

项目一

创建和维护数据库

项目要点

- 创建数据库表
- 修改和删除数据库表
- 创建或删除主键和外键

引言

数据库表是存储数据的基本单元，在创建数据库表之前，必须做出有关表结构的规划，包括表要包含的内容，反映到表结构上是指表将要包含哪些字段以及这些字段的数据类型。本项目通过 3 个工作任务，形象地向读者展示数据库的基本操作。

任务一：创建数据库表

任务描述

小王是某学校学生处教师，为了加强对学生的管理，小王需要统计本校在校学生人数及学生的相关信息。

任务分析

在 SQL Server 中，用户可以通过创建数据库表来进行数据的统计与整理，并有不同的数据库类型可供选择。因此，小王决定通过建立数据库表来进行统计。

准备知识

数据库 (Database) 是按照数据结构来组织、存储和管理数据的仓库，它产生于距今六十年前，随着信息技术和市场的发展，特别是 20 世纪 90 年代以后，数据管理不再仅仅是存储和管理数据，而转变成用户所需要的各种数据管理的方式。数据库有很多种类型，从最简单的存储有各种数据的表格到能够进行海量数据存储的大型数据库系统都在各个方面得到了广泛的应用。

1. 数据库相关概念

(1) 数据

数据 (Data) 实际上就是描述事物的符号记录。

计算机中的数据一般分为两部分，其中一部分与程序仅有短时间的交互关系，随着程序的结束而消亡，它们称为临时性 (Transient) 数据，这类数据一般存放于计算机内存中；而另一部分数据则对系统起着长期持久的作用，它们称为持久性 (Persistent) 数据。数据库系统中处理的就是这种持久性数据。

软件中的数据是有一定结构的。首先，数据有型 (Type) 与值 (Value) 之分，数据的型给出了数据表示的类型，如整型、实型、字符型等，而数据的值给出了符合给定型的值，如整型值 1。随着应用需求的扩大，数据的型有了进一步的扩大，它包括了将多种相关数据以一定结构方式组合构成特定的数据框架，这样的数据框架称为数据结构 (Data Structure)，数据库中在特定条件下称之为数据模式 (Data Schema)。

过去的软件系统是以程序为主体，而数据则以私有形式从属于程序，此时数据在系统中是分散、凌乱的，这也造成了数据管理的混乱，如数据冗余度高，数据一致性差以及数据的安全性差等多种弊病。



知识链接

近 10 多年来, 数据在软件系统中的地位产生了变化, 在数据库系统及数据库应用系统中数据已占有主体地位, 而程序已退居附属地位。在数据库系统中需要对数据进行集中、统一的管理, 以达到数据被多个应用程序共享的目标。

(2) 数据库

数据库(Database, 简称 DB) 是数据的集合, 它具有统一的结构形式并存放于统一的存储介质内, 是多种应用数据的集成, 并可被各个应用程序共享。

数据库存放数据是按数据所提供的数据库模式存放的, 它能构造复杂的数据结构以建立数据间的内在联系与复杂的关系, 从而构成数据的全局结构模式。

数据库中的数据具有“集成”“共享”之特点, 也就是说数据库集中了各种应用的数据, 进行统一的构造与存储, 而使它们可被不同应用程序使用。

(3) 数据库管理系统

数据库管理系统(Database Management System, 简称 DBMS) 是管理数据库的一种系统软件, 负责数据库中的数据组织、数据操纵、数据维护、控制及保护和数据服务等。数据库中的数据是海量级的数据, 并且其结构复杂, 因此需要提供管理工具。数据库管理系统是数据库系统的核心, 它主要有如下几方面的具体功能:

① 数据模式定义。数据库管理系统负责为数据库构建模式, 也就是为数据库构建其数据框架。

② 数据存储的物理构建。数据库管理系统负责为数据模式的物理存取及构建提供有效的存取方法与手段。

③ 数据操纵。数据库管理系统为用户使用数据库中的数据提供方便, 它一般提供查询、插入、修改以及删除数据的功能。此外, 它自身还具有做简单算术运算及统计的功能, 而且还可以与某些过程性语言结合, 使其具有强大的过程性操作功能。

④ 数据的完整性、安全性定义与检查。数据库中的数据具有内在语义上的关联性与一致性, 它们构成了数据的完整性, 数据的完整性是保证数据库中数据正确的必要条件, 因此必须经常检查以维护数据的正确。



知识链接

数据库中的数据具有共享性, 而数据共享可能会引发数据的非法使用, 因此必须对数据正确使用做出必要的规定, 并在使用时做检查, 这就是数据的安全性。

数据完整性与安全性的维护是数据库管理系统的基本功能。

⑤ 数据库的并发控制与故障恢复。数据库是一个集成、共享的数据集合体, 它能

chapter
01chapter
02chapter
03chapter
04chapter
05

为多个应用程序服务，所以就存在着多个应用程序对数据库的并发操作。在并发操作中如果不加控制和管理，多个应用程序间就会相互干扰，从而对数据库中的数据造成破坏。因此，数据库管理系统必须对多个应用程序的并发操作做必要的控制以保证数据不受破坏，这就是数据库的并发控制。

数据库中的数据一旦遭受破坏，数据库管理系统必须有能力及及时进行恢复，这就是数据库的故障恢复。

⑥ 数据的服务。数据库管理系统提供对数据库中数据的多种服务功能，如数据拷贝、转存、重组、性能监测、分析等。

为完成以上六个功能，数据库管理系统一般提供相应的数据语言 (Data Language)，它们是：

数据定义语言 (Data Definition Language，简称 DDL)。该语言负责数据的模式定义与数据的物理存取构建。

数据操纵语言 (Data Manipulation Language，简称 DML)。该语言负责数据的操纵，包括查询及增、删、改等操作。

数据控制语言 (Data Control Language，简称 DCL)。该语言具有负责数据完整性、安全性的定义与检查以及并发控制、故障恢复等功能，包括系统初启程序、文件读写与维护程序、存取路径管理程序、缓冲区管理程序、安全性控制程序、完整性检查程序、并发控制程序、事务管理程序、运行日志管理程序、数据库恢复程序等。

上述数据语言按其使用方式具有两种结构形式：

交互式命令语言。它的语言简单，能在终端上即时操作，它又称为自含型或自主型语言。

宿主型语言。它一般可嵌入某些宿主语言 (Host Language) 中，如 C、C++ 和 COBOL 等高级过程性语言中。

此外，数据库管理系统还有为用户提供服务的服务性 (Utility) 程序，包括数据初始装入程序、数据转存程序、性能监测程序、数据库再组织程序、数据转换程序、通信程序等。

目前流行的 DBMS 均为关系数据库系统，比如 Oracle、Sybase 的 PowerBuilder 及 IBM 的 DB2、微软的 SQL Server 等，它们均为严格意义上的 DBMS 系统。



知识链接

另外有一些小型的数据库，如微软的 Visual Foxpro 和 Access 等，它们只具备数据库管理系统的一些简单功能。

(3) 数据库管理员

由于数据库的共享性,因此对数据库的规划、设计、维护、监视等需要有专人管理,他们被称为数据库管理员(Database Administrator,简称DBA)。其主要工作如下:

① 数据库设计(Database Design)。DBA的主要任务之一是做数据库设计,具体地说是进行数据模式的设计。由于数据库的集成与共享性,因此需要有专门人员(即DBA)对多个应用的数据需求作全面的规划、设计与集成。

② 数据库维护。DBA必须对数据库中的数据安全性、完整性、并发控制及系统恢复、数据定期转存等进行实施与维护。

③ 改善系统性能,提高系统效率。DBA必须随时监视数据库运行状态,不断调整内部结构,使系统保持最佳状态与最高效率。



知识链接

当效率下降时,DBA需采取适当的措施,如进行数据库的重组、重构等。

(4) 数据库系统

数据库系统(Database System,简称DBS)由如下几部分组成:数据库(数据)、数据库管理系统(软件)、数据库管理员(人员)、系统平台之一——硬件平台(硬件)、系统平台之二——软件平台(软件)。这五个部分构成了一个以数据库为核心的完整的运行实体,称为数据库系统。

在数据库系统中,硬件平台包括:

计算机:它是系统中硬件的基础平台,目前常用的有微型机、小型机、中型机、大型机及巨型机。

网络:过去数据库系统一般建立在单机上,但是近年来它较多地建立在网络上,从目前形势看,数据库系统今后将以建立在网络上为主,而其结构形式又以客户/服务器(C/S)方式与浏览器/服务器(B/S)方式为主。

在数据库系统中,软件平台包括:

操作系统:它是系统的基础软件平台,目前常用的有各种UNIX(包括LINUX)与WINDOWS两种。

数据库系统开发工具:为开发数据库应用程序所提供的工具,它包括过程性程序设计语言如C,C++等,也包括可视化开发工具VB、PB、Delphi等,它还包括近期与INTERNET有关的HTML及XML等以及一些专用开发工具。

接口软件:在网络环境下数据库系统中数据库与应用程序,数据库与网络间存在着多种接口,它们需要用接口软件进行联接,否则数据库系统整体就无法运作,这些接口软件包括ODBC,JDBC,OLEDB,CORBA,COM,DCOM等。

chapter
01chapter
02chapter
03chapter
04chapter
05

(5) 数据库应用系统

数据库应用系统 (Database Application System, 简称 DBAS) 是由数据库系统再加上应用软件及应用界面三者组成, 具体包括: 数据库、数据库管理系统、数据库管理员, 硬件平台、软件平台、应用软件、应用界面。其中应用软件是由数据库系统所提供的数据库管理系统 (软件) 及数据库系统开发工具组成, 而应用界面大多由相关的可视化工具开发而成。

数据库应用系统的 7 个部分以一定的逻辑层次结构方式组成一个有机的整体。如果不计数据库管理员 (人员) 并将应用软件与应用界面记成应用程序, 则数据库应用系统的结构如图 1-1 所示。

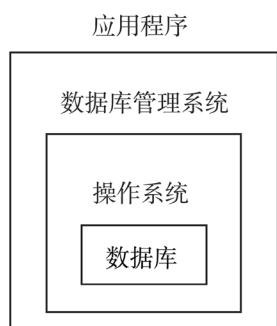


图 1-1 数据库系统的软硬件层次结构图

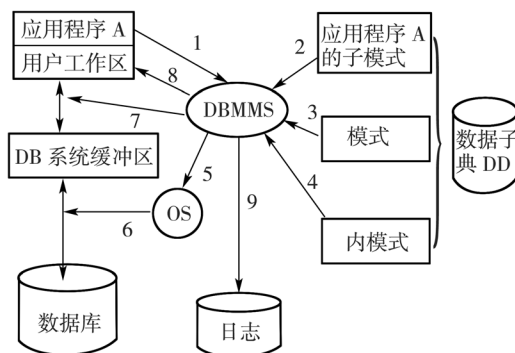


图 1-2 数据库访问数据的步骤

数据库访问数据的步骤如图 1-2 所示。对其各个步骤简单说明如下:

- ① 用户程序中有一条读数据库记录的 DML 语句, 当计算机执行到该语句时, 即向 DBMS 发出读取相应记录的命令。
- ② DBMS 接到该命令后, 首先访问该用户对应的子模式, 检查该操作是否在合法授权范围内及欲读记录的正确性、有效性, 若不合法则拒绝执行, 并向应用程序状态返回区发出回答状态信息; 反之执行下一步。
- ③ DBMS 读取模式描述并从子模式映像到全局模式, 从而确定所需的逻辑记录类型。
- ④ DBMS 从逻辑模式映像到存储模式, 从而确定读入哪些物理记录以及具体的地址信息。
- ⑤ DBMS 向操作系统发出从指定地址读取记录的命令。
- ⑥ 操作系统执行读命令, 按指定地址从数据库中把记录读入系统缓冲区, 并在操作结束后向 DBMS 做出回答。
- ⑦ DBMS 按照模式将读入系统缓冲区中的内容映像成用户要求读取的逻辑记录。
- ⑧ DBMS 将导出的逻辑记录送入用户工作区, 并将操作执行情况的状态信息返回

给用户。

⑨ DBMS 将已执行的操作载入运行日志。

⑩应用程序根据返回的状态信息决定是否利用该数据进行操作等。



知识链接

如果用户是更新一个记录内容，则执行过程类似。首先读出目标记录，并在用户工作区中进行修改，然后向 DBMS 发出“写回修改数据”的数据库指令即可。

本任务先介绍 SQL Server 中的数据类型、空值的含义和 Identity 列，最后介绍如何使用 SQL Server Management Studio 和使用 T-SQL 创建数据库表。

2. 数据类型

在创建数据库表时，必须为每一表列指定数据库类型。SQL Server 提供了许多内建系统数据库类型，这些类型大致可以分成以下几类：

数值类型。如 int, numeric 等。

字符类型。如 char, varchar 等。

文本和图像。如 text 和 image 等。

时间日期类型。如 datetime, smalldatetime 等。

其他特殊数据类型。如 table, sysname 和 uniqueidentifier 等。

首先看看数值类型。表 1-1 列出了所有的数值类型的数据类型以及它们的存储容量。

表 1-1 数值类型

数据类型	描述	存储大小
bit	值为 1、0 或 Null 的整型数据类型，bit 列不抱括索引	1 字节
tinyInt	介于 0 ~ 255 之间的整数数据	1 字节
smallInt	介于 -2^{15} (-32768) 与 $2^{15}-1$ (32767) 之间的整数数据	2 字节
int	介于 -2^{31} (-2147483648) 与 $2^{31}-1$ (-2147483647) 之间的整数数据	4 字节
bigInt	一个 8 字节的整数数据	8 字节
binary[n]	n 字节固定长度的二进制数据，n 是介于 1 ~ 8000 之间的一个值。当一列的数据输入差别很小时，请使用 binary	N+2 字节
varbinary[n]	n 字节可变长度的二进制数据，n 是介于 1 ~ 8000 之间的一个值。当一列的数据输入差别很大时，请使用 varbinary	输入数据的实际长度 +2 字节。实际长度可以是 0 字节

chapter
01

chapter
02

chapter
03

chapter
04

chapter
05

(续上表)

数据类型	描述	存储大小
varbinary(max)	这是 SQL Server 新增的类型, max 最大可以到达 $2^{31}-1$ 字节, 即 2G。当长度超过 8000 字节时, 请使用 varbinary(max)	输入数据的实际长度 +2 字节。实际长度可以是 0 字节
decimal[p,s]	固定精度和刻度的数字。精度 (p) 指被保存数字的总位数, 刻度 (s) 指定数字的小数点后面的位数	根据精度不同情况, 其大小可以在 5 ~ 17 字节之间
numeric[p,s]	numeric 是 Decimal 的同义词。在功能上等价于 decimal 数据类型	根据精度不同情况, 其大小可以在 5 ~ 17 字节之间
real	范围在 $-3.40E+38 \sim 3.40E+38$ 之间的浮点精度数字数据	4 字节
float[(n)]	$-1.79E + 308 \sim -2.23E - 308$, 0, $2.23E - 308 \sim 1.79E + 308$	取决于 n 的值
money	货币数据。介于 $-2^{63}(-9\ 223\ 372\ 036\ 854\ 775\ 808) \sim 2^{63}-1(9\ 223\ 372\ 036\ 854\ 775\ 807)$ 之间, 精确到货币单位的万分之一	8 字节
smallmoney	货币数据。介于 -2 147 483 648 到 2 147 483 647	4 字节

备注: 所有标有 [n]、[(n)] 或 [p,s] 表示要指定的字段数据长度, 如果未在数据定义或变量声明语句中指定 n, 则默认长度为 1。如果在使用 CAST 和 CONVERT 函数时未指定 n, 则默认长度为 30。下面列出的数据类型也一样。

表 1-2 列出了 SQL Server 字符类型及其存储容量。

表 1-2 字符类型

数据类型	描述	存储大小
char[(n)]	固定长度, 非 Unicode 字符数据, 长度为 n 个字节。n 的取值范围为 1 至 8 000	n 个字节
varchar[(n)]	可变长度, 非 Unicode 字符数据。n 的取值范围为 1 至 8 000	输入数据的实际长度加 2 个字节
varchar[(max)]	可变长度, 非 Unicode 字符数据, max 指示最大存储大小是 $2^{31}-1$ 个字节	输入数据的实际长度加 2 个字节
nchar[(n)]	n 个字符的固定长度的 Unicode 字符数据。n 值必须在 1 到 4 000 之间	$2 \times n$ 字节 + 2 字节
nvarchar[(n)]	可变长度 Unicode 字符数据。n 值在 1 到 4 000 之间	$2 \times n$ 字节 + 2 字节
nvarchar[(max)]	可变长度 Unicode 字符数据, max 指示最大存储大小为 $2^{31}-1$ 字节	$2 \times n$ 字节 + 2 字节

表 1-3 列出了文本和图像 (image) 类型及其存储容量。

表 1-3 文本和图像类型

数据类型	描述	存储大小
text	服务器代码页中长度可变的非 Unicode 数据，最大长度为 $2^{31}-1$ (2 147 483 647) 个字符。当服务器代码页使用双字节字符时，存储仍是 2 147 483 647 字节。根据字符串，存储大小可能小于 2 147 483 647 字节	$2^{31}-1$ 字节
ntext	长度可变的 Unicode 数据，最大长度为 $2^{30}-1$ (1 073 741 823) 个字符。输入字符个数的两倍（以字节为单位）	$2^{30}-1$ 字节
image	长度可变的二进制数据，从 0 到 $2^{31}-1$ (2 147 483 647) 个字节	$2^{31}-1$ 字节

备注：在开发新程序时，尽量避免使用表 1-3 中列出的数据类型，这些类型在微软 SQL Server 未来的版本中将被删除。改用 Varbinary(max), Varchar(max), Nvarchar(max) 数据类型。

表 1-4 列出了日期时间类型。

表 1-4 日期和时间类型

数据类型	描述	存储大小
smalldatetime	1900 年 1 月 1 日到 2079 年 6 月 6 日，精度为 1 分钟	两个 2 字节的整数
datetime	日期和时间数据，从 1753 年 1 月 1 日到 9999 年 12 月 31 日，准确度为三百分之一秒或 3.33 毫秒。值被圆整到 .000、.003 或 .007 毫秒增量	两个 4 字节
timestamp	公开数据库中自动生成的唯一一个二进制数字的数据类型。Timestamp 通常用作给表行加版本戳的机制。一个表只能有一个 Timestamp 类型字段	8 个字节

表 1-5 列出了其他特殊数据类型。

表 1-5 其他特殊数据类型

数据类型	描述	存储大小
sysname	由特殊系统提供的、SQL Server 用户定义的数据类型。SQL Server 将 sysname 类型定义为 nvarchar(128)，可以存储 128 个 Unicode 字符（或 256 字节）	256 字节
xml	这是新增的数据类型，用来存储 xml 文档和片段	xml 数据类型实例的存储表示形式不能超过 2 GB
uniqueidentifier	全局唯一标识符 (GUID)。可以通过 NEWID() 函数产生	16 字节
sql_variable	用于存储 SQL Server 2008 支持的各种数据类型（不包括 text、ntext、image、timestamp 和 sql_variant）的值	最大长度可以是 8016 个字节
table	类似于使用临时表，此类型包括列表和数据类型，可以用来定义本地变量或用于用户定义的函数的返回值	随着表定义的变化而变化

chapter
01chapter
02chapter
03chapter
04chapter
05

3. 空值

Null 值是一个未知值，将该值引用为 Null。列值的为空性是指该列接受或拒绝 Null 值的能力。一列中的 Null 值通常表明对于一个特殊的数据行，该列中没有输入项，因为该值既不为空也不为 0；其真实的值是未知的。因此，没有相等的两个值。



拓展提高

如果用户需要的信息仍然不可用，则可能需要为空列，例如，客户信息的初始化，有些信息在初始化时可能未知，需要以后补充，这些需要以后补充的信息字段可以设为 Null 值。

一般来说，都该避免使用 Null 值。因为对这些值的更新或查询，以及对有这些值的列的一些选项的设置都会更加复杂，例如主键和 Identity 属性都不能在为空性列中使用。

如果定义了允许 Null 值的列，则可以通过以下方法将 Null 值输入到这列中：

如果将行插入到该表，但没有为这个空列指定数据值，这样 SQL Server 将为该列指派为 Null 值。

用户可以键入 Null 这个词，不加引号是为了和加了引号的字符串“Null”区别开。

4. Identity 列

在创建表时，可以通过向列定义中添加 Identity 属性来将某一列指定为标识列。创建带有 Identity 属性的列时，SQL Server 将根据种子值和增量值自动为该列生成一个行值，默认情况下不允许手工为 Identity 列插入行值，除非将标识列的 IDENTITY_INSERT 属性设置为 ON。种子 (Seed) 值是插入到表中的第一行的标识值。增量值 (Increment) 是 SQL Server 为了连续插入而递增标识值的数量。当每次插入行时，SQL Server 都将为该行的标识列指派一个当前的标识值。被插入的下一行收到这个标识值，它是一个比当前最大标识值更大的增量。这样，每个被插入的行都将收到一个唯一的标识符。为了唯一地标识某一行，通常都将标识列用作表中的主键约束。

例如，如果为标识列指定了 Identity(1,1)，其中第一个参数是种子值，第二个参数是增量值，那么被插入的第一行将得到值为 1 的标识列，第 2 行得到的值将是 2，而第三行得到的值是 3，依次类推。



任务实施

可以使用 SQL Server Management Studio 工具创建数据库表，也可以通过编码的方式使用 T-SQL 创建数据库表。具体操作步骤如下。

STEP 1 打开 SQL Server Management Studio 并连接到数据库服务器，在“对象资源管

理器”中展开“数据库”，接着展开“数据库”/“DataCollationSys”节点，在下拉菜单中选择“表”，单击鼠标右键，如图 1-3 所示。

STEP 2 在弹出式菜单中执行“新建表(N)…”命令，将出现图 1-4 所示的界面。

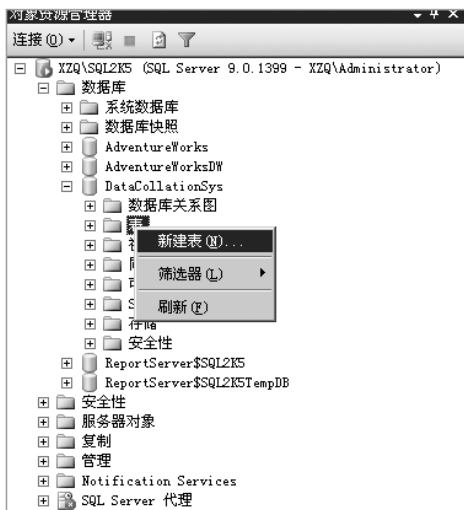


图 1-3 创建表



图 1-4 添加表列

STEP 3 在图 1-4 的界面中添加构成表的字段，为每个字段指定数据类型并设置是否允许为空。在下半部分的“列属性”中可以为字段设置更多的属性，例如名称、长度、数据类型及标识规范等。添加完表所需的所有字段后，单击“保存”按钮，将弹出图 1-5 所示的保存表界面，在“输入表名称”编辑框中输入表的名称后单击“确定”按钮即可。

STEP 4 在创建表的时候，可以为表指定主键。方法非常简单，在图 1-4 的界面中，选择要设置为主键的字段，然后单击工具条上的“设置主键”按钮，如图 1-6 所示。



图 1-5 保存表



图 1-6 设置主键

STEP 5 在图 1-4 的界面中，在“列名”列中输入 AreaID，“数据类型”列中选择 int 数据类型，去掉“允许空”列中“√”，在“列属性”中，展开“标识规范”，设置“(是标识)”属性值为“是”，“标识增量”属性为“1”（缺省值为 1），标识种子属性为“1”（缺省值为 1）。同样地，增加列 Area，分别指定列名“Area”，数据类型 varchar(20)，允许空。将字段 AreaID 设置为主键，完成后保存表名为 Area_Tbl。表 Area_Tbl 的结构如图 1-7 所示。

chapter
01

chapter
02

chapter
03

chapter
04

chapter
05



图 1-7 表 Area_Tbl

STEP 6 可以采用编码的方式，使用 T-SQL 来创建表。创建表的语法为：

```
CREATE TABLE
    [ database_name.[ owner ] .| owner.] table_name
    ( { < column_definition >
      | column_name AS computed_column_expression
      | < table_constraint > ::= [ CONSTRAINT constraint_
name ] }
      | [ { PRIMARY KEY | UNIQUE } [ ,...n ]
    )
    [ ON { filegroup | DEFAULT } ]
    < column_definition > ::= { column_name data_type }
    [ COLLATE < collation_name > ]
    [ [ DEFAULT constant_expression ]
      | [ IDENTITY [ ( seed , increment ) [ NOT FOR
REPLICATION ] ] ]
    ]
    [ ROWGUIDCOL ]
    [ < column_constraint > ] [ ...n ]
```

其中相关参数说明如下：

CREATE TABLE：主关键字；**table_name**：表示要创建的表的名称。

column_definition：表示列的定义，包括名称、数据类型、长度、是否为空等。

以创建表 Area_Tbl 为例，看看用 T-SQL 创建表的编码。创建表 Area_Tbl 的编码如下：

```
USE [DataCollationSys]
GO
/***** 对象： Table [dbo].[Area_Tbl]      脚本日期： 07/24/2006
19:01:02 *****/
CREATE TABLE [dbo].[Area_Tbl] (
    [AreaID] [int] IDENTITY(1,1) NOT NULL,
    [Area] [varchar](20) COLLATE Chinese_PRC_CI_AS NULL,
    CONSTRAINT [PK_AREA_TBL] PRIMARY KEY CLUSTERED
(
    [AreaID] ASC
)WITH (IGNORE_DUP_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO
```

其中相关参数说明如下：

USE [DataCollationSys]：指定了所要在其中创建表的库。

[dbo]：指定表的拥有者；[Area_Tbl] 是表名。AreaID 是自增列字段。

Area：该字段的字符集是 Chinese_PRC_CI_AS。

WITH (IGNORE_DUP_KEY = OFF)：说明表主键不会忽略重复值。

chapter
01chapter
02chapter
03chapter
04chapter
05

任务二：修改和删除数据库表

任务描述

小李是某酒店的大堂经理，为了能够统一规划酒店所承办的各种宴会以及宴会的类型、时间、准备工作等事项，小李需要确认酒店的数据库表。

任务分析

因为在创建了数据库表以后，有可能根据业务系统和存储数据内容的变化，对表做相应的修改。因此，小李决定根据实际情况对数据库表做相应的修改或删除。

准备知识

修改数据库表涉及的内容包括修改表名、字段名、增加或删除字段以及修改表字

段的属性等。

任务实施

1. 修改表名

首先看看通过 SQL Server Management Studio 来修改表名。打开 SQL Server Management Studio 并连接到数据库服务器,在“对象资源管理器”中展开“数据库”节点,接着展开数据库“DataCollationSys”节点,展开“表”文件夹,右键单击要更改名字的表,单击弹出菜单中的“重命名”菜单,如图 1-8 所示。然后输入新的表名,这样就完成了表名的修改。

再来看看使用 T-SQL 如何更改表名。可以使用系统存储过程 `sp_rename` 修改表名。`sp_rename` 语法:

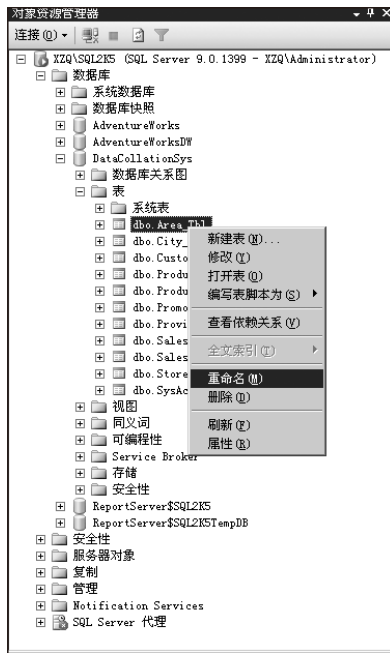


图 1-8 修改表名

```
sp_rename [ @objname = ] 'object_name' ,
          [ @newname = ] 'new_name'
          [ , [ @objtype = ] 'object_type' ]
```

其中相关参数说明如下:

@objname: 要修改的对象的名字; @newname: 新的数据库对象名

@objtype: 要修改的数据库对象的类型,这里是“object”。

以修改 Area_Tbl 表名为 Tbl_Area 为例,说明 `sp_rename` 的用法。

```
USE DataCollationSys
Go
exec sp_rename 'Area_Tbl','Tbl_Area','Object'
```

这样可以将表 Area_Tbl 更名为 Tbl_Area,因为更改表名,所以类型为“Object”。

拓展提高

修改表名将引起使用该表的视图、存储过程或函数产生“找不到对象”的错误,因此在更改表名前,必须确定是否有视图、存储过程或函数引用了该表。如果有,而且一定要更改该表名,要记得更改引用该表的视图、存储过程或函数中该表表名为对应的新表的表名。

2. 设置字段属性

本节介绍在 SQL Server Management Studio 中如何修改字段名及其相关属性。

打开 SQL Server Management Studio 并连接到数据库服务器，在“对象资源管理器”中展开“数据库”文件夹，接着展开“数据库”/“DataCollationSys”节点，展开“表”文件夹，右键单击要更改名字的表 Area_Tbl，执行“修改”命令，如图 1-9 所示。

在如图 1-10 所示界面中可以修改字段名，数据类型以及字段的其他相关属性。



图 1-9 展开表字段



图 1-10 修改表字段

在图 1-10 中，可以修改字段 Area 的名字为 AreaName，并将字段类型改为 char 长度改为 15，并且不允许改字段为 Null。在“列名”列中单击 Area 字段，然后输入 AreaName，将 Area 改为 AreaName，在“数据类型”列中，选择 char 并键入 15 将数据类型改为 char，长度改为 15，在“允许空”列中去掉“√”将该字段设置为不允许为空，然后单击工具条上的“保存”按钮保存修改，这样就修改了表字段及其相关的属性。

也可以采用编码的方式使用 T-SQL 来修改表字段名及其相关的属性。使用存储过程 sp_rename 修改字段名。sp_rename 的语法在上面修改表名中已介绍，这里不再介绍，直接看看使用 sp_rename 存储过程修改字段 Area 为 AreaName 的 SQL 语句。

```
USE DataCollationSys
GO
EXEC sp_rename 'Area_Tbl.Area', 'AreaName', 'COLUMN'
```

chapter
01

chapter
02

chapter
03

chapter
04

chapter
05

因为是修改表字段，所以必须在第一个参数中指明要修改的字段属于哪个表，在第三个参数中指定修改的对象类型为 COLUMN。



拓展提高

同修改表名一样，修改表字段类型也会影响使用该字段的视图、存储过程或函数。因此在修改了字段名后，别忘了将引用该字段的视图、存储过程或函数中相应的字段名改成新的字段名。

使用 T-SQL 修改字段数据类型、长度及其他属性的 SQL 语法：

```
ALTER TABLE table
{ [ ALTER COLUMN column_name
  { new_data_type [ ( precision [ , scale ] ) ]
    [ COLLATE < collation_name > ]
    [ NULL | NOT NULL ]
    | {ADD | DROP } ROWGUIDCOL }
  ] }
```

其中相关参数说明如下：

ALTER TABLE 和 ALTER COLUMN：主关键字。

table：修改字段的表的名字；

Column_Name：是要修改的表字段名。

new_data_type：指定新的数据类型以及相关的长度。

COLLATE：指定字段使用的字符集；

NULL|NOT NULL：指明字段是否允许为空。

例如：修改 Area_Tbl 字段 Area 的 SQL 语句如下：

```
USE DataCollationSys
Go
ALTER TABLE Area_Tbl
ALTER COLUMN Area char(15) NOT NULL
```

注意：不是所有的列都可以改变。通常，下列类型的列不能修改：

属于主键或外键约束的列；

用于复制的列；

具有 text, ntext, image 或 timestamp 数据类型的列；

在索引中使用的列；

用于检查或约束的列；

用于计算的列；

通过明确地执行 CREATE STATISTICS 语句创建统计的列。

3. 添加表列

本小节介绍如何在 SQL Server Management Studio 中添加列。可以在如图 1-4 所示的添加表列界面中添加表列，直接在“列名”列中输入要新添加的列名，在“数据类型”列中为新添加的列选择数据类型，并指定数据长度，在“允许空”列中设定该列是否允许为空。你也可以在 SQL Server Management Studio 中连接数据库服务后，依次展开“数据库”文件夹、你要添加列的表所属的数据库（这里是 DataCollationSys 数据库）、“表”文件夹、你要修改的表（这里是 Area_Tbl 表）、“列”文件夹，右键单击“列”文件夹，在弹出菜单中执行“新建列”命令，如图 1-11 所示界面。接着会出现图 1-4 所示的添加表列界面，在该界面中进行添加表列。



图 1-11 新建列

同样，也可以采用编码的方式来为表添加新列。添加表列的语法如下：

```
ALTER TABLE table_name
ADD column_name datatype NULL|NOT NULL
DEFAULT default_value
```

其中相关参数说明如下：

ALTER TABLE、ADD 和 DEFAULT： 主关键字。

table_name： 指定要添加表列的表；**column_name：** 新添加的表列的名字。

datatype： 新添加的表列的数据类型；**NULL：** 设定该字段允许为空；**NOT NULL：** 设定该字段不允许为空。

default_value： 为该字段指定缺省值，当向该表插入数据时，如果没有为该字段指定要插入的数据时，将向该字段插入缺省值，缺省值项是可选的。

下面的 SQL 语句实现向表 Area_Tbl 中添加字段“memo”：

```
USE DataCollationSys
Go
```

chapter
01

chapter
02

chapter
03

chapter
04

chapter
05

```
ALTER TABLE Area_Tbl  
ADD memo varchar(200) NULL
```

4. 删除表列

本小节介绍在 SQL Server Management Studio 中删除表列。打开 SQL Server Management Studio, 连接数据库服务, 依次展开“数据库”文件夹、所要删除列的表所属的数据库(这里是 DataCollationSys 数据库)、“表”文件夹、所要修改的表(这里是 Area_Tbl 表)、“列”文件夹, 右键单击要删除的字段, 在弹出的菜单中执行“删除”命令, 如图 1-12 所示界面, 在接着出现的界面中单击“确认”按钮删除表列。也可以在如图 1-13 所示的界面中, 选中要删除的列, 右键单击该列, 在弹出的菜单中执行“删除列”命令删除表列。

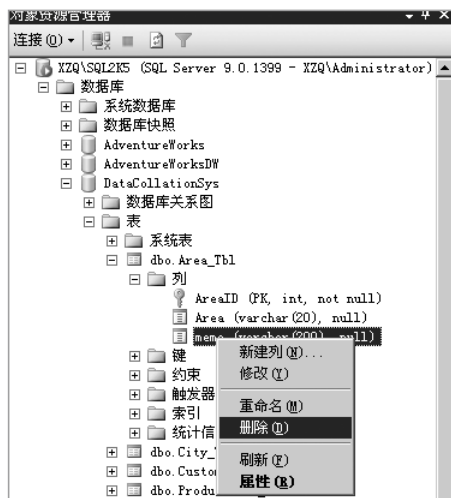


图 1-12 删除表列



图 1-13 在表设计界面中删除列

同样你也可以采用编码的方式编写 SQL 脚本删除表列。删除表列的语法如下:

```
ALTER TABLE table_name  
DROP COLUMN column_name
```

其中 ALTER TABLE 和 DROP COLUMN 是主关键字, table_name 是要删除字段的表, column_name 是要删除的字段名。

下面的 SQL 脚本是删除表 Area_Tbl 中的字段 memo(先前新添加的列):

```
USE DataCollationSys  
Go  
ALTER TABLE Area_Tbl  
DROP COLUMN memo
```



拓展提高

需要注意的是删除列后，列中的数据内容也随之一起被删除。另外，像修改表字段一样，不是所有的字段都可以删除。下面类型的字段不能删除：

- 用于主键或外键的字段；
- 用于复制的字段；
- 用作索引中的列（除非先删除索引）；
- 符合规则的列；
- 与默认值关联的列。

删除数据库表非常简单，打开 SQL Server Management Studio，连接数据库服务，依次展开“数据库”文件夹、表所在的数据库、“表”文件夹，右键单击要删除的表，在弹出式菜单中执行“删除”命令，接着单击“确定”按钮完成表的删除，如图 1-14 所示。

也可以在查询分析器中编写 SQL 脚本删除表，删除表的语法如下：

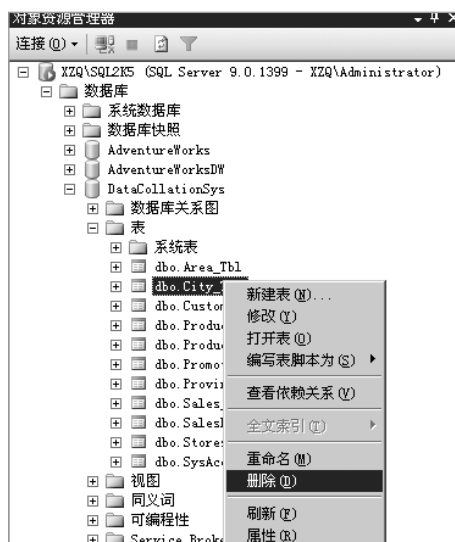


图 1-14 删除表

```
DROP TABLE table_name
```

主关键字 DROP TABLE 表示要删除表，table_name 指定要删除的表的名称。例如删除 City_Tbl 表，SQL 语句如下：

```
DROP TABLE City_Tbl
```

任务三：创建或删除主键和外键



任务描述

小张是某学校高二年级的年级主任，期中考试过后，小张想创建数据库表来统计

chapter
01

chapter
02

chapter
03

chapter
04

chapter
05

学生的考试成绩，要求能够根据学生成绩的数据库表来对全校学生进行排名。

任务分析

由于在 SQL Server 的关系数据库中，表与表间通过关系进行关联，最重要的关系就是主键和外键。因此，小张决定通过创建或删除主键和外键来完成这一任务。

准备知识

主键是表中唯一标识一行的约束，主键通常定义在一列中，有时也定义在几列中，通过几列组合在一起唯一标识一行，通过主键可强制表的实体完整性，即确保数据库中所代表的任何事物都不存在重复的数据，每一个表只能有一个主键约束，而且主键不接受空值。

拓展提高

由于主键约束确保唯一数据，所以经常用来定义标识列。所谓标识列，指表中已指派了标识属性的列。标识属性生成唯一数字。

例如表 Area_Tbl 中 AreaID 是主键，因此字段 AreaID 唯一标识一行，如表 1-6 所示。

表 1-6 Area Tbl 样例数据

AreaID	Area
2	华东区
3	华中区
4	华北区
5	西南区
7	华南区

外键是用于建立和加强两个表数据之间的链接的一列或多列。外键一般建立在一列上，有时也通过几列组合起来建立外键，通过将一个表中的主键值的一列或多列添加到另一个表，可以创建两个表之间的链接，这个列就成为第二个表的外键。

外键约束并不仅仅只可以与另一个表的主键约束相链接，它还可以定义为引用另一个表的唯一约束。外键约束不允许空值，但是，如果任何组合外键的列包含空值，则跳过外键约束的校验。

如图 1-15 所示的表 Area_Tbl 和 Province_Tbl，表 Province_Tbl 中 AreaID 是表 Area_Tbl 的主键，把 AreaID 字段加入表 Province_Tbl 中，那么在表 Province_Tbl 中 AreaID 字段就是外键，表 Area_Tbl 和 Province_Tbl 通过 AreaID 字段关联。

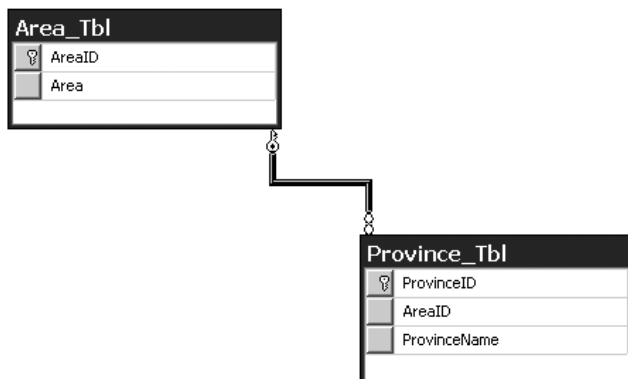


图 1-15 主键和外键关联示例



拓展提高

尽管外键的主要目的是控制存储在外键表中的数据，但它还可以控制对主表数据的修改。

例如图 1-15 中，如果删除了表 Area_Tbl 中的数据，那就破坏了表 Area_Tbl 与 Province_Tbl 关联的完整性，如果删除了表 Area_Tbl 中的一条记录数据，那么表 Province_Tbl 中与该条记录关联的数据就因为没有关联而变得孤独。外键约束就是防止这种情况的发生，如果有外键约束，要删除表 Area_Tbl 中的某条与表 Province_Tbl 关联的数据，那么将发生错误。



任务实施

1. 创建表主键和外键

像创建其他数据库对象一样，既可以在 SQL Server Management Studio 中创建主键和外键，也可以通过编码的方式，用 T-SQL 在查询分析器中创建主键和外键。此外，既可以在创建表的同时创建主键和外键，也可以在创建表后，将表的某个字段或某几个字段设置为主键和外键。首先看看在 SQL Server Management Studio 中创建主键和外键。

STEP 1 打开 SQL Server Management Studio，建立数据库链接。依次展开“数据库”文件夹、你要添加主键和外键的表所属的数据库（这里是 DataCollationSys 数据库）、“表”文件夹，如果在新建表的同时创建主键和外键，单击右键执行“新建表”命令，如图 1-3 所示。在出现如图 1-4 所示的界面中，右键单击选择作为主键或外键的列，在弹出菜单中执行“设置主键”命令，如图 1-16 所示，这样就将选择的列设置为主键了。

STEP 2 创建外键的过程稍微要复杂一些，在图 1-16 所示的界面中，执行“关系”命令，在出现的界面中单击“添加”按钮添加关系，如图 1-17 所示。

chapter
01chapter
02chapter
03chapter
04chapter
05

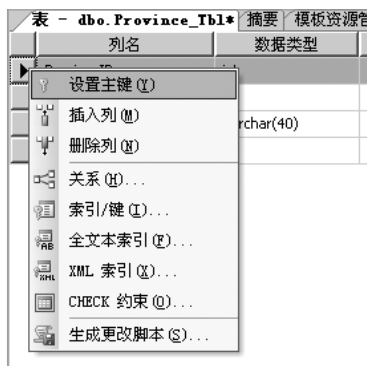


图 1-16 创建表主键

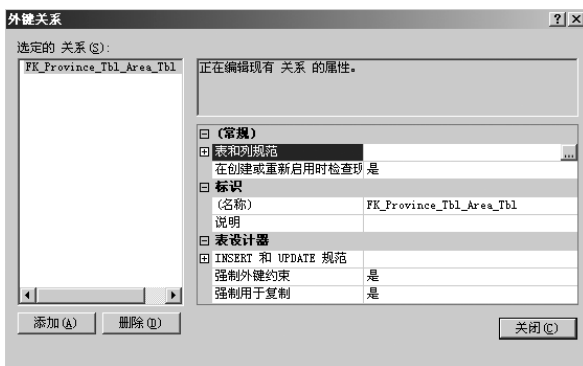



图 1-17 外键关系

STEP 3 在图 1-17 中，选择新添加的关系，在右边的关系属性中，单击“表和列规范”编辑框旁的  按钮，出现图 1-18 的界面。在“主键表”下拉框中选择主键所在的表，例如 Area_Tbl，当鼠标光标移到下面的编辑框中时，将出现一个下拉列表框，列出所选主键表中的所有字段，选择主键，例如 AreaID，然后将鼠标光标移动到右边“外键表”下面的编辑框中，在出现的下拉列表框中选择相应的字段，例如 AreaID，单击“确定”按钮，这样，外键表中的选择字段就成为外键，外键就这样被创建了。

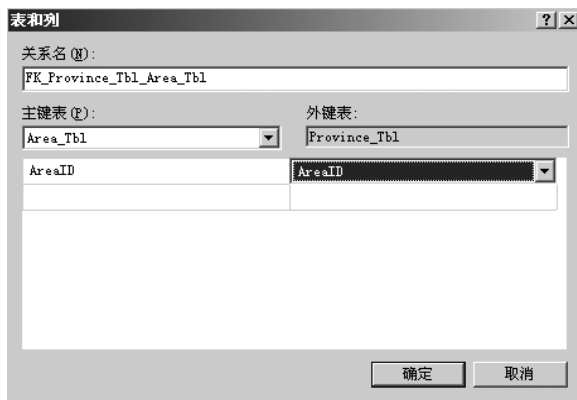


图 1-18 表和列规范

STEP 4 通过修改表来设置表的主键，操作方式和创建表时创建主键一样。打开 SQL Server Management Studio，建立数据库链接。依次展开“数据库”文件夹、你要创建主键和外键的表所属的数据库（这里是 DataCollationSys 数据库）、“表”文件夹。右键单击所创建主键和外键的表（这里是 Province_Tbl），在弹出菜单中执行“修改”命令，出现如图 1-19 所示界面（参考前面修改表的项目）。此后，创建主键和外键的方法同新建表时创建主键和外键的方法一样。

下面来看看如何通过编码的方式在查询分析器中编写 SQL 语句创建主键和外键。以下是在创建表 Province_Tbl 的同时创建主键 ProvinceId 和外键 AreaID 的 SQL 语句：

```
CREATE TABLE [dbo].[Province_Tbl] (  
    [ProvinceId] [int] IDENTITY(1,1) NOT NULL,  
    [AreaID] [int] NULL,
```

```
[ProvinceName] [varchar](50) NULL,  
CONSTRAINT [PK_Province_Tbl] PRIMARY KEY CLUSTERED  
(  
    [ProvinceId] ASC  
)  
) ON [PRIMARY]
```

其中 CONSTRAINT 表示要创建一个约束，PRIMARY KEY 表示该约束是一个主键约束。

创建外键的 SQL 语句如下：

```
ALTER TABLE [dbo].[Province_Tbl] WITH CHECK ADD CONSTRAINT  
[FK_Province_Tbl_Area_Tbl] FOREIGN KEY([AreaID])  
REFERENCES [dbo].[Area_Tbl] ([AreaID])
```

创建外键都是通过修改表，向表添加外键约束来实现的。

通过修改表实现创建主键和外键。创建主键语法：

```
ALTER TABLE table  
[ WITH CHECK | WITH NOCHECK ] ADD  
    CONSTRAINT constraint_name  
PRIMARY KEY [ CLUSTERED | NONCLUSTERED ]  
( column [ ,...n ] )  
[ WITH FILLFACTOR = fillfactor ]  
[ ON { filegroup | DEFAULT } ]  
]
```

其中相关参数说明如下：

ALTER TABLE：表示要修改表；table 是要修改的表名。

WITH CHECK：为可选项，表示在创主键时要检验表中已有数据是否符合主键约束的要求，即数据的唯一性；**WITH NOCHECK**：表示不进行数据检验。

ADD CONSTRAINT：表示要添加约束；constraint_name：要添加的约束标识名称；

PRIMARY KEY：关键字标明该约束是主键约束。

column：是要作为主键的字段，可以是多个字段，字段间用逗号分隔。

WITH FILLFACTOR = fillfactor：为可选项，为主键索引指定填充因子，这将在后面创建索引一节中进行详细介绍。

例如，在创建表 Province_Tbl 后，为表 Province_Tbl 创建主键 PK_Province_Tbl，SQL 语句如下：

chapter
01chapter
02chapter
03chapter
04chapter
05

```
ALTER TABLE [dbo].[Province_Tbl]
WITH CHECK ADD CONSTRAINT [PK_Province_Tbl]
PRIMARY KEY([ProvinceId])
```

创建外键的语法如下：

```
ALTER TABLE table
[ WITH CHECK | WITH NOCHECK ] ADD
CONSTRAINT constraint_name
FOREIGN KEY
    [ ( column [ ,...n ] ) ]
REFERENCES ref_table [ ( ref_column [ ,...n ] ) ]
[ ON DELETE { CASCADE | NO ACTION } ]
[ ON UPDATE { CASCADE | NO ACTION } ]
[ NOT FOR REPLICATION ]
```

其中相关参数说明如下：

ALTER TABLE：为主关键字，表示要修改表。**table**：所要修改的表的名称。

[WITH CHECK | WITH NOCHECK]：为可选项，表示在创建外键约束时是否要对表已有数据进行规则检验。

ADD CONSTRAINT：为关键字，表示要添加约束。**constraint_name**：添加的约束的标识名称。**FOREIGN KEY**：为关键字，表示添加外键。

Column：指定要作为外键的表字段，可以是多个字段，字段间用逗号分隔。

REFERENCES：为关键字，表示要参照表。

ref_table：表示要参照的表的名称。**ref_column**：表示要参照的字段。

[ON DELETE { CASCADE | NO ACTION }]、**[ON UPDATE { CASCADE | NO ACTION }]**、**[NOT FOR REPLICATION]**：为可选项，分别表示对表数据做删除、更新和复制时采取的相应的级联动作。

例如，在表 Province_Tbl 创建好后，要在表字段 AreaID 上创建外键 FK_Province_Tbl_Area_Tbl 和表 Area_Tbl 关联，SQL 语句如下：

```
ALTER TABLE [dbo].[Province_Tbl]
WITH CHECK ADD CONSTRAINT [FK_Province_Tbl_Area_Tbl]
FOREIGN KEY([AreaID])
REFERENCES [dbo].[Area_Tbl] ([AreaID])
```

2. 删除主键和外键

当要删除对列或列组合中的输入值的唯一性要求时，应删除主键约束。当要删除强制引用完整性要求时，应删除外键。

在 SQL Server Management Studio 中删除主键的方法如下。

STEP 1 在 SQL Server Management Studio 中连到数据库服务器，在“对象资源管理器”中展开“数据库”文件夹，展开“数据库”/“DataCollationSys”/“表”文件夹，右键单击要删除主键或外键的表 Area_Tbl，执行“修改”命令，如图 1-19 所示。

STEP 2 在出现的界面中用右键单击主键所在的字段，在弹出式菜单中执行“移除主键”命令，然后保存对表的修改，这样主键就从表中删除了，如图 1-20 所示。

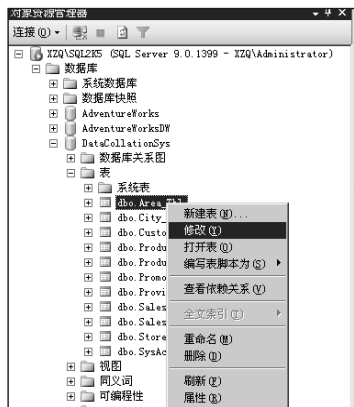


图 1-19 修改表



图 1-20 删除主键

STEP 3 在 SQL Server Management Studio 中删除外键约束。在如图 1-19 所示的界面中，右键单击任何行，在弹出式菜单中执行“关系”命令。在接着出现的界面中选中要删除的外键约束后，单击“删除”按钮删除外键约束，如图 1-21 所示。

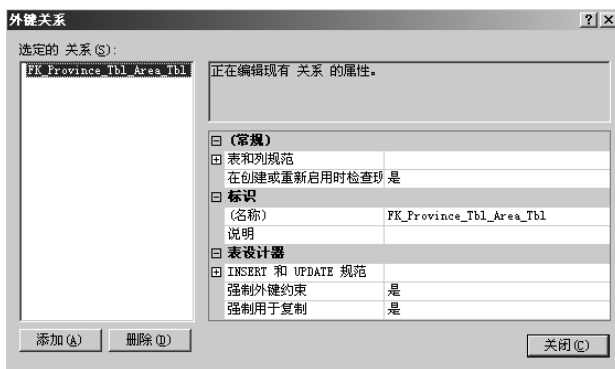


图 1-21 删除外键约束

同样，删除主键、外键约束也可以在查询分析器中通过编写 SQL 语句来进行。删除主键、外键约束的语法如下：

```
ALTER TABLE table
DROP [ CONSTRAINT ] constraint_name
```

其中相关参数说明如下：

chapter
01chapter
02chapter
03chapter
04chapter
05

主关键字 ALTER TABLE 表示要修改表。table 是要修改的表名。

DROP [CONSTRAINT] 表示要删除约束，CONSTRAINT 是可选的关键字。

constraint_name: 要删除的约束的名称。

例如删除表 Area_Tbl 中的主键 PK_Area_Tbl, SQL 语句如下:

```
ALTER TABLE Area_Tbl  
DROP CONSTRAINT PK_Area_Tbl
```

删除表 Area_Tbl 中的外键约束 FK_Province_Tbl_Area_Tbl, SQL 语句如下:

```
ALTER TABLE Area_Tbl  
DROP CONSTRAINT FK_Province_Tbl_Area_Tbl
```

项目小结

本项目主要讲解了在 SQL Server 2008 中如何创建和维护数据库, 并从创建数据库表、修改和删除数据库表、认识主键和外键等方面做了具体介绍。

项目考核



填空题

- (1) 数据 (Data) 实际上就是 _____。
- (2) 数据库是数据的集合, 它具有统一的 _____ 并存放于统一的 _____ 内, 是多种应用数据的集成, 并可被各个应用程序所共享。
- (3) 由于 _____ 约束确保唯一数据, 所以经常用来定义标识列。



判断题

- (1) 应用软件是由数据库系统所提供的数据库管理系统 (软件) 及数据库系统开发工具所书写而成的。 ()
- (2) 修改表名不会引起使用该表的视图、存储过程或函数产生“找不到对象”的错误, 读者可放心使用。 ()



问答题

- (1) 简述数据和数据库的概念。
- (2) 简述创建数据库表的意义。
- (3) 简述主键和外键在数据库中的地位, 并简述主键和外键的创建过程。