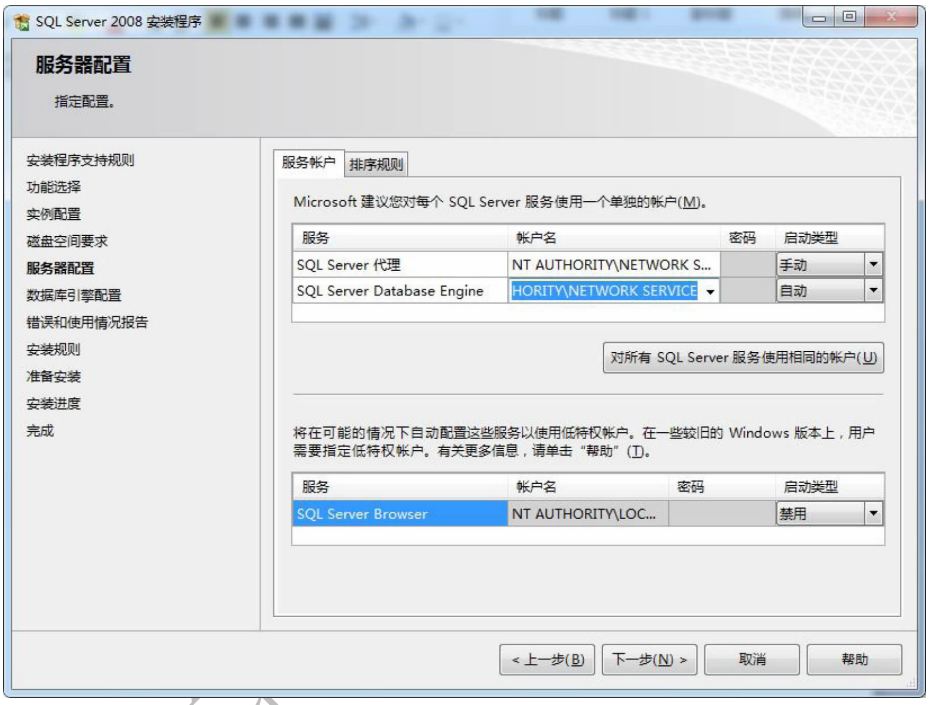

图 1-1-7　“服务器配置”窗口

第 8 步：配置完成后，单击“下一步”按钮，进入“数据库引擎配置”窗口，在“账户”设置选项卡中选择身份验证模式。身份验证模式是一种安全模式，用于验证客户端与服务器的连接，它有两种模式：Windows 身份验证模式和混合模式。在 Windows 身份验证模式中，用户通过 Windows 账户连接时，使用 Windows 操作系统中的信息验证账户名和密码；混合模式允许用户使用 Windows 身份验证或 SQL Server 身份验证进行连接，而建立连接后，系统的安全机制对于两种连接方式是一样的。

本书将身份验证模式设置为“混合模式”，并为内置的系统管理员账户“sa”设置密码“123456”，如图 1-1-8 所示。另外，必须为 SQL Server 实例指定至少一个 SQL Server 管理员，单击“添加当前用户”按钮，添加当前 Windows 账户为 SQL Server 管理员，若要添加其他账户可以单击“添加”按钮。

第 9 步：单击“下一步”按钮，进入“错误和使用情况报告”窗口，用户可根据需求在复

选框中选择选项。

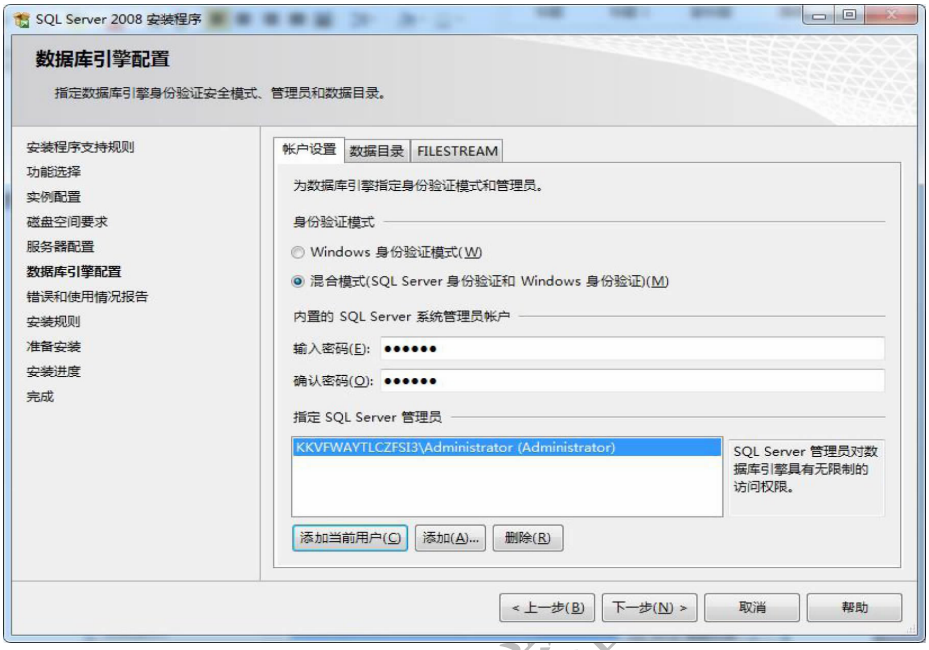

图 1-1-8　设置“身份验证模式”

第 10 步：单击“下一步”按钮，进入“安装规则”窗口，窗口中将显示安装规则的通过情况，如图 1-1-9 所示。如果全部通过，则可以单击“下一步”按钮。

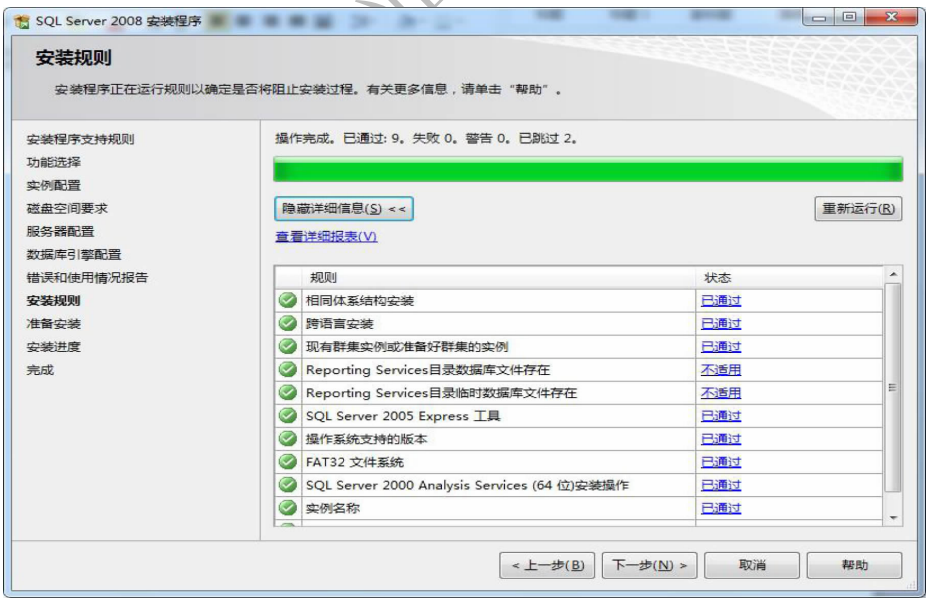

图 1-1-9　“安装规则”窗口

第 11 步：进入“准备安装”窗口，单击“安装”按钮开始安装，等待一段时间后，安装

完成，窗口中将显示已经成功安装的功能组件，如图 1-1-10 所示。单击“下一步”，在“完成”窗口中单击“关闭”按钮结束安装。

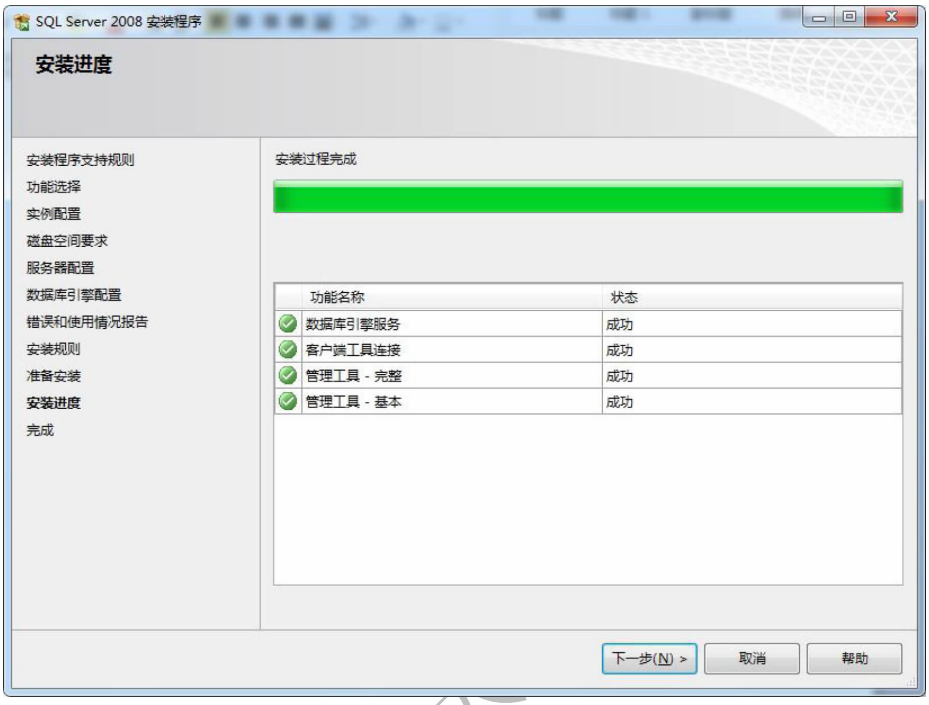

图 1-1-10　“安装进度”窗口

3. 启动、暂停和停止 SQL Server 2008 服务

(1)使用 SQL Server 配置管理器

打开“开始”菜单，依次选择“所有程序”→“Microsoft SQL Server 2008”→“配置工具”→“SQL Server 配置管理器”，打开“SQL Server Configuration Manager”，单击“SQL Server 服务”选项，在右侧栏中可以看到本地所有的 SQL Server 服务，选中“SQL Server (MSSQLSERVER)”，右键单击选择“启动”“停止”或“暂停”服务即可，如图 1-1-11 所示。

图 1-1-11　在配置管理器中启动、停止或暂停 SQL Server 2008 服务

(2)使用 NT 服务器命令行

在“运行”中输入“cmd”命令进入 DOS 命令窗口，接着就可以使用 net start mssqlserver、net stop mssqlserver、net pause mssqlserver 命令分别启动、停止和暂停 SQL Server 服务，如图 1-1-12 所示。

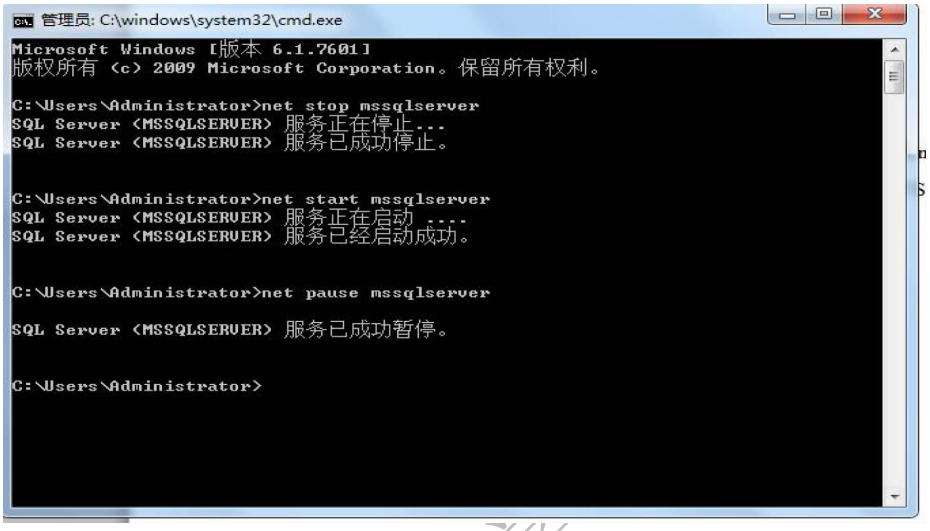

图 1-1-12　使用命令启动、停止或暂停 SQL Server 2008 服务

(3)通过控制面板中的服务对话框

单击[控制面板]→[管理工具]→[服务]→[SQL Server(MSSQLSERVER)]控制服务器的启动、暂停和停止。

4. 启动 SQL Server Management Studio

①打开“开始”菜单，依次选择“所有程序”→“ Microsoft SQL Server 2008”→“SQL Server Management Studio”，打开 SQL Server Management Studio 窗口。

②弹出“连接到服务器”界面，如图 1-1-13 所示。在“服务器类型”中选择“数据库引擎”；服务器名称默认是安装时的实例名，如果安装时采用的是“默认实例”，则服务器名为本地计算机名，在“服务器名称”下拉框中选择“浏览更多”，可以选择运行本地服务或是远程服务，对于本地服务，输入的服务器名称可以是默认实例，也可输入“localhost”或“.”；身份验证方式若选择“Windows 身份验证”，直接单击“连接”按钮即可，若选择“SQL Server 身份验证”，则需要已经开启 sa 账户，并为其设置密码。

③单击“连接”按钮后，如果连接服务器成功，在左侧的“对象资源管理器”中会显示该服务器信息。若连接失败，则会提示错误信息，极有可能的原因是服务器没启动或服务器名称错误。

5. 开启 sa 用户，设置其密码

若 sa 用户登录失败，除密码错误、服务未启动外，也可能是未开启 sa 用户，开启 sa

用户的方法如下。

图 1-1-13　“连接到服务器”界面

①首先以 Windows 身份验证方式登录。

②在“对象资源管理器”中，右击要设置的服务器名称，选择“属性”命令，弹出“服务器属性”对话框，单击“安全性”，在“服务器身份验证”中选择“SQL Server 和 Windows 身份验证模式”，如图 1-1-14 所示。单击“确定”按钮，即可启动混合验证模式，即开启 sa 用户。

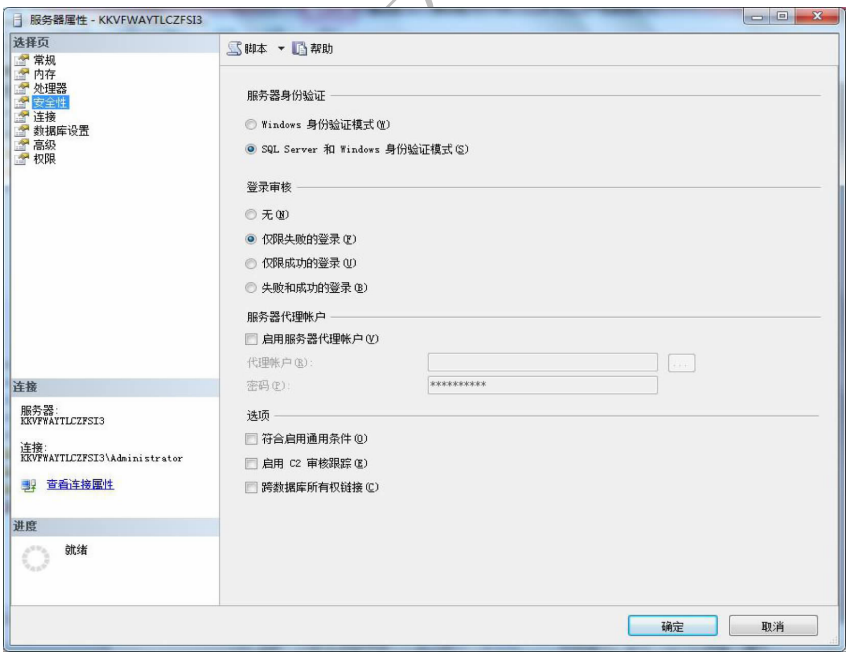

图 1-1-14　启动混合验证模式

③回到“对象资源管理器”，依次展开“安全性”→“登录名”，右击“sa”，选择“属性”命

令，在“常规”页上，勾选“强制实施密码策略”，输入 sa 用户的密码，并再次输入确认密码，如图 1-1-15 所示；在“状态”页上，设置登录为“启用”，如图 1-1-16 所示，最后单击“确定”按钮。

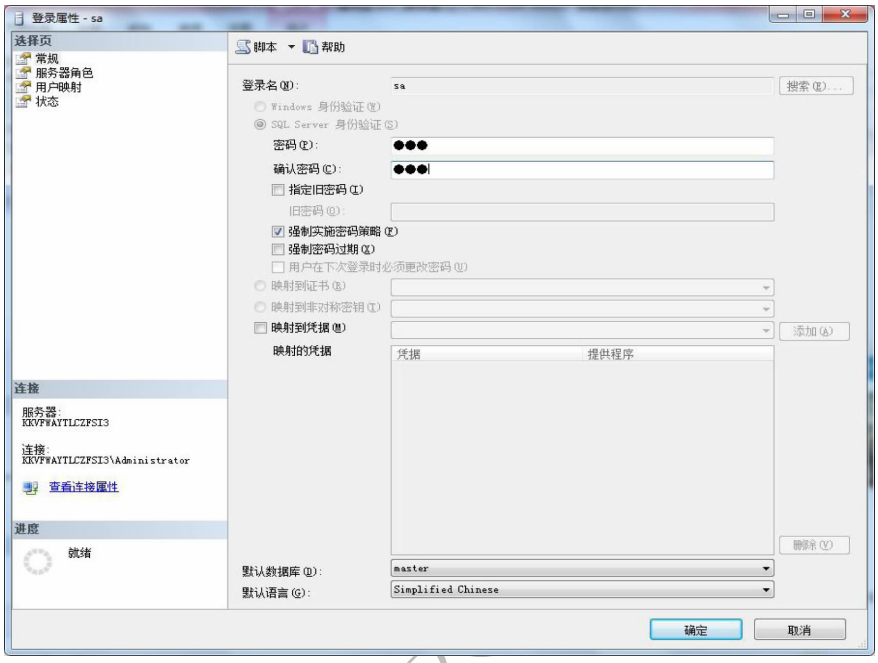

图 1-1-15　设置 sa 用户的密码

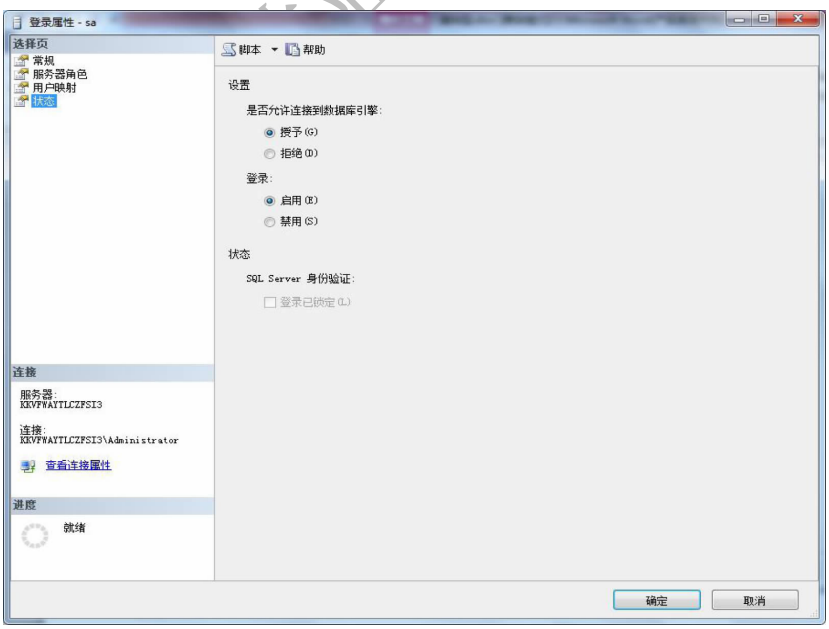

图 1-1-16　启用 sa 用户

五、扩展练习

①练习安装 SQL Server 2008。

②在操作系统的“控制面板”→“管理工具”→“服务”中查看安装成功的 SQL Server 服务，并启动、暂停或停止服务。

③启动 sa 用户，修改其密码后，用 sa 用户登录。

实验 2　创建及管理数据库

一、实验目的

- 了解 SQL Server 2008 数据库的逻辑结构和物理结构；
- 熟练掌握用 SQL Server Management Studio 创建数据库的方法；
- 熟练掌握用 T-SQL 语句创建数据库的方法；
- 熟练掌握数据库属性的设置；
- 掌握修改和删除数据库的方法。

二、实验内容

- 用 SQL Server Management Studio 创建、修改和删除数据库；
- 用 T-SQL 语句创建、修改和删除数据库。

三、相关知识

1. 数据库的逻辑结构

SQL Server 的数据库是存储数据的容器，由若干个用户可视的对象组成，如表、视图、索引、约束、默认值、规则、触发器、存储过程、用户定义数据类型、用户定义函数等。

SQL Server 2008 数据库分为两类，即系统数据库和用户数据库。系统数据库是系统安装时自动创建的，包括 master、model、msdb、tempdb，用来存储有关 SQL Server 的信息，SQL Server 使用系统数据库来管理系统。用户数据库由用户自己创建，可以创建一个或多个用户数据库。

2. 数据库的物理结构

数据库的物理结构指的是保存在数据库中的各种逻辑对象是如何存储在磁盘上的，数

据库以文件的形式存储在磁盘上，SQL Server 2008 将数据库映射为一组操作系统文件，文件分为三类。

(1)主数据文件

主数据文件简称主文件，用来存放数据库的启动信息和部分或全部数据，它是所有数据库文件的起点，包含指向其他数据库文件的指针，默认扩展名是 .mdf。每个数据库都必须包含且只能包含一个主文件。

(2)辅助数据文件

辅助数据文件用来存储主文件中没有存储的其他数据，默认扩展名为 .ndf。辅助文件是可选的，一般当数据库很大时，有可能需要创建多个辅助文件。而当数据库较小时，则只需要创建主文件而不需要创建辅助文件。

(3)事务日志文件

事务日志文件简称日志文件，用于存放恢复数据库所需的事务日志信息。每个数据库至少有一个日志文件，也可以有多个，日志文件的默认扩展名为 .ldf。

创建数据库时，一个数据库至少包含一个主数据文件和一个日志文件，也可以包含辅助数据文件，默认状态下，数据库文件存放在 \ MSSQL \ DATA 目录下，主数据文件名为“数据库名_Data.mdf”，日志文件名为“数据库名_Log.ldf”。可在创建数据库时指定存储路径和文件名，也可添加多个辅助数据文件和日志文件。

为了便于分配和管理数据，可以为一个磁盘驱动器创建一个文件组，将多个数据文件集合起来。使用文件组可以提高表中数据的查询性能，SQL Server 2008 中有两类文件组：主文件组和用户定义文件组。

主文件组包含了所有的系统表、主数据文件和未指定组的其他文件。当新建一个数据库时，会自动创建主文件组，且被设为默认组。用户定义的文件组是用户自己创建的。

每个数据库中都有一个文件组作为默认文件组运行，可以设置某个用户定义的文件组为默认文件组，这样在创建数据库对象时，如果没有指定将其存放在哪个文件组中，将会把它放在默认文件组中。如果没有指定默认文件组，则主文件组为默认文件组。

一个数据库中可创建多个文件组，一个数据文件只能属于一个文件组，一个文件组也只能被一个数据库使用。日志文件是独立的，它不属于任何文件组。

3. 用 T-SQL 创建数据库的语法

```
create database database_name
[on
[PRIMARY][ <filespec>[, …n] ][ <filegroupspec>[, …n] ]
[LOG ON {<filespec>[, …n] }]
]
```

其中，<filespec>:: =

```
{ (
name = logical_file_name,
filename = {'os_file_name' | 'filestream_path'}
[, SIZE = size[KB | MB | GB | TB]]
[, MAXSIZE = {max_size[KB | MB | GB | TB] | UNLIMITED}]
[, FILEGROWTH = grow_increment[KB | MB | GB | TB | % ]]
)[, …n]
}
<filegroup>::= 
{FILEGROUP filegroup_name [DEFAULT]
<filespec>[, …n]
}
```

说明：用[ ]括起来的表示可选；[, …n]表示重复前面的内容；用{ }括起来的表示必选；A | B格式表示只能选择A和B的中一个，DEFAULT关键字表示设置文件组为默认文件组。

四、实验指导

创建图书管理数据库TSGL(包含一个主数据文件和一个日志文件)，主数据文件的逻辑名为“TSGL”，初始大小为4 MB，最大为40 MB，按10%的增长方式增长；日志文件的逻辑名为“TSGL_log”，初始大小为1 MB，最大文件大小不受限制，按1 MB增长。主数据文件的存放路径为D盘根目录，日志文件的存放路径为E盘根目录。

1. 利用SQL Server Management Studio(SSMS)创建数据库

第1步：启动SSMS，连接到数据库服务器，在“对象资源管理器”中，右击“数据库”，在弹出的快捷菜单中选择“新建数据库”命令，弹出“新建数据库”对话框。

第2步：“新建数据库”窗口左侧有三个选项卡——“常规”“选项”和“文件组”，这里只配置“常规”选项卡，其他选项卡使用系统默认配置。在“数据库名称”文本框中输入数据库名TSGL，可以通过“所有者”文本框后面的“…”按钮设置数据库的所有者，如sa，一般就用默认值。单击“路径”列中的“…”按钮，可以设置数据文件或日志文件的存放位置，如图1-2-1所示。

注意：在默认情况下，数据库的数据文件和日志文件保存在同一磁盘中，这不是最佳的方案，为了提高读取数据的速度，建议将数据文件和日志文件存放在不同的磁盘上。

第3步：根据要求设置主数据文件和日志文件的初始大小，自动增长用于在文件容量不足时设置文件依据何种增长方式增长。单击“自动增长”列中的“…”按钮，可以设置数据文件或日志文件是否自动增长、增长方式、最大文件大小。本例主数据文件的自动增长设

置如图 1-2-2 所示，日志文件的自动增长设置如图 1-2-3 所示。

图 1-2-1 “新建数据库”对话框

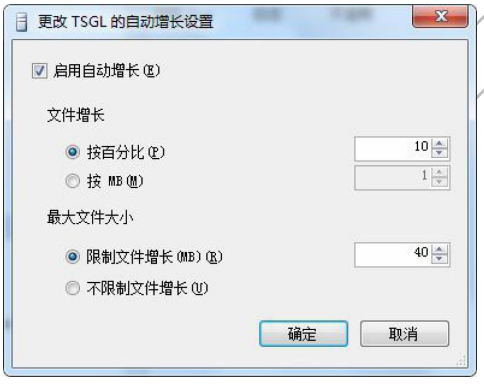

图 1-2-2 主数据文件自动增长设置

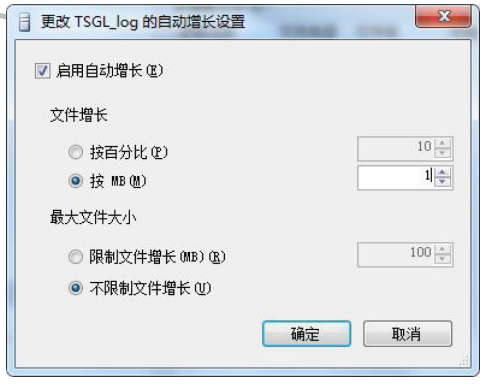

图 1-2-3 日志文件自动增长设置

第 4 步：完成上述操作后，单击“新建数据库”窗口的“确定”按钮，在对象资源管理器的“数据库”目录下，就可以看到新创建的数据库 TSGL 了。

2. 利用 SQL Server Management Studio(SSMS)修改和删除数据库

数据库创建成功后，数据文件名和日志文件名不能改变，但可以对数据库进行以下修改：增加或删除辅助数据文件；改变数据文件或日志文件的大小和增长方式；增加或删除日志文件；重命名数据库。

选择需要修改的数据库 TSGL，右击鼠标，在弹出的快捷菜单中选择“属性”菜单项，出现“数据库属性-TSGL”窗口，它包含 9 个选项卡，如图 1-2-4 所示，在“名称”文本框输

入新的数据库名即可重命名数据库。

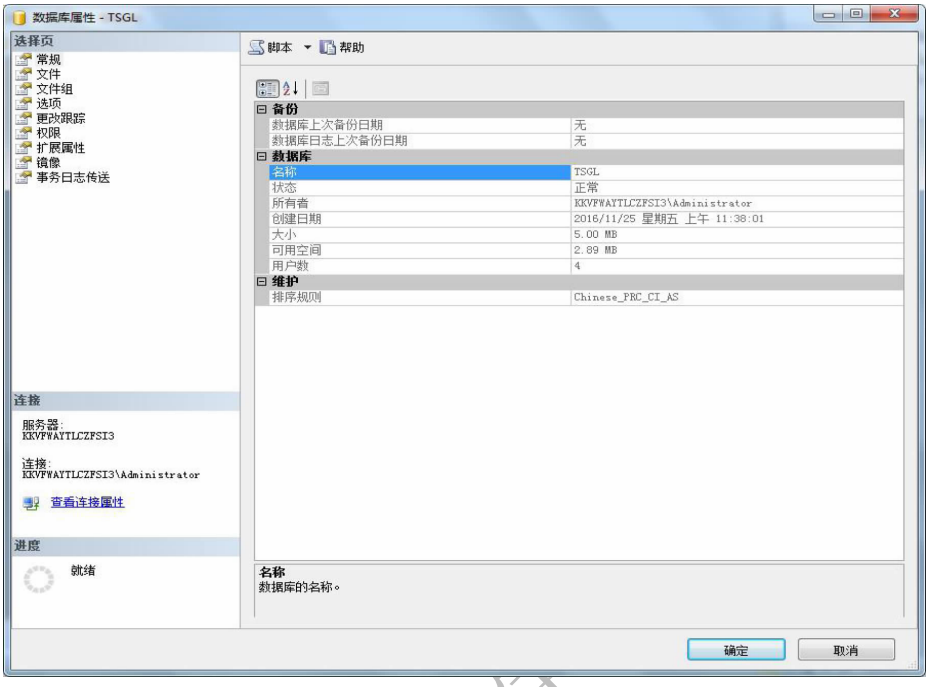

图 1-2-4　“数据库属性-TSGL”窗口

(1)修改数据文件或日志文件的大小和增长方式

单击“数据库属性-TSGL”窗口左侧的“文件”选项卡，可以修改数据文件或日志文件的大小和增长方式，如图 1-2-5 所示。

图 1-2-5　修改数据文件或日志文件的大小和增长方式

(2)添加辅助数据文件

向 TSGL 数据库中添加辅助数据文件 TSGL_1，其属性均用系统默认值。单击“数据库属性-TSGL”窗口左侧的“文件”选项卡，单击右下角的“添加”按钮，数据库文件下方会增加一行文件项，在“逻辑名称”列中输入 TSGL_1，在“文件类型”列中选择“行数据”，其他属性采用默认设置，单击“确定”按钮，完成辅助数据文件的添加，其扩展名为“.ndf”，如图 1-2-6 所示。

图 1-2-6　添加辅助数据文件

(3)添加文件组

在 TSGL 数据库中添加一个名为 Group 的文件组，单击“数据库属性-TSGL”窗口左侧的“文件组”选项卡，单击右下角的“添加”按钮，在 PRIMARY 行的下方会增加一行，在该行的“名称”列输入 Group，单击“确定”按钮，如图 1-2-7 所示。若要将新增的文件组设置为默认文件组，可选中“默认值”列的复选框。

图 1-2-7　添加文件组

添加了文件组后，就可以在新文件组中加入数据文件了。例如，为新增的文件组 Group 添加数据文件 TSGL_1 的操作方法如下：选择“文件”选项卡，在文件 TSGL_1 所在行的“文件组”列中选择 Group，如图 1-2-8 所示，单击“确定”按钮。

图 1-2-8　将数据文件添加到新增的文件组

(4)删除文件组

删除文件组的操作方法如下：选中需要删除的文件组，单击右下角的“删除”按钮，再

单击“确定”按钮即可。

注意：只能删除用户定义文件组，不能删除主文件组 PRIMARY。

(5)删除数据库

若数据库不再需要，可以删除它以便释放其占用的资源，在对象资源管理器中选择要删除的数据库“TSGL”，右击鼠标，在弹出的快捷菜单中选择“删除”命令，弹出“删除对象”对话框，单击“确定”按钮即可。

3. 利用 T-SQL 语句创建数据库

使用 T-SQL 语句创建图书管理数据库 TSGL，主数据文件的逻辑名为“TSGL”，初始大小为 4 MB，最大为 40 MB，按 10%的增长方式增长，存储路径为“D:\TSGL.mdf”；日志文件的逻辑名为“TSGL_log”，初始大小为 1 MB，最大文件大小不受限制，按 1 MB 增长，存储路径为“E:\TSGL_log.ldf”。

```
CREATE DATABASE TSGL
ON PRIMARY
(NAME = TSGL,
FILENAME = 'D:\TSGL.mdf',
SIZE = 4MB,
MAXSIZE = 40MB,
FILEGROWTH = 10 % )
LOG  ON
(NAME = TSGL_log,
FILENAME = 'E:\TSGL_log.ldf',
SIZE = 1MB,
MAXSIZE = UNLIMITED,
FILEGROWTH = 1MB)
```

4. 利用 T-SQL 语句修改和删除数据库

①在 TSGL 数据库中添加一个文件组 Grp1。

```
ALTER  DATABASE  TSGL
ADD  FILEGROUP  Grp1
```

②为 TSGL 数据库添加一个辅助数据文件 TSGL_1，存储路径为“D:\TSGL_1.ndf”，初始大小默认，最大文件大小为 50 MB，每次增长 5 MB，并将其添加到文件组 Grp1 中。

```
ALTER DATABASE TSGL
ADD FILE(
NAME = TSGL_1,
```

```
FILENAME = 'D:\TSGL_1.ndf',
MAXSIZE = 50MB,
FILEGROWTH = 5MB)
TO  FILEGROUP  Grp1
```

③将辅助数据文件“TSGL_1”的最大大小改为不受限制。

```
ALTER DATABASE TSGL
MODIFY FILE(
NAME = TSGL_1,
MAXSIZE = UNLIMITED)
```

④删除辅助数据文件“TSGL_1”。

```
ALTER  DATABASE TSGL
REMOVE FILE TSGL_1
```

⑤删除文件组“Grp1”。

```
ALTER  DATABASE TSGL
REMOVE FILEGROUP Grp1
```

注意：空文件组才能被删除，否则必须先删除该文件组包含的文件，再删除该文件组，主文件组不能被删除。

⑥将图书管理数据库“TSGL”的名字修改为“TSGL1”。

```
ALTER DATABASE TSGL
MODIFY NAME = TSGL1
```

⑦删除数据库 TSGL1。

```
DROP DATABASE TSGL1
```

五、扩展练习

①用 T-SQL 语句创建数据库 JXGL，它包含 1 个主数据文件和 2 个日志文件，其中主数据文件的逻辑名为“JXGL”，初始大小为 4 MB，最大为 40 MB，按 10%的增长方式增长，存储路径为“D:\JXGL.mdf”；第 1 个日志文件的逻辑名为“JXGL_log”，初始大小为 2 MB，最大值不受限制，按 1 MB 增长，存储路径为“E:\JXGL_log.ldf”；第 2 个日志文件的逻辑名为“JXGL_log1”，存储路径为“E:\JXGL_log1.ldf”，其他属性取默认值。

②在 JXGL 数据库中添加一个文件组 g1。

③在 JXGL 数据库中添加一个辅助数据文件 JXGL_1，存储路径为“D:\JXGL_

1. ndf”，初始大小为 10 MB，最大值不受限制，按 5 MB 增长，并将其添加到文件组 g1 中。

④删除数据库 JXGL。

实验 3　数据表的创建及管理

一、实验目的

- 熟悉各种数据类型；
- 了解六种数据完整性约束；
- 熟练掌握使用 T-SQL 语句创建及管理数据表。

二、实验内容

- 使用 T-SQL 语句创建数据表并添加完整性约束；
- 使用 T-SQL 语句修改和删除数据表。

三、相关知识

1. 数据类型

数据类型决定了数据的存储格式，代表了各种不同的信息类型。定义表的属性时需要指明其数据类型及长度。SQL Server 中的数据类型可分为系统内置数据类型和用户自定义数据类型两种。系统内置数据类型是 SQL Server 预先定义好的，可以直接使用。常用数据类型如下。

(1)ASCII 字符型

- char(n)：定长字符数据类型。当列定义为 char(n)类型时，若实际存储的串长度不足 n，则在串的尾部添加空格以达到长度 n，所以 char(n)的长度为 n。例如，某列的数据类型为 char(6)，而输入的字符串为“abcd”，则存储的是字符 abcd 和 2 个空格。若输入的字符个数超出了 n，则超出的部分被截断。

- varchar(n)：变长字符数据类型，存储大小为输入数据的实际长度，n 表示字符串可达到的最大长度。例如，某列的数据类型为 varchar(6)，而输入的字符串为“abcd”，则存储的就是字符 abcd，长度为 4 个字节。

(2)Unicode 字符型

Unicode 是统一字符编码标准，用于支持国际上非英语语种的字符数据的存储和

处理。

• nchar(n)：存放固定长度的 n 个 Unicode 字符数据，n 的值为 1～4 000，长度为 2n 字节。

• nvarchar(n)：最多存放长度可变的 n 个 Unicode 字符数据，n 的值为 1～4 000，长度是所输入字符个数的两倍。

备注：char 一个字母占一个字节，汉字占两个字节；nchar 所有字符都占 2 个字节。例如：char(8)只能存 8 个字母或 4 个汉字；nchar(8)则能存 8 个汉字。

(3)文本型

文本型包括 text 和 ntext 两种类型，分别对应 ASCII 字符和 Unicode 字符。

(4)整型

整型包括 bigint(大整型)、int(整型)、smallint(短整型)和 tinyint(微短整型)，它们表示数的范围逐渐缩小。

(5)精确数值型

精确数值型数据由整数和小数部分构成，包括 demical 和 numeric 两种，在 SQL Server 2008 中，这两类数据类型功能一样。

声明精确数值型的格式是 demical(p，d)或 numeric(p，d)，其中 p 为精度(可存储数据的最大位数为 p，不包含符号和小数点)，d 为小数位数。例如，若某列的数据类型为 numeric(5，3)，表示小数部分为 3 位，整数部分不能超过 2 位，当向某记录的该列输入 88.8888 时，该列实际储存的是 88.889；若输入 8888.88，则提示出错，因为整数部分超过了 2 位。

注意：声明精确数值型数据时，其小数位数必须小于精度。在给精确数值型数据赋值时，必须使赋值的整数部分不大于列的整数部分的长度。

(6)浮点型

浮点型也称近似数值型，使用它来存储数据时，可能会损失一些精度。浮点型有两种：float 和 real，两者都使用科学计数法表示数据。

2. 六种约束类型

约束是通过限制列中数据、行中数据和表之间数据来保证数据完整性的非常有效的方法。约束可以确保把有效的数据输入列中，维护表和表之间的特定关系。其中列约束是针对表中一个列的约束，表约束是针对表中一个或多个列的约束。SQL Server 2008 提供了六种约束类型。

(1)PRIMARY KEY(主键)约束

在表中定义一个主键值，这是唯一确定表中每一行数据的标识符，该约束强制实体完整性。一个表中最多只能有一个主键，主键列不允许取空值。

(2)UNIQUE(唯一值)约束

UNIQUE 约束指定表中某一个列或多个列不能有相同的两行或两行以上的数据存在。这种约束通过实现唯一性索引来强制实体完整性。当表中已经有了一个主键约束时，如果需要在其他列上实现实体完整性，又因为表中不能有两个或两个以上的主键约束，所以只能通过创建 UNIQUE 约束来实现。设置 UNIQUE 约束的列允许空值，但是设置主键约束的列不允许空值。

(3)CHECK(检查)约束

CHECK 约束通过指定的逻辑表达式来限制用户当输入某一列的数据时，只能在指定范围内。

(4)DEFUALT(默认值)约束

当插入数据时，如果没有为某一个列指定数据，那么 DEFAULT(默认值)约束就在该列中输入一个默认值。

(5)FOREIGN KEY(外键)约束

外键约束定义一个或多个列，这些列可以引用同一个表或另外一个表中的主键约束列。实际上，通过创建外键约束可以实现表与表之间的依赖关系。

(6)NULL/NOT NULL(空值/非空值)约束

如果表中的某列被指定具有 NOT NULL 属性，则在插入数据时，该列值不允许为空，默认情况列值具有 NULL 属性，允许为空。

3. 使用 T-SQL 语句创建表的语法

```
CREATE TABLE＜表名＞

(＜列名＞ ＜数据类型＞[列级完整性约束条件]
[，＜列名＞ ＜数据类型＞[列级完整性约束条件]]
[，＜表级完整性约束条件＞])
```

4. 使用 T-SQL 语句修改表的语法

```
ALTER TABLE＜表名＞

[ ADD＜新列名＞ ＜数据类型＞[ 完整性约束条件 ]]
[ ADD＜表级完整性约束＞]
[ DROP COLUMN＜列名＞ ]
[ DROP CONSTRAINT＜完整性约束名＞]
[ ALTER COLUMN＜列名＞ ＜数据类型＞ ]
```

5. 使用 T-SQL 语句删除表的语法

```
DROP TABLE＜表名＞
```

四、实验指导

在图书管理数据库 TSGL 中，使用 T-SQL 语句创建 4 个表，其结构如表 1-3-1 至表 1-3-4所示。

表 1-3-1　读者类型表 ReaderType 的结构

字段名	字段含义	数据类型	允许为空	备注
ReaderTypeID	读者类型编号	INT	N	主键
TypeName	类型名称	VARCHAR(15)	N	
LimitNum	限借数量	INT	N	
BorrowTerm	借阅期限(月)	INT	N	

表 1-3-2　读者表 Reader 的结构

字段名	字段含义	数据类型	允许为空	备注
ReaderID	读者编号	CHAR(6)	N	主键
ReaderName	读者姓名	VARCHAR (8)	N	
Sex	性别	CHAR(2)	Y	只能取值男或女
ReaderTypeID	读者类型编号	INT	N	外键，参照 ReaderType 表的主键
BorrowedNum	已借数量	TINYINT	Y	默认值为 0
Dept	所在系部	VARCHAR (20)	Y	
Tel	电话	CHAR(11)	N	取值唯一

表 1-3-3　图书表 Book 的结构

字段名	字段含义	数据类型	允许为空	备注
BookID	图书编号	CHAR(4)	N	主键
ISBN	ISBN	VARCHAR (15)	N	
BookName	图书名称	VARCHAR (20)	N	
Author	作者	VARCHAR (8)	Y	
Publisher	出版社	VARCHAR (20)	Y	
PublisherDate	出版日期	DATE	Y	
Price	价格	NUMERIC(4，1)	Y	
SNum	库存量	INT	N	默认值为 0

表 1-3-4　借阅表 Lend 的结构

字段名	字段含义	数据类型	允许为空	备注
LendID	借阅流水号	INT(自动增长)	N	主键
ReaderID	读者编号	CHAR(6)	N	外键，参照 Reader 表的主键
BookID	图书编号	CHAR (4)	N	外键，参照 Book 表的主键
LendDate	借阅日期	DATE	N	
BackDate	归还日期	DATE	Y	

用 T-SQL 语句完成以下操作：

①创建读者类型表 ReaderType。

```
USE TSGL
CREATE TABLE ReaderType(
ReaderTypeID INT PRIMARY KEY,                    /*列级完整性约束条件*/
TypeName VARCHAR(15) NOT NULL,
LimitNum INT NOT NULL,
BorrowTerm INT NOT NULL )
```

②创建读者表 Reader。

```
USE TSGL
CREATE TABLE Reader(
ReaderID CHAR(6)PRIMARY KEY,                     /*列级完整性约束条件*/
ReaderName VARCHAR (8) NOT NULL,
Sex CHAR(2) CHECK (Sex in('男', '女')),          /*列级完整性约束条件*/
ReaderTypeID INT NOT NULL FOREIGN KEY
REFERENCES ReaderType(ReaderTypeID),
BorrowedNum TINYINT DEFAULT 0,                   /*列级完整性约束条件*/
Dept VARCHAR (20),
Tel CHAR(11) NOT NULL UNIQUE)
```

上例未指定约束名，系统会自动生成约束名，除 NULL/NOT NULL 约束外，其他约束都可以定义一个约束名，用 CONSTRAINT<约束名> 来定义。NOT NULL 和 DEFUALT 只能是列级约束，其他约束既可以是列级，也可以是表级约束。因此上例等价于：

```
USE TSGL
CREATE TABLE Reader(
ReaderID CHAR(6),
```

```
ReaderName VARCHAR (8) NOT NULL,
Sex CHAR(2),
ReaderTypeID INT NOT NULL,
BorrowedNum TINYINT CONSTRAINT DF_BN DEFAULT 0,
Dept VARCHAR (20),
Tel CHAR(11) NOT NULL,
CONSTRAINT PK_RID PRIMARY KEY(ReaderID),                    /*表级完整性约束条件*/
CONSTRAINT FK_RTID FOREIGN KEY(ReaderTypeID)
REFERENCES ReaderType(ReaderTypeID),                        /*表级完整性约束条件*/
CONSTRAINT CK_Sex CHECK(Sex = '男' OR Sex = '女'),          /*表级完整性约束条件*/
CONSTRAINT UQ_Sex UNIQUE(tel))
```

③创建图书表 Book。

```
USE TSGL
CREATE TABLE Book(
BookID CHAR(4),
ISBN VARCHAR(15) NOT NULL,
BookName VARCHAR (20) NOT NULL,
Author VARCHAR (8),
Publisher VARCHAR (20),
PublisherDate DATE,
Price NUMERIC(4, 1),
SNum INT NOT NULL DEFAULT 0,
CONSTRAINT PK_BookID PRIMARY KEY(BookID)
)
```

④创建借阅表 Lend。

```
USE TSGL
CREATE TABLE Lend(
LendID INT IDENTITY(1, 1) PRIMARY KEY,
ReaderID CHAR(6) NOT NULL,
BookID CHAR(4) NOT NULL,
LendDate DATE NOT NULL,
BackDate DATE,
CONSTRAINT FK_RID FOREIGN KEY(ReaderID) REFERENCES Reader(ReaderID),
```

```
CONSTRAINT FK_BID FOREIGN KEY(BookID) REFERENCES Book(BookID)
)
```

⑤向 Reader 表添加电子邮件列 Email，类型为 VARCHAR(20)，并添加取值唯一的约束，约束名为 UQ_Email。

```
USE TSGL
ALTER TABLE Reader
ADD Email VARCHAR (20) CONSTRAINT UQ_Email UNIQUE(Email)
```

⑥删除 Reader 表的唯一约束 UQ_Emai。

```
USE TSGL
ALTER TABLE Reader
DROP CONSTRAINT UQ_Email
```

⑦将 Reader 表 Email 列的数据类型修改为 CHAR(15)。

```
USE TSGL
ALTER TABLE Reader
ALTER COLUMN Email CHAR (15)
```

⑧删除 Reader 表的 Email 列。

```
USE TSGL
ALTER TABLE Reader
DROP COLUMN Email
```

⑨删除借阅表 Lend。

```
USE TSGL
DROP TABLE Lend
```


五、扩展练习

商品销售数据库 SPXS 中有三张表，其结构如表 1-3-5 至表 1-3-7 所示。

表 1-3-5　商店表 Shop 的结构

字段名	字段含义	数据类型	允许为空	备注
ShopNo	商店编号	CHAR(3)	N	主键
ShopName	商店名称	CHAR (10)	N	
ShopAddr	商店地址	CHAR (20)	Y	

表 1-3-6　商品表 Product 的结构

字段名	字段含义	数据类型	允许为空	备注
ProNo	商品编号	CHAR(3)	N	主键
ProName	商品名称	VARCHAR(10)	Y	
ProPrice	商品价格	DECIMAL	Y	

表 1-3-7　销售表 Sale 的结构

字段名	字段含义	数据类型	允许为空	备注
ShopNo	商店编号	CHAR(3)	N	外键，参照 Shop 表的主键
ProNo	商品编号	CHAR(3)	N	外键，参照 Product 表的主键
Amount	销售量	INT	Y	默认值为 0
ShopNo＋ProNo 为主键				

用 T-SQL 语句完成以下操作：

①创建 Shop、Product 和 Sale 这三个表，并添加相应的约束。

②为 Product 表添加一个字段生产厂家 Factory，数据类型为 VARCHAR(10)。

③将 Product 表 Factory 字段的数据类型修改为 CHAR(15)。

④删除 Product 表的 Factory 字段。

⑤为 Shop 表的 ShopName 字段增加唯一约束，约束名为 UQ_ShopName。

⑥删除 Shop 表的约束 UQ_ShopName。

⑦删除销售表 Sale。

实验 4　单表查询

一、实验目的

- 掌握 SELECT 查询语句的基本语法；
- 熟练掌握用 SELECT 语句进行单表查询；
- 熟练掌握聚合函数的使用；
- 学会分组统计。

二、实验内容

- 使用 SELECT 子句进行简单查询；

- 带 WHERE 子句的查询；
- 带 GROUP BY、ORDER BY 子句的查询；
- 聚合函数的应用。

三、相关知识

1. SELECT 查询的语法格式

SELECT[ALL | DISTINCT] [TOP n[PERCENT]] ＜目标列表达式＞[，＜目标列表达式＞] …
[INTO＜新表名＞]
FROM＜表名或视图名＞[，＜表名或视图名＞] …
[WHERE＜条件表达式＞]
[GROUP BY＜列名 1＞[HAVING＜条件表达式＞]]
[ORDER BY＜列名 2＞[ASC | DESC]]；

含义是：根据 WHERE 子句的条件表达式，从 FROM 子句指定的基本表或视图中找出满足条件的元组，再按 SELECT 子句中的目标列表达式，筛选出元组中的属性值形成结果表。

如果带有 GROUP BY 子句，则将结果按＜列名 1＞的值进行分组，该属性列值相等的元组为一个组，通常会在每个组中作用聚集函数。如果 GROUP BY 子句带 HAVING 短语，则只有满足指定条件的组才予以输出。

如果带有 ORDER BY 子句，则结果表还要按＜列名 2＞的值进行升序或降序排序。

四、实验指导

1. SELECT 子句的应用

①查询读者表 Reader 的全部信息。

```
SELECT * FROM Reader
```

②查询所有读者的编号、姓名，并将输出结果中的列名显示为中文的“读者编号”“姓名”。

```
SELECT ReaderID 读者编号, ReaderName 姓名
FROM Reader
```

③查询前 20％的读者的姓名、所在系部。

```
SELECT TOP 20 PERCENT ReaderName, Dept
```

```
FROM Reader
```

④运行以下两组 SQL 语句，观察结果有何区别。

```
SELECT ReaderID FROM Lend
SELECT DISTINCT ReaderID FROM Lend
```

使用 DISTINCT 语句可以从查询结果中消除重复的行。如果在对某列进行查询时，该列中包含 NULL 值，则 NULL 值也会出现在结果集中。

2. WHERE 子句的应用

①查询女读者的姓名和所在系部。

```
SELECT ReaderName, Dept
FROM Reader
WHERE Sex = '女'
```

②查询计科系、英语系的读者的全部信息。

```
SELECT  *
FROM Reader
WHERE Dept IN('计科系', '英语系')
或者
SELECT *
FROM Reader
WHERE Dept = '计科系' OR Dept = '英语系'
```

③查询姓张的读者的姓名、性别和所在系部。

```
SELECT ReaderName, Sex, Dept
FROM Reader
WHERE ReaderName LIKE '张%'
```

④查询还书日期为空的借阅记录。

```
SELECT *
FROM Lend
WHERE BackDate IS NULL
```

⑤查询电话号码尾数为 0～5 的读者的全部信息。

```
SELECT * FROM Reader
WHERE Tel LIKE '%[0-5]'
```

⑥查询图书编号为“B005”的图书出版的年和月，并用 AS 别名。

```
SELECT YEAR(PublisherDate) AS 年，MONTH(PublisherDate) AS 月
FROM Book
WHERE BookID = 'B005'
```

其中，函数 YEAR（　）返回指定日期的年份，MONTH（　）返回指定日期的月份，DAY（　）返回指定日期的天。

3. ORDER BY 子句的应用

①查询“计科系”读者的姓名和电话，并按姓名升序排列。

```
SELECT ReaderName，Tel
FROM Reader
WHERE Dept = '计科系'
ORDER BY ReaderName
```

②查询所有读者的全部信息，查询结果按所在系部升序排列，同一系中的学生按读者编号降序排列。

```
SELECT * FROM Reader
ORDER BY Dept，ReaderID DESC
```

排序输出的默认顺序是升序(ASC)，如果要按列值降序输出，需在列名后加上 DESC。ORDER BY 子句是对查询结果的排序，因此一般出现在查询语句的最后。

4. 聚合函数和 GROUP BY 子句的应用

①查询读者的总数量。

```
SELECT COUNT(*) AS 总人数
FROM Reader
```

②查询读者表中女读者和男读者的数量。

```
SELECT Sex，COUNT(*) AS 人数
FROM Reader
GROUP BY Sex
```

GROUP BY 分组的目的：细化聚集函数的作用对象。

①若未对查询结果分组，聚集函数将作用于整个查询结果。

②若对查询结果分组后，聚集函数将分别作用于每个组，即每个组都有一个函数值。

使用 GROUP BY 子句后，SELECT 子句的后面若要显示列名，则只能出现分组属性(即 GROUP BY 之后的属性)和聚合函数，不能出现除此之外的任何属性，否则会出现错

误，图 1-4-1 是错误的用法。

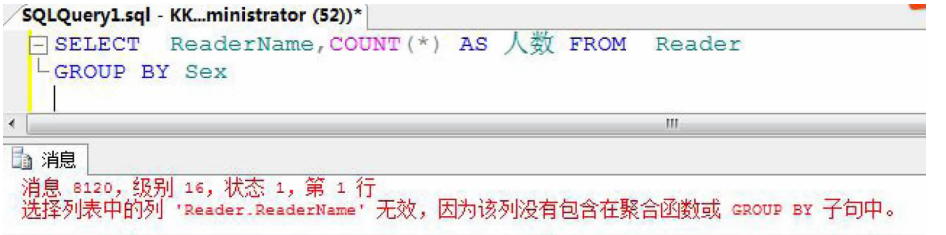

图 1-4-1　错误示例

③查询图书表中图书的最高和最低价格。

```
SELECT  MAX(Price)  AS  '最高价格', MIN(Price)  AS  '最低价格'
FROM  BOOK
```

④查询“电子工业出版社”出版的图书的平均价格。

```
SELECT  AVG(Price)  平均价格
FROM  BOOK
WHERE  Publisher = '电子工业出版社'
```

⑤查询借阅了两本及两本以上图书的读者编号。

```
SELECT  ReaderID  FROM  Lend
GROUP  BY  ReaderID
HAVING  COUNT( * )>= 2
```

如果分组后还要求按一定的条件对这些组进行筛选，则可在 GROUP BY 子句后使用 HAVING 子句指定筛选条件。

HAVING 子句与 WHERE 子句的区别是作用对象不同，即

①WHERE 子句作用于基表或视图，从中选择满足条件的元组。

②HAVING 子句作用于 GROUP BY 分组查询后产生的组，从中选择满足条件的组。

③HAVING 中的条件使用聚合函数，WHERE 中的条件不能使用聚合函数。

5. COMPUTE 和 INTO 子句的应用

从上例发现，在 SELECT 子句中出现聚合函数时，结果集中的数据全是聚合值，没有明细值。使用 COMPUTE 子句可以解决此问题。

在 SELECT 语句中使用 COMPUTE 子句时，查询结果由两个部分组成：前一部分就是未用 COMPUTE 子句时产生的结果集；后一部分只有一行，是由 COMPUTE 子句产生附加的汇总数据，出现在整个结果集的末尾。语法格式如下：

```
COMPUTE  聚合函数名(表达式)
```

[BY　　列名]

①查询图书表中所有图书的详细信息，并计算图书的平均价格。

```
SELECT * FROM BOOK
COMPUTE AVG(Price)
```

结果如图 1-4-2 所示。

结果　消息

	BookID	ISBN	BookName	Author	Publisher	PublisherDate	Price	SNum
1	B001	9787302328438	计算机组成原理	蒋本珊	电子工业出版社	2015-06-01	35.3	8
2	B002	9787302309338	通信电子线路	侯丽敏	清华大学出版社	2013-11-02	28.0	4
3	B003	9787302266297	大学计算机基础	高敬阳	电子工业出版社	2013-11-05	27.6	3
4	B004	9787302369646	C语言程序设计	谭浩强	机械工业出版社	2014-01-05	48.3	5
5	B005	9787302188940	数据结构与算法	唐宁九	清华大学出版社	2016-01-05	49.0	4
6	B006	9787302270683	计算机网络安全	姚永雷	清华大学出版社	2014-07-05	25.0	0

	avg
1	35.533333

图 1-4-2　COMPUTE 子句的运行结果

COMPUTE 不带 BY 时，生成合计作为附加的汇总列出现在结果集的最后。

②查询图书表中所有图书的详细信息，并计算每个出版社出版的图书的平均价格。

```
SELECT * FROM BOOK
ORDER BY Publisher
COMPUTE AVG(Price) BY Publisher
```

结果如图 1-4-3 所示。

结果　消息

	BookID	ISBN	BookName	Author	Publisher	PublisherDate	Price	SNum
1	B003	9787302266297	大学计算机基础	高敬阳	电子工业出版社	2013-11-05	27.6	3
2	B001	9787302328438	计算机组成原理	蒋本珊	电子工业出版社	2015-06-01	35.3	8

	avg
1	31.450000

	BookID	ISBN	BookName	Author	Publisher	PublisherDate	Price	SNum
1	B004	9787302369646	C语言程序设计	谭浩强	机械工业出版社	2014-01-05	48.3	5

	avg
1	48.300000

	BookID	ISBN	BookName	Author	Publisher	PublisherDate	Price	SNum
1	B005	9787302188940	数据结构与算法	唐宁九	清华大学出版社	2016-01-05	49.0	4
2	B006	9787302270683	计算机网络安全	姚永雷	清华大学出版社	2014-07-05	25.0	0
3	B002	9787302309338	通信电子线路	侯丽敏	清华大学出版社	2013-11-02	28.0	4

	avg
1	34.000000

图 1-4-3　COMPUTE BY 子句的运行结果

COMPUTE 与 BY 一起使用时，在结果集内生成控制中断和分类汇总。

如果使用了 COMPUTE BY 子句，则必须使用 ORDER BY 子句，而且 COMPUTE BY 子句中的列必须包含在 ORDER BY 子句中。

③从读者表中查询“数学系”读者的姓名和性别，并将结果存放在新表“MA_table”中。

```
SELECT  ReaderName, Sex
INTO  MA_table
FROM  Reader
WHERE  Dept = '数学系'
```

五、扩展练习

①查询价格在 20～30 元的图书的全部信息。

②查询清华大学出版社、机械工业出版社出版的图书名称和作者，并将输出结果中的列名显示为中文的“图书名称”“作者”。

③查询图书名称中包含“计算机”的图书信息。

④查询“清华大学出版社”不姓“唐”的作者编写的图书名称。

⑤查询各系部读者的人数。

⑥查询图书表中“清华大学出版社”出版的图书的总库存量。

⑦查询各出版社中图书的平均价格大于 32 元的出版社名称，并按出版社名称降序排列。

实验 5　连接查询、子查询

一、实验目的

- 掌握多表连接查询的方法，包括内连接、外连接等；
- 熟练掌握子查询的方法，包括相关子查询与不相关子查询。

二、实验内容

- 内、外连接查询；
- IN 子查询、比较子查询、EXISTS 或 NOT EXISTS 子查询。

三、相关知识

1. 连接查询

连接查询是涉及多个表的查询，分为内连接、外连接、自身连接等。

(1)内连接

内连接操作只输出满足连接条件的元组，内连接的语法格式如下：

```
SELECT  目标列表达式[，＜目标列表达式＞] …
FROM  表 1[INNER]JOIN  表 2  ON＜连接条件＞
WHERE＜条件表达式＞
```

或

```
SELECT  目标列表达式[，＜目标列表达式＞]…
FROM  表 1，表 2
WHERE＜连接条件＞[AND 条件表达式＞]
```

(2)外连接

外连接的查询结果中，不仅包括那些满足条件的行，而且某些表不满足条件的行也会显示在结果集中。外连接分为三种。

①左外连接：列出连接条件中左边表的所有元组。

②右外连接：列出连接条件中右边表的所有元组。

③全外连接：对左右表都不进行限制，两个表的所有元组都会包括在结果集中。

外连接的语法格式如下：

```
SELECT 目标列表达式[，＜目标列表达式＞]…
FROM  表 1  LEFT | RIGHT | FULL[OUTER]JOIN  表 2
ON  表 1. 列 1 = 表 2. 列 2
```

(3)自身连接

连接可以在同一张表内进行自身连接，即将同一个表的不同行连接起来。自身连接可以看作一张表的两个副本之间的连接，必须为表指定两个别名，使之在逻辑上成为两张表。

自身连接的语法格式如下：

```
SELECT  目标列表达式[，＜目标列表达式＞]…
FROM  表 1  别名 1，表 1  别名 2 [，…]
WHERE＜连接条件＞[AND 条件表达式＞]
```

2. 子查询

子查询是指在一个 SELECT 查询语句的 WHERE 子句中包含另一个 SELECT 查询语句。外层 SELECT 查询语句称为父查询，WHERE 子句中的 SELECT 查询语句称为子查询。

(1)不相关子查询(IN 子查询)

子查询的查询条件不依赖于父查询。不相关子查询求解方法：由里向外逐层处理。即每个子查询在上一级查询处理之前求解，子查询的结果用于建立其父查询的查找条件。

(2)比较子查询

比较子查询是指将父查询的某个字段或表达式与子查询的结果进行比较运算，格式为：

＜表达式＞＜比较运算符＞[ALL | SOME | ANY]＜子查询＞

(3)相关子查询

子查询的查询条件依赖于父查询。相关子查询的求解方法有以下几种。

①首先取外层查询中表的第一个元组，外部查询将子查询引用的列的值传给子查询。

②若子查询的任何行与其匹配，外部查询就返回结果行。

③然后再取外层表的下一个元组。

④重复此过程，直至外层表的所有行全部检查完为止。

常见的相关子查询有：EXISTS 子查询和 NOT EXISTS 子查询。

四、实验指导

1. 内连接

①查询借阅了蒋本珊编写的《计算机组成原理》这本书的读者编号。

方法一：

```
SELECT  ReaderID
FROM  Lend, Book
WHERE  Book.BookID = Lend.BookID
        AND  Author = '蒋本珊'
        AND  BookName = '计算机组成原理'
```

方法二：

```
SELECT  ReaderID
FROM  Lend  a  INNER  JOIN  Book  b
ON  a.BookID = b.BookID
```

```
    AND   Author = '蒋本珊'
    AND   BookName = '计算机组成原理'
```

②查询读者“刘明”已借阅的图书编号和书名。

```
SELECT   Book. BookID, BookName
FROM   Reader, Book, Lend
WHERE   Reader. ReaderID = Lend. ReaderID
          AND   Book. BookID = Lend. BookID
          AND   ReaderName = '刘明'
```

注意：第一行 BookID 前的表名 Book 不能省略，也可以写成 Lend，因为 BookID 是 Book 和 Lend 表的公有属性，因此必须加上前缀，否则无法区分此属性。

2. 外连接

①查询每个读者及其借阅情况，并比较以下两条命令的执行结果。

命令一：

```
SELECT a. *, b. *
FROM Reader a LEFT OUTER JOIN Lend b
ON a. ReaderID = b. ReaderID
```

命令二：

```
SELECT   a. *, b. *
FROM   Reader   a,   Lend   b
WHERE   a. ReaderID = b. ReaderID
```

命令一是左外连接，会显示读者表中所有的读者信息，即使该读者没有借过书籍；命令二是内连接，只显示借阅过书籍的读者及其借阅信息，而不会显示没有借过书籍的读者信息。

②查询被借阅过的图书信息。

```
SELECT DISTINCT Book. *
FROM Book RIGHT JOIN Lend
ON Book. BookID = Lend. BookID
```

观察查询结果，发现仅仅显示出借阅过的图书信息。

3. 自身连接

查询同时借阅了 B001 和 B004 图书的读者编号。

```
SELECT   a. ReaderID
```

```
FROM  Lend  a, Lend  b
WHERE  a.ReaderID = b.ReaderID
AND  a.BookID = 'B001'
AND  b.BookID = 'B004'
```

4. IN 子查询

①查询“赵青青”读者借阅的图书名称(用多重子查询实现)。

```
SELECT  BookName
FROM  Book
WHERE  BookID  IN
        (SELECT  BookID
        FROM  Lend
        WHERE  ReaderID IN
            (SELECT  ReaderID
            FROM  Reader
            WHERE  ReaderName = '赵青青'
            )
        )
```

②查询借阅过图书的读者姓名及部门(用子查询实现)。

```
SELECT  ReaderName, Dept
FROM  Reader
WHERE  ReaderID  IN
(SELECT  ReaderID
          FROM    Lend)
```

5. 比较子查询

①查询比“电子工业出版社”的所有图书价格都低的图书信息，并按价格降序排列。

方法一：用谓词比较。

```
SELECT  *
FROM  Book
WHERE  Price<ALL
        (SELECT  Price
        FROM  Book
        WHERE  Publisher = '电子工业出版社')
```

```
    AND  Publisher! ='电子工业出版社'
ORDER  BY  Price DESC
```

方法二：用聚合函数比较。

```
SELECT  *
FROM  Book
WHERE  Price<
            (SELECT  MIN(Price)
            FROM    Book
            WHERE  Publisher = '电子工业出版社')
        AND  Publisher! ='电子工业出版社'
ORDER  BY  Price  DESC
```

6. EXISTS 子查询、NOT EXISTS 子查询

①查询借阅了编号为“B001”的图书的读者姓名。

方法一：EXISTS 子查询。

```
SELECT  ReaderName
FROM  Reader
WHERE  EXISTS
              (SELECT  *
              FROM  Lend
              WHERE  ReaderID = Reader. ReaderID
              AND  BookID = 'B001' )
```

方法二：内连接查询。

```
SELECT  ReaderName
FROM  Reader, Lend
WHERE  Reader. ReaderID = Lend. ReaderID
        AND  BookID = 'B001'
```

②查询从未被借阅过的图书名称。

方法一：NOT EXISTS 子查询。

```
SELECT  BookName
FROM  Book
WHERE  NOT  EXISTS
        (SELECT  *
```

```
        FROM  Lend
        WHERE  BookID = Book.BookID)
```

方法二：IN 子查询。

```
SELECT  BookName
FROM  Book
        WHERE  BookID  NOT  IN
        (SELECT  BookID
        FROM  Lend)
```

EXISTS 或 NOT EXISTS 子查询，只返回真值或假值，不返回实际数据，因此目标列表达式通常都用 *，给出列名无实际意义。

五、扩展练习

①利用外连接，查询所有的图书信息及其被借阅的情况(没有被借阅过的图书，也要显示其信息)。

②查询姓赵的读者借阅的图书名称，分别用连接查询和子查询实现。

③用子查询实现：查询借阅次数大于等于 2 次的图书的 ISBN 和图书名称。

④查询借阅了“清华大学出版社”出版的图书的读者姓名，分别用连接查询和子查询实现。

⑤用比较子查询实现：查询图书表中库存比“清华大学出版社”出版的所有图书库存量都大的图书名称和作者名，分别用谓词比较和聚合函数比较实现。

实验6　表数据的插入、修改和删除

一、实验目的

- 熟练掌握使用 SQL Server Management Studio 编辑表中的数据；
- 熟练掌握使用 T-SQL 语句对表数据进行插入、修改和删除操作。

二、实验内容

- 使用 SQL Server Management Studio 插入、修改和删除数据；
- 使用 T-SQL 语句插入、修改和删除数据。

三、相关知识

1. 插入记录

INSERT INTO＜表名＞[(＜列名 1＞[，＜列名 2 ＞…)]

VALUES (＜常量 1＞ [，＜常量 2＞]…)

VALUES 子句中的各个值之间用逗号隔开，并且要与表名后面的括号中所指出的属性个数相同且排列顺序一致。当要插入的记录包含表中全部属性值时，表名后面的列名可以省略。

2. 修改记录

UPDATE＜表名＞

SET＜列名＞＝＜表达式＞[，＜列名＞＝＜表达式＞]…
[WHERE＜修改条件＞]

3. 删除记录

DELETE　[FROM]＜表名＞

[WHERE＜删除条件 ＞]

注意：UPDATE 和 DELET 命令中，WHERE 子句指定要删除或修改的元组，省略 WHERE 子句表示修改或删除表中的全部元组。

四、实验指导

首先导入备份好的数据库“TSGL_data. bak”。

1. 利用 SQL Server Management Studio 插入、修改和删除数据

①在“对象资源管理器”中依次展开“数据库”—“TSGL”—“表”，右击需要输入数据的表如“Reader”，在弹出的快捷菜单中选择“编辑前 200 行” 命令，如图 1-6-1 所示。

②将鼠标定位在最后一行，插入一条新记录(‘091104’，‘张三’，‘男’，1，0，‘英语系’，‘13112340000’)，编辑完后，单击下一空行，再单击工具栏中的“执行 SQL”按钮“!”，即可进行保存。

③如果要删除该记录，右击该记录第一列前的按钮，选择“删除”命令，在弹出的删除提示对话框中单击“是”按钮；如果要修改记录，将光标移到需要修改的字段直接修改即可。

注意：当输入外键属性的值时，输入的值需要在被参照表的主码中对应存在。

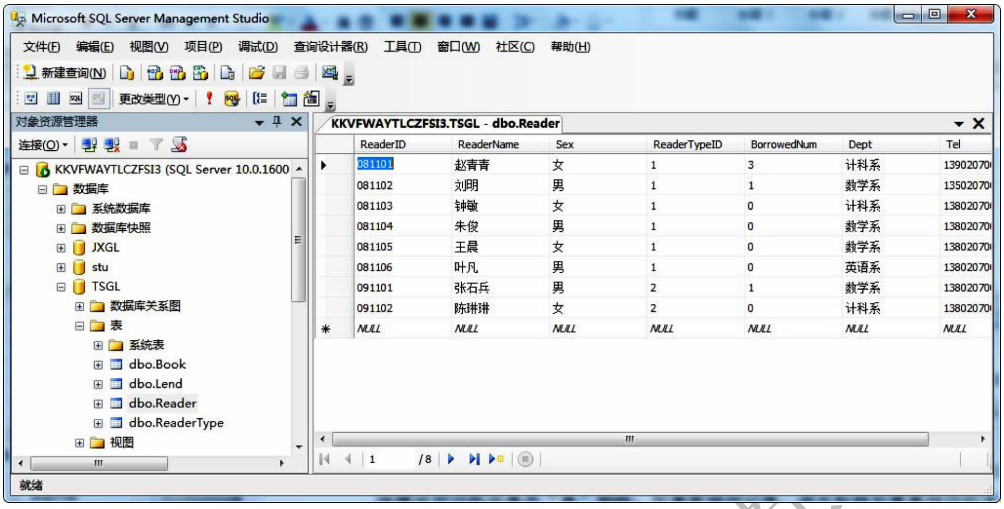

图 1-6-1　SQL Server Management Studio 编辑数据的界面

2. 利用 T-SQL 语句插入、修改和删除数据

①向 Lend 表中插入两条记录。

```
USE  TSGL
GO
INSERT  INTO  Lend(ReaderID, BookID, LendDate, BackDate)
VALUES('091101', 'B001', '2017-09-01', NULL), ('091102', 'B005', '2017-09-01', NULL)
```

注意：Lend 表中的 BookID 和 ReaderID 是外键，由于在 Book 表中存在 BookID 为'B001'和'B005'的图书，Reader 表中存在编号为'091101'的读者，故可以在 Lend 表插入这两条记录，若不存在，则不能插入。

②向 Reader 表插入一条记录："编号：091104""姓名：王丽""读者类型：2""电话：13678911234"。

```
INSERT  INTO  Reader(ReaderID, ReaderName, ReaderTypeID, Tel)
VALUES('091104', '王丽', 2, '13678911234')
```

③将 Reader 表中编号为"091104"的读者的系部修改为"数学系"，读者类型编号修改为 1。

```
UPDATE  Reader
SET  ReaderTypeID = 1, Dept = '数学系'
WHERE  ReaderID = '091104'
```

④将 ReaderType 表中所有读者类型的限借数量增加 2 本。

```
UPDATE  ReaderType  SET  LimitNum + = 2
```

⑤删除 Reader 表中编号为“091104”的读者信息。

```
DELETE  FROM  Reader
WHERE  ReaderID = '091104'
```

⑥从 Lend 表中删除所有借阅了图书“通信电子线路”的借阅记录。

```
DELETE  FROM  Lend
WHERE  BookID  IN
          (SELECT  BookID
          FROM  Book
          WHERE  BookName = '通信电子线路')
```

五、扩展练习

①向 Book 表中插入一条记录：“BookID：B007”“ISBN：9787302270123”“BookName：操作系统原理”“SNum：5”。

②将 ReaderType 表中所有读者类型的限借数量减少 2 本。

③将 ReaderType 表中“本科生”类别读者的限借数量修改为 4 本，借阅期限修改为 2 个月。

④从 Lend 表删除读者'091101'借阅图书'B001'的借阅记录。

⑤从 Lend 表中删除所有“计科系”读者的借阅记录。

⑥删除 Lend 表中的所有借阅记录。

实验 7　视图与索引

一、实验目的

• 熟练掌握利用 SQL Server Management Studio 和 T-SQL 语句创建、修改、使用和删除视图的方法；

• 熟练掌握利用 SQL Server Management Studio 和 T-SQL 语句创建和删除索引的方法；

• 会根据实际情况设计视图来解决实际问题。