



数据库是存放数据的容器,在设计一个应用系统时,必须先设计数据库。数据库中的数据及相关信息通常被存储在一个或多个磁盘文件(即数据库文件)中,而数据库管理系统为用户或数据库应用程序提供统一的接口来访问和控制这些数据,使得用户不需要直接访问数据库文件。

数据库中最重要的对象是数据表,简称表(table),表中存储了数据库的数据。对数据库和表的操作是开发人员的一项重要工作。

5.1 SQL Server 数据库概述

SQL Server 数据库是存放表和视图、索引、存储过程和触发器等数据库对象的逻辑实体,从逻辑角度组织与管理数据。

5.1.1 数据库文件类型

在 SQL Server 中,数据库是由数据文件和事务日志文件组成的,一个数据库至少应包含一个数据文件和一个事务日志文件。包括系统数据库在内的每个数据库都有自己的文件集,而且不与其他数据库共享这些文件。SQL Server 数据库具有如下三种类型的文件。

1. 主数据文件

主数据文件是数据库的起点,其中包含数据库的初始信息,记录数据库所拥有的文件指针。每个数据库有且仅有一个主数据文件,这是数据库必需的文件。主数据文件的扩展名是 .mdf。

2. 辅助数据文件

除主数据文件以外的所有其他数据文件都是辅助数据文件。辅助数据文件存储主数据文件未存储的所有其他数据和对象,它不是数据库必需的文件。当一个数据库需要存储的数据量很大(超过了 Windows 操作系统对单一文件大小的限制)时,可以用辅助数据文件来保存主数据文件无法存储的数据。辅助数据文件可以分散存储在不同的物理磁盘中,从而提高数据的读写效率。辅助数据文件扩展名为 .ndf。

3. 事务日志文件

在 SQL Server 中,每个数据库至少拥有一个自己的日志文件,也可以拥有多个日志文件。日志文件最小是 1MB,用来记录所有事务以及每个事务对数据库所做的修改。日志文件的扩展名是 .ldf。

在创建数据库的时候,日志文件也会随之被创建。如果系统出现故障时,常常需要使用

事务日志将数据库恢复到正常状态。这是 SQL Server 的一个重要的容错特性,它可以有效地防止数据库的损坏,维护数据库的完整性。

在 SQL Server 中,用户还可以指定数据文件的大小能够自动增长。在定义数据文件时,指定一个特定的增量,每次扩大文件时均按此增量来增长。另外,每个文件的大小可以指定一个最大值,当文件大小达到最大值时,就不再增长。如果没有指定文件最大值,文件可以一直增长到磁盘没有可用空间为止。

5.1.2 数据库文件组

为了有助于数据布局和管理任务,SQL Server 允许用户将多个文件划分为一个文件集合,这些文件可以在不同的磁盘上,并为这一集合命名,这就是文件组。

文件组是数据库中数据文件的逻辑组合,数据库文件组有主文件组、用户定义文件组和默认文件组三类。

1. 主文件组

主文件组是包含主要文件的文件组。所有系统表 and 没有明确分配给其他文件组的任何文件都被分配到主文件组中,一个数据库只有一个主文件组。

2. 用户定义文件组

用户定义文件组是用户首次创建数据库时,或修改数据库时自定义的,其目的是将数据存储进行合理的分配,以提高数据的读写效率。

3. 默认文件组

每个数据库中均有一个文件组被指定为默认文件组。如果在数据库中创建对象时没有指定对象所属的文件组,对象将被分配给默认文件组。在任何时候,只能将一个文件组指定为默认文件组。

关于默认文件组有如下说明:

- (1) 默认文件组中的文件必须足够大,能够容纳未分配给其他文件组的所有新对象。
- (2) 如果没有指定默认文件组,则将主文件组作为默认文件组。
- (3) PRIMARY 文件组是默认文件组。

5.2 SQL Server 数据库基本管理

在 SQL Server 中,所有类型的数据库管理操作有两种方式:一是 SSMS 图形化界面方式;二是 Transact-SQL 语句代码方式。

5.2.1 创建用户数据库

创建数据库就是为数据库确定名称、大小、存放位置、文件名和所在文件组的过程。在一个 SQL Server 实例中,最多可以创建 32 767 个数据库,数据库的名称必须满足系统的标识符规则。在命名数据库时,一定要使数据库名称简短并有一定的含义。

例 5.1 创建教学管理数据库 JXGL。主数据文件逻辑名为 JXGL.mdf,保存路径为 D:\JXGLSYS\DATA,日志文件的逻辑名为 JXGL_log.ldf,保存路径为 D:\JXGLSYS\Data_log。主数据文件大小为 5MB,文件大小不受限制,增长量为 1MB;日志文件的初始

(6) 在图 5.2 的“数据库文件”列表中,包括两行:一行是行数据文件,另一行是日志文件。通过单击下面相应的按钮,可以添加或者删除相应的数据文件。该列表中各字段值的含义如下。

逻辑名称:指定该文件的文件名。

文件类型:用于区别当前文件是数据文件还是日志文件。

文件组:显示当前数据库文件所属的文件组。

初始大小:指定该文件的初始容量,在 SQL Server 2012 中数据文件的默认值为 5MB,日志文件的默认值为 2MB。

自动增长:用于设置在文件的容量不够用时,文件根据何种增长方式自动增长。因为本例中日志文件的最大值为 20MB,通过单击“自动增长”列中的 JXGL_log 省略号按钮,打开“更改 JXGL_Log 的自动增长设置”对话框进行设置,如图 5.3 所示。做日志文件大小修改,本例中设为 20。

路径:指定存放该文件的目录。

(7) 单击“选项”选择页,设置数据库的排序规则、恢复模式、兼容级别和其他需要设置的内容,如图 5.4 所示。

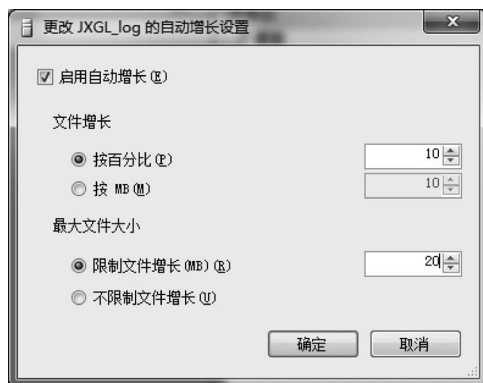


图 5.3 JXGL_log 自动增长设置修改



图 5.4 “选项”选择页

(8) 单击“文件组”选择页,可以设置数据库文件所属的文件组,还可以通过“添加”或者“删除”按钮更改数据库文件所属的文件组,如图 5.5 所示。

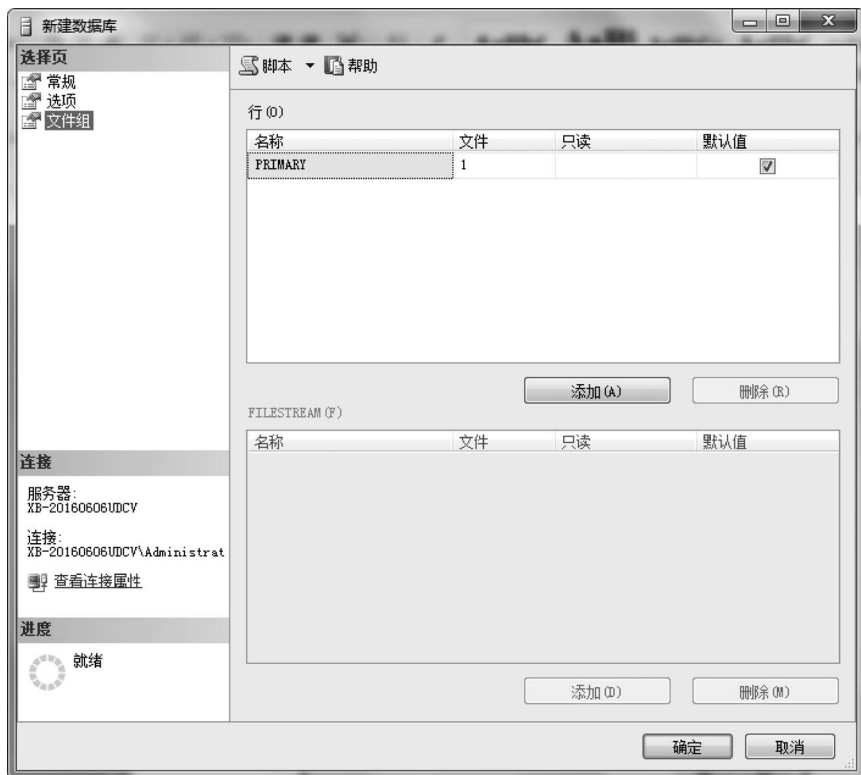


图 5.5 “文件组”选择页

(9) 完成以上操作后,就可以单击“确定”按钮关闭“新建数据库”对话框。至此,便成功创建了一个数据库。

可以通过“对象资源管理器”窗口查看新建的数据库。

注意: 在 SQL Server 中创建新的对象时,它可能不会立即出现在“对象资源管理器”窗口中,可右击对象所在位置的上一层文件夹,并选择“刷新”命令,即可强制 SQL Server 重新读取系统表并显示出数据中的所有新对象。

2. 利用 Transact-SQL 语句

利用 SSMS 工具创建数据库可以方便应用程序对数据的直接调用。但是,在有些情况下,不能使用图形化方式创建数据库。比如,在设计一个应用程序时,开发人员会直接使用 Transact-SQL 语句在程序代码中创建数据库及其他数据库对象,而不用在制作应用程序安装包时再放置数据库或让用户自行创建。

可以利用 Transact-SQL 所提供的 CREATE DATABASE 语句来创建数据库,语句格式如下:

```
CREATE DATABASE <数据库名>
ON
{[PRIMARY](NAME = <逻辑文件名>
  FILENAME = <物理文件名>
```

```

    [, SIZE = <初始大小>]
    [, MAXSIZE = {<文件最大长度>|UNLIMITED}]
    [, FILEGROWTH = <文件增长幅度>])
}[, ... n]
LOG ON
{[PRIMARY](NAME = <逻辑文件名>,
    FILENAME = <物理文件名>
    [, SIZE = <初始大小>]
    [, MAXSIZE = {<文件最大长度>|UNLIMITED}]
    [, FILEGROWTH = <文件增长幅度>])
}[, ... n]

```

这里大括号{}用来表示语句块。

参数说明如下。

<数据库名>: 新建数据库的名称,可长达 128 个字符。

ON: 指定显式定义,用来存储数据库数据部分的磁盘文件(数据文件)。

PRIMARY: 在主文件组中指定文件。

LOG ON: 指定显式定义,用来存储数据库日志的磁盘文件(日志文件)。

NAME: 用来定义数据库的逻辑名称,这个逻辑名称用来在 T_SQL 代码中引用数据库。

FILENAME: 用于定义数据库文件在硬盘上的存放路径与文件名称。这必须是本地目录(不能是网络目录),并且不能是压缩目录。

SIZE: 用来定义数据文件的初始大小,可以使用 KB、MB、GB 或 TB 为计量单位。如果没有为主数据文件指定大小,那么 SQL Server 将创建与 model 系统数据库相同大小的文件。如果没有为辅助数据库文件指定大小,那么 SQL Server 将自动为该文件指定 1MB 大小。

MAXSIZE: 用于设置数据库允许达到的最大长度,可以使用 KB、MB、GB、TB 为计量单位,也可以为 UNLIMITED,或者省略整个子句,使文件可以无限制增长,直至磁盘被充满为止。在高版本的 SQL Server 中,规定日志文件可增长的最大长度为 2TB,而数据文件的最大长度为 16TB。

FILEGROWTH: 用来定义文件增长所采用的递增量或递增方式。可以使用 KB、MB 或百分比(%)为计量单位。如果没有指定这些符号之中的任一符号,则默认 MB 为计量单位。

下面利用 CREATE DATABASE 语句完成例 5.1 中教学数据库的创建。步骤如下:

(1) 在 Windows 环境中,创建文件夹 D:\JXGLSYS\DATA。

(2) 在 Microsoft SQL Server Management Studio 集成环境窗口中,单击标准工具栏上的“新建查询”按钮,创建一个查询输入窗口。

(3) 在查询窗口内输入如下 CREATE DATABASE 语句:

```

CREATE DATABASE JXGL
ON
( NAME = JXGL,
  FILENAME = 'D:\JXGLSYS\DATA\JXGL.mdf',
  SIZE = 5,

```

```

FILEGROWTH = 1
)
LOG ON
(NAME = JXGL_log,
FILENAME = 'D:\JXGLSYS\DATA\JXGL_log.ldf',
SIZE = 2,
MAXSIZE = 20,
FILEGROWTH = 10 %
)

```

(4) 单击工具栏中的“执行”按钮  运行程序语句。如果执行成功,在“查询”结果窗口中,可以看到一条“命令已成功完成。”的消息。然后在“对象资源管理器”窗口中刷新,展开数据库结点就能看到刚创建的 JXGL 数据库。

注意: 如果感觉以后数据库会不断增长,那么就指定其自动增长方式;反之,最好不要指定其自动增长,以提高数据的使用效率。

在创建数据库时,所要创建的数据库名称必须是系统当中不存在的。如果存在相同名称的数据库,在创建数据库时系统将会报错。因此,一般在创建数据库前先用下列语句进行判断。

```

IF EXISTS (SELECT * FROM sysdatabases WHERE name = 'JXGL')
PRINT 'JXGL 数据库已经存在'

```

5.2.2 数据库结构的修改

创建数据库后,还可以对数据库结构进行修改,通常包括增加/删除文件,修改文件属性(包括更改文件名和文件大小)、修改数据库选项等。

1. 利用 SSMS 图形化方式

对于已经建立的数据库,可以利用 SSMS 图形化方式查看或修改数据库信息。在“对象资源管理器”窗口中,右击要修改结构的数据库(如教学管理数据库 JXGL),选择“属性”命令,出现“数据库属性-JXGL”窗口,如图 5.6 所示。

可以在“数据库属性-JXGL”窗口所包含的“常规”“文件”“文件组”“选项”“更改跟踪”“权限”“扩展属性”“镜像”“事务日志传送”9 个选择页中修改数据库的相关信息。

常规: 查看所选数据库的常规属性信息。

文件: 查看或修改所选数据库的数据文件和日志文件属性。

文件组: 查看文件组,或为所选数据库添加新的文件组。

选项: 查看或修改所选数据库的选项,包括所选数据库的排序规则、恢复模式和兼容级别等信息。

更改跟踪: 查看或修改所选数据库的更改跟踪设置,启用或禁用数据库的更改跟踪。

权限: 查看或设置安全对象的权限,包括用户、角色和权限信息。

扩展属性: 通过使用扩展属性向数据库对象添加自定义属性,也可以查看或修改所选对象的扩展属性。

镜像: 查看或设置镜像的主体服务器、镜像服务器和见证服务器。

事务日志传送: 配置和修改数据库的日志传送属性。



图 5.6 数据库属性窗口

2. 利用 Transact-SQL 语句

在 SQL Server 服务器上,可能存在多个用户数据库。默认情况下,用户连接的是 master 系统数据库。在 Transact-SQL 中用 USE 语句来完成不同数据库之间的切换,语句格式如下:

USE <数据库名>

其中,<数据库名>为所要选择的数据库的名称。

1) 查看数据库信息

在 Microsoft SQL Server 系统中,查看数据库信息有很多种方法,例如,可以使用 4.1 节中的系统表 sysdatabases 和 sysobjects,还可以使用目录视图、函数和系统存储过程等查看有关数据库的基本信息。下面分别来介绍几种查看数据库信息的基本方式。

(1) 利用目录视图。常见的查看数据库基本信息的视图有以下几种。

- ① sys.databases: 查看有关数据库的基本信息。
- ② sys.database_files: 查看有关数据库文件的信息。
- ③ sys.filegroups: 查看有关数据库文件组的信息。
- ④ sys.master_files: 查看数据库文件的基本信息和状态信息。

(2) 利用函数。常见的查看数据库基本信息的函数有以下几种。

① DATABASEPROPERTYEX(<数据库名>,<选项>): 返回指定数据库中指定选项的属性。如查看教学管理系统数据库 JXGL 的 Version 选项的设置信息:

```
SELECT DATABASEPROPERTYEX('JXGL','Version')
```

② DB_ID(<数据库名>)：返回指定数据库名称对应的 id。如查看 JXGL 数据库的 id：

```
SELECT DB_ID('JXGL')
```

③ DB_NAME(<id 号>)：返回指定数据库 id 号的数据库名称。如查看 id 号是 3 的数据库名称：

```
SELECT DB_NAME(3)
```

(3) 利用存储过程。与数据库属性相关的系统存储过程有以下几种。

① sp_tables：返回在当前数据库环境中查询的对象列表。如查看 JXGL 数据库对象列表：

```
USE JXGL
EXEC sp_tables
```

② sp_help：返回当前数据库对象(在系统表 sysobjects 中的对象)的信息。

③ sp_helpdb [<数据库名>]：显示给定数据库或所有数据库的参数信息。

④ sp_spaceused：查看当前数据库空间信息,如查询数据库 JXGL 的空间信息：

```
USE JXGL
EXEC sp_spaceused
```

可以使用执行存储过程语句 EXEC 来查看相关信息。其中,EXEC 是 EXECUTE 的缩写,在执行一个系统存储过程的时候使用。

2) 修改数据库

Transact-SQL 提供了修改数据库的语句 ALTER DATABASE。

(1) 增加数据库空间。利用 Transact-SQL 语句增加已有数据库文件的大小,语句格式如下：

```
ALTER DATABASE <数据库名>
MODIFY FILE
(FILENAME = <逻辑文件名>,
SIZE = <文件大小>,
MAXSIZE = <增长限制>
)
```

例 5.2 为教学管理数据库 JXGL 增加容量,原来的数据库文件 JXGL.mdf 的初始分配空间为 5MB(默认值),现在将增至到 10MB。

```
USE JXGL
ALTER DATABASE JXGL
MODIFY FILE
(NAME = JXGL,
SIZE = 10)
```

(2) 增加数据库文件。利用 Transact-SQL 语句增加新的数据文件或日志文件,语句格式如下：

```
ALTER DATABASE <数据库名>
ADD FILE|ADD LOG FILE
(NAME = <逻辑文件名>,
```

```

FILENAME = <物理文件名>,
SIZE = <文件大小>,
MAXSIZE = <增长限制>,
FILEGROWTH = <文件增长幅度>
)

```

例 5.3 为数据库 JXGL 增加辅助数据文件 JXGL_1.ndf, 初始大小为 5MB, 最大长度为 30MB, 按照 5% 增长。

```

USE JXGL
ALTER DATABASE JXGL
ADD FILE
(NAME = JXGL_1,
FILENAME = 'D:\JXGLSYS\DATA\JXGL_1.ndf',
SIZE = 5,
MAXSIZE = 30,
FILEGROWTH = 5 %
)

```

(3) 删除数据库文件。利用 ALTER DATABASE 的 REMOVE FILE 子句, 可以删除指定的文件。语句格式如下:

```

ALTER DATABASE <数据库名>
REMOVE FILE <逻辑文件名>

```

例 5.4 删除数据库 JXGL 中的辅助数据文件 JXGL_1.ndf。

```

USE JXGL
ALTER DATABASE JXGL
REMOVE FILE JXGL_1

```

5.2.3 数据库文件的更名、删除

对已存在的用户数据库, 可以对其更改名称, 当不使用该数据库时, 还可以删除。在更名或删除数据库之前, 应该确保没有用户正在使用这个数据库。

1. 利用 SSMS 图形化方式

1) 数据库更名

在 SSMS 的“对象资源管理器”窗口中, 选中要更名的数据库对象, 右击, 在弹出的快捷菜单中选择“重命名”命令。

2) 删除数据库

在 SSMS 的“对象资源管理器”窗口中, 选中要删除的数据库对象, 右击, 在弹出的快捷菜单中选择“删除”命令, 在随后出现的“删除对象”对话框中单击“确定”按钮, 即可完成对指定数据库的删除。

2. 利用 Transact-SQL 语句

1) 更名数据库

在查询窗口执行系统存储过程 sp_renamedb 可以更改数据库的名字。语句格式如下:

```
sp_renamedb <数据库名 1>, <数据库名 2>
```

其中, “数据库名 1”是欲改名的数据库文件名, “数据库名 2”是改名后的数据库文件名。

例 5.5 将已存在的数据库 JXGL 改名为 GX_JXGL。

```
EXEC sp_renamedb 'JXGL', 'GX_JXGL'
```

2) 删除数据库

在查询窗口执行 DROP DATABASE 语句可以删除数据库。语句格式如下：

```
DROP DATABASE <数据库名>
```

例 5.6 删除更名后的数据库 GX_JXGL。

```
DROP DATABASE GX_JXGL
```

5.3 SQL Server 数据表管理

数据表(简称表)是 SQL Server 数据库中最重要数据对象,也是构建高性能数据库的基础。在程序开发与应用过程中,创建数据库的目的是存储、管理和查询数据,而数据表是存储数据的基本单元。

例 5.7 假设教学管理数据库 JXGL 中含有学生表 S、选课表 SC 和课程表 C,结构分别如表 5.1、表 5.2 和表 5.3 所示。

表 5.1 学生表 S 的结构

列 名	描 述	数 据 类 型	允 许 空 值	说 明
SNO	学号	nchar(9)	NO	主键
SNAME	姓名	nchar(8)	NO	
SEX	性别	nchar(2)	YES	
BIRTHDATE	出生日期	date	YES	
COLLEGE	学院	varchar(50)	YES	

表 5.2 选课表 SC 的结构

列 名	描 述	数 据 类 型	允 许 空 值	说 明
SNO	学号	nchar(9)	NO	主键(同时都是外键)
CNO	课程号	nchar(4)	NO	
GRADE	成绩	real	YES	

表 5.3 课程表 C 的结构

列 名	描 述	数 据 类 型	允 许 空 值	说 明
CNO	课程号	nchar(4)	NO	主键
CNAME	课程名	nchar(20)	NO	
DESCRIPTION	课程说明	text	YES	
CREDIT	学分	real	YES	
C_COLLEGE	开课单位	varchar(50)	YES	

在 SQL Server 中,所有类型的表管理操作只有 SSMS 图形化界面或 Transact-SQL 代码命令两种方式。

5.3.1 表的创建与维护

创建表就是定义一个新表的结构及其与其他表之间的关系。表的维护是指在数据库中创建表之后,对表进行修改、删除等操作。修改表是指更改表结构或表之间的关系,而删除

表是指从数据库中去掉表结构、表之间关系和表中所有数据。所谓表结构,就是构成表的列、各列的定义(列名、数据类型、数据精度、列上的约束等)和表上的约束。

1. 利用 SSMS 图形化方式

1) 创建和修改表

SSMS 提供一个前端的、填充式的表设计器以简化表的设计工作,利用图形化的方法可以非常方便地创建数据表。操作步骤如下:

(1) 启动并登录 SQL Server Management Studio,在“对象资源管理器”面板中展开“数据库”结点,可以看到自己创建的数据库,比如 JXGL。展开 JXGL 结点,右击“表”结点,在弹出的快捷菜单中选择“新建表”命令,进入“表设计器”窗口。

(2) 在“列名”栏中输入各个字段的名称,如输入表 S 的各个字段名,在“数据类型”栏中选择相应数据类型并输入字段长度。“允许 Null 值”列的复选框未勾选状态表明该字段不允许“空值”,如图 5.7 所示。



图 5.7 “表设计器”窗口

(3) 单击“保存”按钮,并在弹出的“选择名称”对话框输入表名,本例中输入表名 S。单击“确定”按钮,保存数据表,如图 5.8 所示。

表 SC 和表 C 结构可以用相同的方法创建。

(4) 如果需要修改表结构,展开“数据库”结点,在需要修改的表上右击,从弹出的快捷菜单



图 5.8 “选择名称”对话框

中选择“设计”命令,可重新打开表设计器进行上述操作。

2) 创建完整性约束

数据完整性约束是数据库设计方面的一个非常重要的问题,数据完整性代表数据的正确性、一致性与可靠性。实施完整性的目的在于确保数据的质量,约束是保证数据完整性的重要方法。

在 SQL Server 中,根据数据完整性措施所作用的数据库对象和范围不同,将数据完整性分类如下。

- (1) 实体完整性:把表中的每行看作一个实体,它要求所有行都具有唯一性。
- (2) 域完整性:要求表中指定列的数据具有正确的数据类型、格式和有效的数据范围。
- (3) 参照完整性:维持被参照表和参照表之间的数据一致性。

在 SQL Server 中,可以通过建立“约束”等措施来实现数据完整性约束,约束包括 5 种类型:主键(PRIMARY KEY)约束、唯一性(UNIQUE)约束、检查(CHECK)约束、默认值(DEFAULT)约束和外键(FOREIGN KEY)约束。

(1) 创建主键约束。表中的主键经常为一列或多列属性的组合,其值能唯一地标识表中的每一行。一个表只能有一个主键,而且主属性不能为空值。在表设计器中可以创建和删除主键约束。具体方法如下:

在表设计器中,单击要定义为主键的列,如果要设置多列为主键,则选中所有主键列(按住 Ctrl 或 Shift 键并单击其他列),右击,在弹出的快捷菜单中选择“设置主键”命令,这时,主键列的左边会显示“黄色钥匙”图标,完成主键设置。

在表设计器中,选择主键列,右击,在弹出的快捷菜单中选择“删除主键”命令,则删除了表的主键。

(2) 创建唯一性约束。唯一性约束用来限制非主键列上的数据唯一性。一个表上可以放置多个唯一性约束。唯一性约束可以用于允许空值的列。

在表设计器中可以创建和删除唯一性约束。例如,当学生表 S 中的 SNAME 列的值不能有重复值时,可设置唯一性约束操作步骤如下:

① 在 S 表设计器中右击,在弹出的快捷菜单中选择“索引/键”命令,打开“索引/键”对话框。

② 在弹出的“索引/键”对话框中,单击“添加”按钮添加新的主/唯一键或索引;在“(常规)”栏的“类型”右边选择“唯一键”,在“列”的右边单击  按钮,选择列名 SNAME 和排序规律 ASC(升序)或 DESC(降序),如图 5.9 所示。

③ 设置完成后,单击“关闭”按钮返回表设计窗口,然后单击工具栏中的“保存”按钮,即完成唯一性约束的创建。

(3) 创建检查约束。检查约束用于限制输入一系列或多列值的范围,从逻辑表达式判断数据的有效性,限制不满足 CHECK 约束条件的数据输入。

例如,在学生表 S 中的 BIRTHDATE 列,设置大学生的出生日期为 1988 年 1 月 1 日至 2010 年 1 月 1 日,可以通过 CHECK 约束完成。具体方法如下:

在表设计器中右击任一列,在快捷菜单中选择“CHECK 约束”命令,在弹出的“CHECK 约束”对话框中,单击“表达式”右边的添加按钮 ,在“表达式”文本框中输入检查表达式“[BIRTHDATE]>='1988-01-01' AND [BIRTHDATE]<='2010-01-01'”,然后进行其他选项的设置,如图 5.10 所示。最后单击“关闭”按钮完成设置。

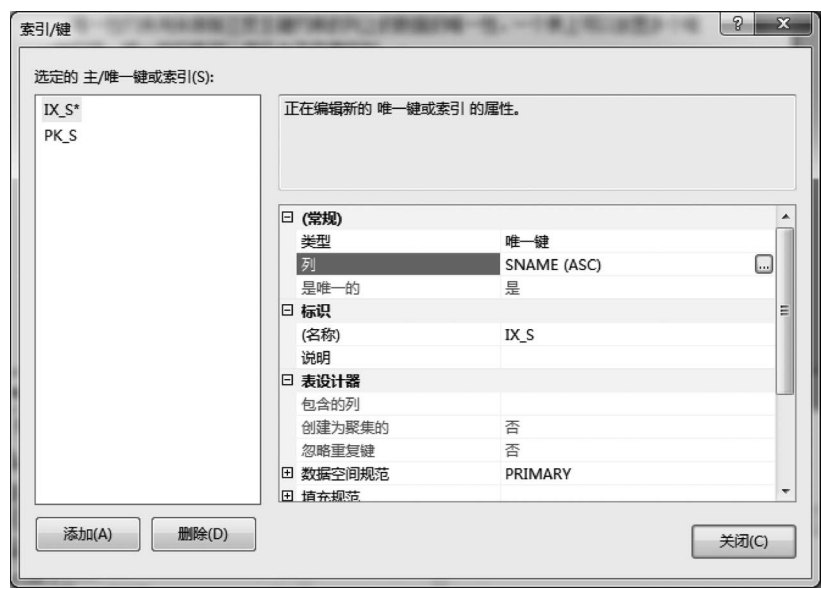


图 5.9 “索引/键”对话框

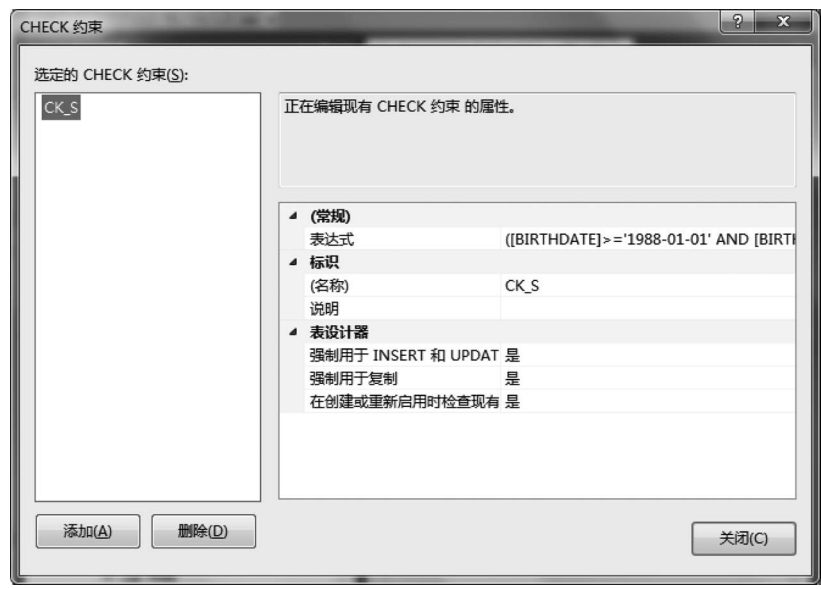


图 5.10 设置 CHECK 约束

(4) 创建默认值约束。若表的某列定义了默认值约束后,用户在插入新的数据行时,如果没有为该列指定数据,那么系统就将默认值赋给该列。当然,该默认值也可以是空值(NULL)。

例如,把学生表 S 的 SEX 列默认值设置为“男”,具体方法如下:
在表设计中,选择需要设置默认值的列,在下面“列属性”的“默认值或绑定”栏中输入默认值“男”,然后单击工具栏中的“保存”按钮,即完成 DEFAULT 约束的创建。

(5) 创建外键约束。外键约束用于建立和加强两个表(主表和从表)的一列或多列数据之间的连接,当数据添加、修改或删除时,通过外键约束保证它们之间数据的一致性。

定义外键约束是先定义主表的主键,再对从表定义外键约束。

例如,在选课表 SC 中定义外键 SNO、CNO。外键约束要求 SC.SNO 的值必须在 S.SNO 中,SC.CNO 的取值必须在 C.CNO 中。设置外键约束操作步骤如下:

① 在 SC 的表设计器中,选择要设置外键的列 SNO 并右击,在快捷菜单中选择“关系”命令,弹出“外键关系”对话框。

② 在“外键关系”对话框中单击“添加”按钮,增加新的外键关系,并对新增的外键关系进行设置。

③ 单击“表和列规范”栏右边的 [...] 按钮,弹出“表和列”对话框。在“表和列”对话框中,如果想重新命名外键约束名,可以在“关系名”文本输入框中输入新的名称;在“主键表”下拉列表框中选择 S 表,并单击“主键表”下的下拉按钮选择其中的 SNO 作为被参照列;在“外键表”文本框中输入当前表名 SC,并单击“外键表”下的下拉按钮选择其中的 SNO 作为参照列,如图 5.11 所示。

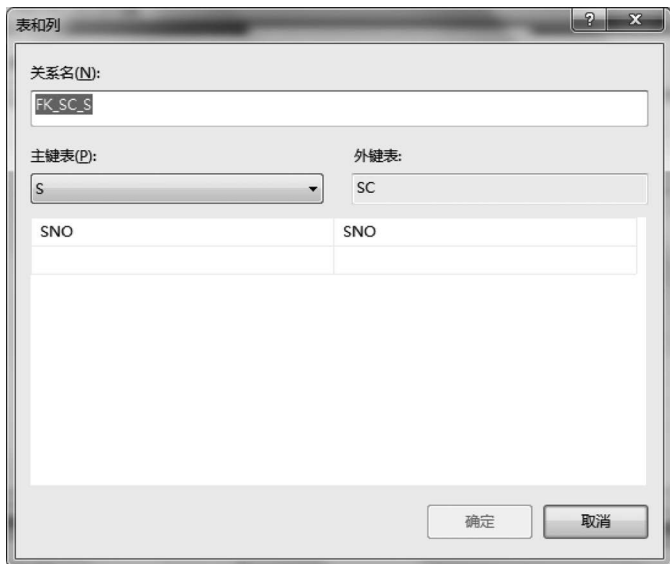


图 5.11 选择外键关系的约束列

④ 设置完成后,单击“确定”按钮返回“外键关系”对话框,检查表和列规范、关系名等属性设置无误后,单击“确定”按钮。即完成外键约束的创建。

在 SQL Server 中,也可以通过“数据库关系图”来建立外键约束,其操作方法与上述操作类似,此处不再赘述。

3) 创建数据库关系

数据库关系是以图形方式显示数据库的结构。使用数据库关系图可以创建和修改表、列、关系和键。此外,还可以修改索引和约束。为使数据库可视化,可创建一个或更多的关系图,以显示数据库中的部分或全部表、列、键和关系。

在 SQL Server 中,创建数据库关系图的方法如下:

在“对象资源管理器”中右击“数据库关系图”文件夹,在快捷菜单中选择“新建数据库关系图”命令,弹出“添加表”对话框,在“表”列表中选择所需的表,然后单击“添加”按钮。这些表将以图形方式显示在新的数据库关系图中,如图 5.12 所示。

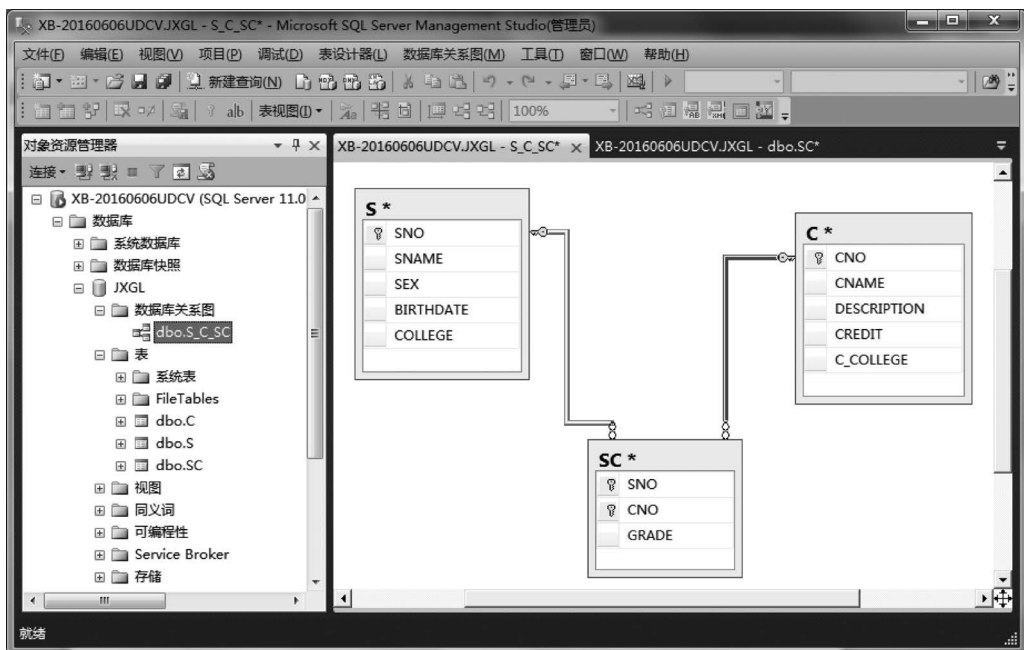


图 5.12 数据库关系图

在数据库关系图中,右击关系图的空白处,通过弹出的快捷菜单,可以新建表或添加数据库中已定义(未出现在关系图中)的表,也可以继续删除表、修改现有表或更改表关系,直到新的数据库关系图创建完成为止。

4) 删除表

当某个表不再使用时,就可以将其删除以释放数据库空间。表被删除后,它的结构定义、数据、全文索引、约束和索引都将永久地从数据库中删除。表上的默认值将被解除绑定,任何与表关联的约束或触发器将自动删除。

利用 SSMS 图形化方式删除表的步骤如下:

展开“对象资源管理器”的文件夹,选择要删除的表并右击,从快捷菜单中选择“删除”命令,弹出如图 5.13 所示的“删除对象”对话框,单击“确定”按钮即可删除表。

要注意,当有对象依赖关系时就不能删除表。单击图 5.13 中的“显示依赖关系”按钮会弹出显示该表所依赖的对象和依赖于该表的对象。

2. 利用 Transact-SQL 语句方式

1) 创建表

利用 CREATE TABLE 语句可以创建数据表,常用语句格式如下:

```
CREATE TABLE <表名>
(
    <列名 1> <数据类型> <列级完整性约束>
    [, ... n]
    <表级完整性约束>
    [, ... n]
)
[ON <filegroup> | <"default">]
```

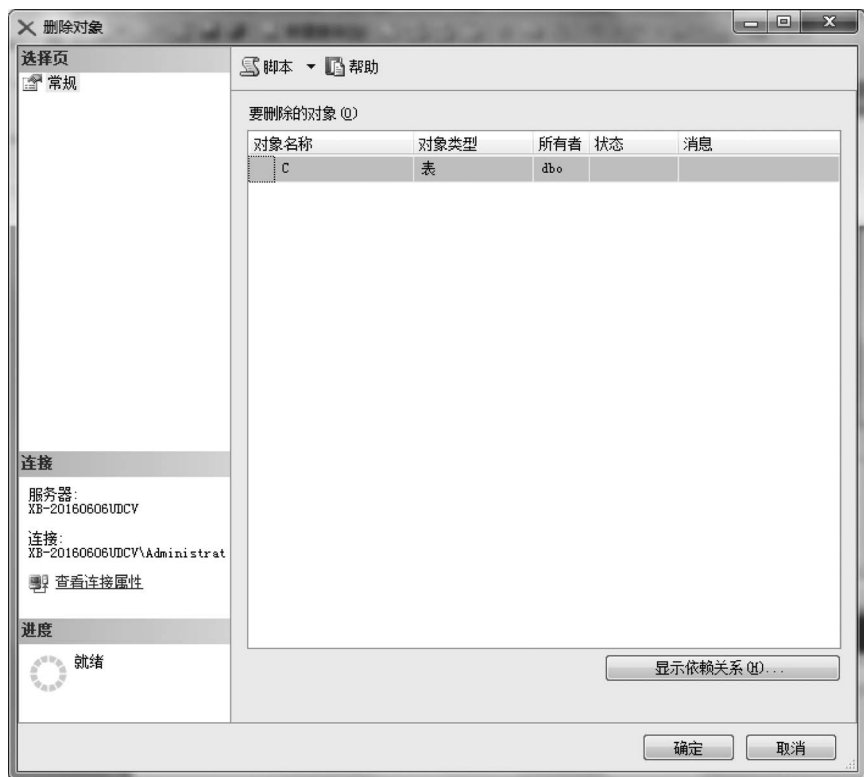


图 5.13 “删除对象”对话框

参数说明如下。

(1) <表名>: 在当前数据库中新建的表名称。

(2) ON <filegroup>|<"default">: 指明存储表的文件组。如果指定了 default, 或根本没有指定 ON, 则表示存储在默认文件组中。

2) 列级完整性约束与表级完整性约束

(1) 列级完整性约束。它是行定义的一部分, 只能应用于一列上。列级完整性约束如下:

① 默认值约束——[CONSTRAINT <默认值约束名>] DEFAULT 常量表达式。

② 空值/非空值约束——NULL/NOT NULL。

③ 主键约束。

[CONSTRAINT <主键约束名>] PRIMARY KEY [CLUSTERED| NONCLUSTERED](<主键列名>)

其中, CLUSTERED|NONCLUSTERED 表示所创建的唯一性约束是聚集索引/非聚集索引, 默认为 CLUSTERED(聚集索引)。

④ 外键约束。

[CONSTRAINT <外键约束名>] [FOREIGN KEY] REFERENCES <父表名>[(<主键列名>)]

⑤ 唯一性约束。

[CONSTRAINT <唯一性约束名>] UNIQUE[CLUSTERED| NONCLUSTERED]

⑥ 检查约束。

[CONSTRAINT <检查约束名>]CHECK(<逻辑表达式>)

(2) 表级完整性约束。它独立于列的定义,可以应用在一个表中的多列上。表级完整性约束如下。

① UNIQUE(列名 1,列名 2,...,列名 n): 多个列名单值约束。

② PRIMARY KEY(列名 1,列名 2,...,列名 n): 多个列名组合主键约束。

③ FOREIGN KEY(外键) REFERENCES 主键表(主键): 多个列名组合外键约束。

④ CHECK(逻辑表达式): 含有多个列名逻辑表达式的检查约束。

如果完整性约束涉及该表的多个属性列,必须定义在表级上,否则既可以定义在列级也可以定义在表级。如 PRIMARY KEY,当只涉及一列时,定义为列级约束;当涉及多列时,则定义为表级约束。

下面通过创建表的例子进一步了解利用 CREATE TABLE 语句创建表的相关选项的含义。

例 5.8 创建例 5.7 中数据库 JXGL 的表 S、表 C 和表 SC。

```
USE JXGL
CREATE TABLE S                                -- 创建表 S
(SNO nchar(9) NOT NULL                        -- 学号字段,非空约束
    CONSTRAINT PK_SNO PRIMARY KEY CLUSTERED   -- 主键约束,列约束
    CHECK(SNO LIKE '201705121[0-9][0-9]'),    -- 检查约束,列约束
SNAME nchar(8) NOT NULL,                      -- 姓名字段,非空约束
SEX nchar(2) NULL,                            -- 性别字段
BIRTHDATE date NULL,                         -- 出生日期字段
COLLEGE nchar(20) NULL                       -- 学院字段
)
USE JXGL
CREATE TABLE C                                -- 创建表 C
(CNO nchar(4) NOT NULL,                       -- 课程号字段,非空约束
CNAME nchar(20) NOT NULL,                    -- 课程名字段,非空约束
DESCRIPTION text NULL,                      -- 课程描述字段
CREDIT real NULL,                           -- 学分字段
C_COLLEGE nchar(20)                          -- 开课学院字段
PRIMARY KEY(CNO)                             -- 主键约束,列约束
)
USE JXGL
CREATE TABLE SC
(SNO nchar(9) NOT NULL,                      -- 学号字段,非空约束
CNO nchar(4) NOT NULL,                      -- 课程号字段,非空约束
GRADE REAL NULL,                           -- 成绩字段
PRIMARY KEY(SNO,CNO),                      -- 主键约束,表约束
FOREIGN KEY(SNO) REFERENCES S(SNO),        -- 外键约束,表约束
FOREIGN KEY(CNO) REFERENCES C(CNO)         -- 外键约束,表约束
)
```

该例中,先定义了参照表 S 和 C,最后定义了表 SC。

3) 更改表结构

利用 ALTER TABLE 语句可以更改原有表的结构,该语句的常用格式如下:

```
ALTER TABLE <表名>
[ ALTER COLUMN <列名> <列定义>]
|[ ADD <列名> <数据类型> <约束>[, ... n]]
|[ DROP COLUMN <列名>[, ... n]]
|[ ADD CONSTRAINT <约束名> <约束>[, ... n]]
|[ DROP CONSTRAINT <约束名>[, ... n]]
```

参数说明如下。

- (1) <表名>: 所要修改的表的名称。
- (2) <列名>: 要修改的字段名。
- (3) ALTER COLUMN: 修改列的 Transact-SQL 子句。
- (4) ADD: 增加新列或约束的 Transact-SQL 子句。
- (5) DROP: 删除列或约束的 Transact-SQL 子句。

注意: 在标准的 SQL 中, 每个 ALTER TABLE 语句中的每个子句只允许使用一次。

例 5.9 在学生表 S 中, 将列 SEX 的原数据长度 2 改为 1。

```
USE JXGL
ALTER TABLE S
ALTER COLUMN SEX CHAR(1) NULL
```

例 5.10 在学生表 S 中, 将 BIRTHDATE 列名改为 AGE, 数据类型为 SMALLINT。

```
USE JXGL
ALTER TABLE S
DROP COLUMN BIRTHDATE
ALTER TABLE S
ADD AGE SMALLINT
```

ALTER COLUMN 子句一次只能更改一个列的属性, 如果需要更改多个列时, 可以多次利用 ALTER TABLE 语句。在本例中, 先删除列 BIRTHDATE, 然后再增加列 AGE。

例 5.11 在学生表 S 中删除列 SNO 的主键约束。

```
USE JXGL
ALTER TABLE S
DROP CONSTRAINT PK_SNO
```

4) 删除表

利用 DROP TABLE 语句可以删除数据表, 该语句的常用格式如下:

```
DROP TABLE <表名>
```

其中, <表名> 为所要删除表的名称。

例 5.12 删除数据库 JXGL 内的学生表 S。

```
USE JXGL
DROP TABLE S
```

5.3.2 表中数据的维护

在数据库中的表对象建立后, 用户对表数据的维护可以归纳为 4 个基本的操作: 添加或插入新数据、查询(Query) 现有数据、更改或更新现有数据和删除现有数据。其中, 查询

操作将在第6章详细讲解。对表中数据维护也有两种方法：一是利用SSMS图形方式；二是利用Transact-SQL语句方式。对于利用SSMS图形方式进行表数据维护时，与前面介绍的利用SSMS图形方式进行创建表等类似，用鼠标右击需要操作的表，在弹出的快捷菜单中选择“编辑前200行”命令，再选择相关操作，即可完成数据插入、修改和删除表中数据的操作。下面重点介绍表中数据维护的Transact-SQL语句方式。

1. 插入表数据

利用INSERT语句可以往原有表中添加数据，常用格式如下：

```
INSERT INTO <表名>[(<列名>[, ... n])]  
VALUES(<常量表达式>|NULL|DEFAULT[, ... n])
```

参数说明如下。

(1) <表名>：要插入数据的表名称。

(2) <列名>：要插入数据所对应的字段名，字段名表的顺序可以与表的列顺序不同。如果向表中的部分列插入数据，则相应的字段名表不能省略；如果向表中所有列插入数据且字段顺序与表结构相同，则字段名可以省略。

(3) <常量表达式>：与列名对应的字段的值，字符型和日期型的值需要用单引号括起来，值与值之间用逗号分隔。

注意：INSERT ... VALUES语句一次只能插入一行数据。

例 5.13 在教学管理数据库JXGL中，向学生表S中插入记录('S13','程晓晴','女','1996-10-11','CS')。

```
USE JXGL  
INSERT INTO S(SNO, SNAME, SEX, AGE, SDEPT)  
VALUES('S13', '程晓晴', '女', '1996-10-11', 'CS')
```

或

```
USE JXGL  
INSERT INTO S  
VALUES('S13', '程晓晴', '女', 21, 'CS')
```

2. 修改表数据

利用UPDATE语句可以更改原有表的数据，该语句的常用格式如下：

```
UPDATE <表名>  
SET <列名> = <表达式>[, ... n]  
[WHERE <逻辑表达式>]
```

参数说明如下。

(1) <表名>：要修改数据的表名称。

(2) <列名>：要修改数据所对应的字段名。

(3) <表达式>：要修改的新值。如果新值违反了约束或与修改列的数据类型不兼容，则取消该语句，并返回错误提示。

(4) <逻辑表达式>：更新条件，只有满足条件的记录才会被更新，如果不设置，则更新所有的记录。

例 5.14 在教学管理数据库JXGL中，把学生表S中学号为S2的学生姓名改为“王中

桥”、出生日期改为 1997-06-06。

```
USE JXGL
UPDATE S
SET SNAME = '王中桥', BIRTHDATE = '1997 - 06 - 06'
WHERE SNO = 'S2'
```

3. 删除表数据

Transact-SQL 支持两种删除现有表中数据的语句,分别是 DELETE 语句和 TRUNCATE TABLE 语句。

1) DELETE

利用 DELETE 语句可以删除原有表或视图中的一条或多条记录,每一条记录的删除都将被记入事务日志文件中。该语句的常用格式如下:

```
DELETE FROM <表名>|<视图名>
[WHERE <逻辑表达式>]
```

参数说明如下。

(1) <表名>|<视图名>: 要删除数据的表或视图名称。

(2) <逻辑表达式>: 删除条件,只有满足条件的记录才会被删除,如果不设置此选项,则删除所有记录。

使用 DELETE 语句可以从表中删除一条或多条记录。如果有关联表存在,那么在删除表数据时,应当首先删除外键表中的相关记录,然后才能删除主键表中的记录。

例 5.15 在教学管理数据库 JXGL 中,删除学生表 S 中姓名为“张丽”的学生记录。

```
USE JXGL
DELETE S
WHERE SNAME = '张丽'
```

2) TRUNCATE TABLE

TRUNCATE TABLE 语句一次会删除指定表中的所有记录,同时不会把每条记录的删除操作记入事务日志文件。所以 TRUNCATE TABLE 语句是一种快速清空表的方法。语句格式如下:

```
TRUNCATE TABLE <表名>
```

其中,<表名>是要清空表的名称。

例 5.16 使用 TRUNCATE TABLE 清空教学管理数据库 JXGL 的表 C。

```
TRUNCATE TABLE C
```

3) DELETE 和 TRUNCATE TABLE 语句的区别

TRUNCATE TABLE 在功能上与不带 WHERE 子句的 DELETE 语句相同,两者均删除表中的全部记录数据,但有如下几个不同点:

(1) DELETE 语句每次删除一行,并在事务日志中进行一次记录,而 TRUNCATE TABLE 通过释放存储表数据所用的数据页来删除数据,并且在事务日志中只记录页的释放。所以 TRUNCATE TABLE 比 DELETE 速度快,且使用的系统和事务日志资源少。

(2) TRUNCATE TABLE 删除表中的所有行,但表结构及其列、约束、索引等保持不变。

(3) DELETE 删除表数据后,标识字段不能复用。也就是说,如果把 id=10(假如 id 是

标识字段)的那条记录删除了,就不可能再插入一条记录让 $id=10$; 而 TRUNCATE 删除表数据后,标识重新恢复初始状态。默认为初始值为 1,也就是说,TRUNCATE 之后,再插入一条数据使得 $id=1$ 。

(4) 对于被外键约束所引用的表,不能使用 TRUNCATE TABLE,而应使用不带 WHERE 子句的 DELETE 语句。

(5) TRUNCATE TABLE 语句不能激活触发器。

习 题 5

扫一扫



习题

扫一扫



自测题