第3章 数据表的创建和管理

学习目标

掌握 SQL Server 2022 中的各种数据类型；掌握建立数据表的方法；掌握查看、修改和删除数据表的方法；掌握使用主键约束和唯一约束保证数据表的完整性；掌握使用检查约束、默认值约束保证列的完整性；掌握使用外键约束保证数据表之间的完整性；掌握约束禁用和启用的方法；掌握表中数据的添加、修改和删除的方法。

观看视频

3.1 数据表概述

3.1.1 表的基本概念

数据库中的表是组织和管理数据的基本单位，数据库的数据保存在表中，数据库的管理和开放都依赖于表。表的特性如下。

（1）表是组织和管理数据的基本单位。

（2）表是由行和列组成的二维结构。

（3）表中的一行称为一条记录，表中的一列称为一个字段。

将“学生选课”数据库恢复到初始状态，打开 student 表，如图 3-1 所示。

Sno	Sname	Ssex	Sage	sdept
95001	刘超华	男	22	计算机系
95002	刘晨	女	21	信息系
95003	王敏	女	20	数学系
95004	张海	男	23	数学系
95005	陈平	男	21	数学系
95006	陈斌斌	男	28	数学系
95007	刘德虎	男	24	数学系
95008	刘宝祥	男	22	计算机系
95009	吕翠花	女	26	计算机系
95010	马盛	男	23	数学系
95011	吴霞	男	22	计算机系
95012	马伟	男	22	数学系
95013	陈冬	男	18	信息系
95014	李小鹏	男	22	计算机系
95015	王娜	女	23	信息系
95016	胡萌	女	23	计算机系
95017	徐晓兰	女	21	计算机系
95018	牛川	男	22	信息系
95019	孙晓慧	女	23	信息系

图 3-1 “学生选课”数据库的 student 表

说明：每个 SQL Server 数据库可容纳多达 20 亿个表，每个表中至多可以有 1024 列，每一行最多允许有 8086 字节。

3.1.2　SQL Server 2022 的数据类型

在 SQL Server 2022 中，每个列、局部变量、表达式和参数都具有一个相关的数据类型。不同的数据类型能保存不同的数据。常用的数据类型有数字数据、字符数据、货币数据、日期和时间数据、二进制字符串等。

1. 数据类型——数字型

SQL Server 2022 支持的数字型数据类型如表 3-1 所示。

表 3-1　SQL Server 2022 支持的数字型数据类型

数 据 类 型	说　　明
bigint	−2^63(−1.8E+19)～ 2^63−1 (1.8E+19)的整型数
int	−2^31 (−2 147 483 648)～ 2^31−1 (2 147 483 647)的整型数
smallint	−2^15 (−32 768)～ 2^15−1 (32 767)的整型数
tinyint	0～255 的整型数
float	浮点型数据，−1.79E +308～1.79E+308
real	浮点精度数字数据，−3.40E+38～3.40E+38
bit	整型数据，值为 1、0
Decimal(p,s)和 numeric(p, s)	固定精度和小数的数字数据，取值范围为 $-10^{38}+1$～$10^{38}-1$。p 变量指定精度，取值范围为 1～38。s 变量指定小数位数，取值范围为 0～p

2. 数据类型——货币型

SQL Server 2022 支持的货币型数型如表 3-2 所示。

表 3-2　SQL Server 2022 支持的货币数据类型

数 据 类 型	范　　围
money	−922 337 203 685 477 5808～922 337 203 685 477.5807
smallmoney	−214 748.3648～214 748.3647

3. 数据类型——日期和时间型

SQL Server 2022 支持的日期和时间型数据类型如表 3-3 所示。

表 3-3　SQL Server 2022 支持的日期和时间型数据类型

数 据 类 型	范　　围	精　确　度
date	0001-01-01～9999-12-31	1 天
time	00:00:00.0000000～23:59:59.9999999	100ns
smalldatetime	1900-01-01～2079-06-06	1min
datetime	1753-01-01～9999-12-31	0.003 33s
datetime2	0001-01-01 00:00:00.0000000～9999-12-31 23:59:59.9999999	100ns
datetimeoffset	0001-01-01 00:00:00.0000000～9999-12-31 23:59:59.9999999(以 UTC 时间表示)	100ns

4. 数据类型——字符型

SQL Server 2022 支持的字符型数据类型如表 3-4 所示。

表 3-4 SQL Server 2022 支持的字符型数据类型

数据类型	说明
char [(n)]	固定长度的字符数据，长度为 n 字节，n 的取值范围为 1 ～8000
varchar [(n)]	可变长度的字符数据，长度为 n 字节，n 的取值范围为 1～8000
nchar [(n)]	固定长度的 Unicode 字符数据，n 的取值范围为 1～4000
nvarchar [(n)]	可变长度的 Unicode 字符数据，n 的取值范围为 1～4000
text	变长度字符数据，最多达到 2 147 483 647 个字符
ntext	变长度的 Unicode 字符数据，最多可达 1 073 741 823 个字符

说明：

(1) n 的默认值均为 1。

(2) 对于一个 char 类型字段，不论用户输入的字符串有多长(不大于 n)，其长度均为 n 字节。当输入字符串长度大于 n 时，系统自动截取 n 个长度的字符串；而变长字符型 varchar(n)的长度为输入字符串的实际长度，而不一定是 n。

5. 数据类型——二进制和图像型

SQL Server 2022 支持的二进制和图像型数据类型如表 3-5 所示。

表 3-5 SQL Server 2022 支持的二进制和图像型数据类型

数据类型	说明
binary [(n)]	固定长度二进制数据，n 的取值范围为 1～8000
varbinary [(n)]	可变长度二进制数据，n 的取值范围为 1～8000
image	可变长度二进制数据，最长为 2 147 483 647 字节

6. 数据类型——其他数据类型

SQL Server 2022 支持的其他数据类型如表 3-6 所示。

表 3-6 SQL Server 2022 支持的其他数据类型

数据类型	说明
uniqueidentifier	存储 16 字节的二进制值
timestamp	当插入或者修改行时，自动生成的唯一的二进制数字的数据类型
CURSOR	允许在存储过程中创建游标变量，游标允许一次一行地处理数据，这个数据类型不能用作表中的列数据类型
sql_variant	可包含除 text、ntext、image 和 timestamp 之外的其他任何数据类型
table	一种特殊的数据类型，用于存储结果集，以进行后续处理
xml	存储 XML 数据的数据类型。可以在列中或者 xml 类型的变量中存储 XML 实例

3.1.3 列的属性

设计数据表，实际上就是设计列的属性，如名称、数据类型、可否为空和数据长度等。

1. 列的为空性

列的为空性是指没有输入的值，或输入的值未知或未定义。列值可以接受空值 NULL，也可以拒绝空值 NOT NULL。

NULL 是一个特殊值，它不同于空字符或者 0。空字符和 0 是有效的字符或数字。如图 3-2 所示，NULL 表示这些课程的先行课未知或者不确定。

2. IDENTITY 属性

IDENTITY 属性可以使表的列包含系统自动生成的数字，可以唯一地标识表的每一

	Cno	Cname	Cpno	Credit	Semester
▶	1	数据库	5	5	4
	10	c++	NULL	3	4
	11	网络编程	NULL	2	5
	2	高等数学	NULL	1	1
	3	信息系统	1	1	3
	4	操作系统	6	1	2
	5	数据结构	7	1	3
	6	数据处理	NULL	1	2
	7	C语言	6	3	1
	8	Java	NULL	3	3
	9	网页制作	NULL	2	5
*	NULL	NULL	NULL	NULL	NULL

图 3-2　NULL 的意思

行，即表中数据列上每行的数字均不相同。IDENTITY 属性的语法格式如下：

```
IDENTITY [(s, i)]
```

其中，s(seed)表示起始值；i(increment)表示增量，其默认值都为 1。

只有整数数据类型的数据列可用于标识列，一个表只能有一个标识列。使用该属性可以指定初始值和增量，但不能更新列。

插入数据到含有 IDENTITY 列的表中时，初始值 s 在插入第一行数据时使用，以后就由 SQL Server 2022 根据上一次使用的 IDENTITY 值加上增量 i 得到新的 IDENTITY 值。

3.2　创建“学生选课”数据库的数据表

3.2.1　使用 Management Studio

【例 3.1】　在“学生选课”数据库中，利用 Management Studio 创建学生表。

分析：使用 Management Studio 创建数据表，即利用 Management Studio 中的表设计器创建表的结构。表设计器是 SQL Server 2022 提供的可视化创建表的工具，主要部分是列管理。用户可以使用表设计器完成对表中所包含列的管理工作，包括创建列、删除列、修改数据类型、设置主键和索引等。

具体操作步骤如下。

(1) 启动 Management Studio。

(2) 在“对象资源管理器”窗格中，展开“数据库”→“学生选课”选项，右击“表”选项，在弹出的快捷菜单中选择“新建”→“表”命令，打开表设计器。

(3) 在表设计器中，在“列名”栏输入字段名 sno，在同一行的“数据类型”栏中设置该字段的数据类型为 char(5)，并在“允许空”栏设置是否允许该字段为空值。如果允许，则选中该复选框；如果不允许，则取消选中该复选框。在学生表中，学号是学生的标识，不能为空，即取消选中该复选框。

(4) 重复步骤(3)，设置 sname 列、ssex 列、Ssage 列和 sdept 列。

(5) 选择“文件”→“保存”命令或单击工具栏上的“保存”按钮，在弹出的对话框中输入表名“student”，新表的相关信息即会出现在对象资源管理器中。

说明：表中各字段属性如图 3-5 所示。

3.2.2 使用 CREATE TABLE 语句

使用 CREATE TABLE 语句创建数据库的语法格式如下：

```
CREATE TABLE <表名>
 ( <列名> <数据类型>
 [ NULL | NOT NULL ] [ IDENTITY [( seed ,increment )] [{<列约束>}]
 [,…n]
)
```

参数说明如下。

NULL | NOT NULL：指定列的为空性，默认值为 NULL。

IDENTITY(seed,increment)：指定为标识列，seed 为初始值，increment 为增量。

【例 3.2】 在“学生选课”数据库中，利用 CREATE TABLE 语句创建课程表和选课表。

```
USE  学生选课
GO
CREATE TABLE  course(                    -- 创建课程表
  cno        char(6) NOT NULL,
  cname     char(20) NOT NULL,
  credit    tinyint,
  semester tinyint)
GO
CREATE TABLE  SC(                        -- 创建选课表
  sno     char(5) NOT NULL,
  cno     char(6) NOT NULL,
  grade   tinyint)
GO
```

说明：在此创建的课程表和选课表，没有创建主键约束，不符合数据库设计要求，在后续的内容中将重新创建带主键约束的数据表。

3.3 管理“学生选课”数据库的数据表

3.3.1 查看表结构

1. 查看数据表的属性

【例 3.3】 利用 Management Studio 查看课程表 course 的属性信息和学期 semester 列的属性信息。

分析：利用 Management Studio 可以以图形方式查看数据表的结构。

具体操作步骤如下。

(1) 在“对象资源管理器”窗格中，展开“数据库”→“学生选课”→“表”选项。

(2) 右击 course 表，在弹出的快捷菜单中选择“属性”命令，打开“表属性-course”界面，如图 3-3 所示。在该界面中可查看表的创建日期、对表拥有权限的用户及权限、数据空间大小、所属文件组以及扩展属性等。

(3) 展开 course→“列”选项，右击 semester 列，在弹出的快捷菜单中选择“属性”命令，打开“列属性-semester”界面，如图 3-4 所示。在该界面中可查看该列的数据类型、是否为主键、是否允许空等属性。

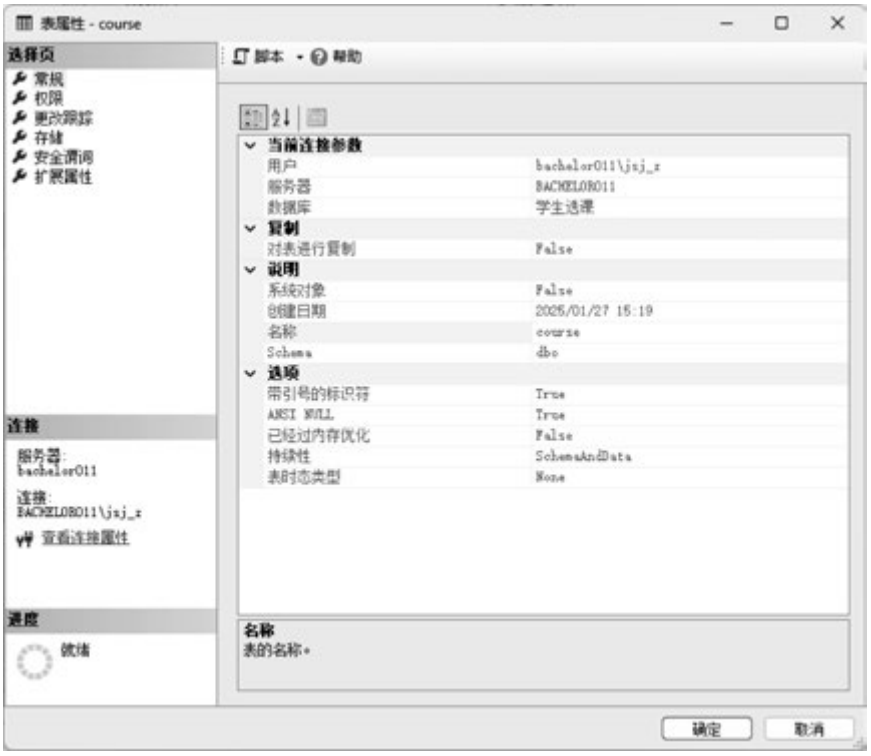

图 3-3　查看课程表 course 的属性

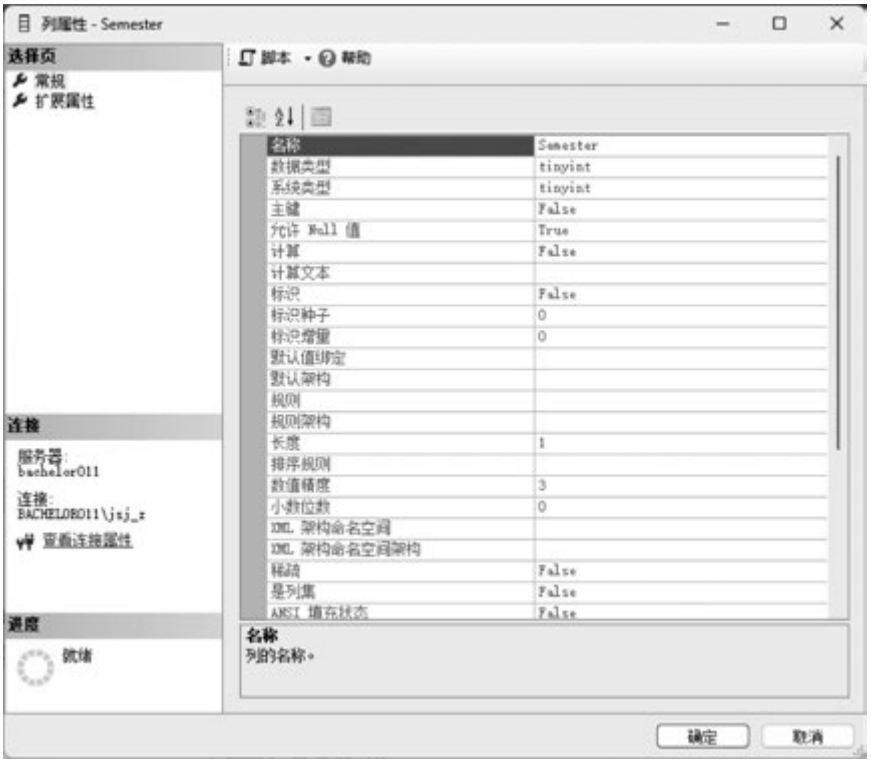

图 3-4　查看课程表 course 中 semester 列的属性

2. 查看表结构

【例 3.4】 查看学生表 student 的表结构、约束、触发器等信息。

具体的操作步骤是：展开 dbo.student 数据库中的“列”“键”“约束”“触发器”“索引”等对象，即可看到相关信息，如图 3-5 所示。

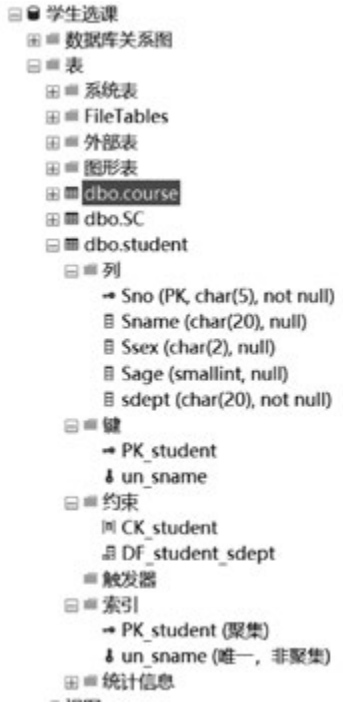

图 3-5 表结构和对象信息

3. 查看表中数据

【例 3.5】 查看课程表 course 中的记录。

在 Management Studio 中，右击 course 表，在弹出的快捷菜单中选择“编辑前 200 行”命令，即会显示该表中的所有数据。在该界面中可以查询、编辑表中的数据。

3.3.2 修改数据表

在创建数据表之后，随着系统应用及用户需求的改变，可能需要修改数据表的相关属性，如增加新的字段、删除字段、修改字段类型、修改主键或索引等。数据表的修改可以在表设计器中完成，也可以通过 SQL 语句在查询编辑器中完成。

1. 使用 Management Studio

【例 3.6】 查看学生表 student，并在系别名称 sdept 列之后，增加“籍贯”列，数据类型为 char(20)。

分析：可在 Management Studio 中修改学生表。

具体操作步骤如下。

(1) 在 Management Studio 中，展开“数据库”→“学生选课”→“表”选项。

(2) 右击 student 表，在弹出的快捷菜单中选择“设计”命令，打开表设计器。

（3）将光标定位到 sdept 列。

（4）右击并在弹出的快捷菜单中选择“插入列”命令，如图 3-6 所示；然后在“列名”栏输入“籍贯”，“数据类型”设为 char(20)。

图 3-6　增加新列

（5）单击工具栏上的“保存”按钮，保存对表结构的修改。

说明：在表设计器中，可以修改列名、列的数据类型、允许空等属性，也可以添加、删除列，还可以指定表的主键约束。

2. 使用 ALTER TABLE 语句

使用 ALTER TABLE 语句可以添加或者删除表约束，也可以禁用或者启用已存在的约束或触发器，ALTER TABLE 语句功能强大，下面逐一介绍。

（1）添加列。其语法格式如下：

```
ALTER TABLE <表名>
ADD <列定义>[,…n]
```

【例 3.7】 在学生表中添加两列：“籍贯”列，数据类型为 char(20)，允许空；“宿舍区”列，数据类型为 char(20)，允许空。

```
ALTER TABLE student
ADD
籍贯 char(20)  NULL,
宿舍区 char(20)  NULL
```

（2）删除列。其语法结构如下：

```
ALTER TABLE <表名>
DROP COLUMN <列名>[,…n]
```

【例 3.8】 在学生表中删除两列：“籍贯”列和“宿舍区”列。

```
ALTER TABLE student
DROP COLUMN 籍贯,宿舍区
```

（3）修改列的定义。

【例 3.9】 在学生表中将所在系 sdept 列的数据类型修改为 varchar(20)。

```
ALTER TABLE student
ALTER COLUMN sdept varchar(20)
```

说明：在修改列的定义时，如果修改后的长度小于原来定义的长度，或者数据类型的更改可能导致数据被更改，则降低列的精度或减少小数位数可能导致数据被截断。

（4）修改列名。

【例 3.10】 在学生表中将 sdept 列重命名为“系别”。

```
Sp_rename 'student.sdept ','系别'
```

说明：进行本例测试后，请将学生表恢复原状，恢复代码为：Sp_rename 'student.系别', 'sdept'。

3.3.3 删除数据表

1. 使用 Management Studio

【例 3.11】 删除学生表。

（1）在“对象资源管理器”窗格中，展开“数据库”→“学生选课”→“表”选项。

（2）右击学生表，在弹出的快捷菜单中选择“删除”命令。

（3）在打开的“删除对象”界面中，单击“确定”按钮，完成删除任务，如图 3-7 所示。

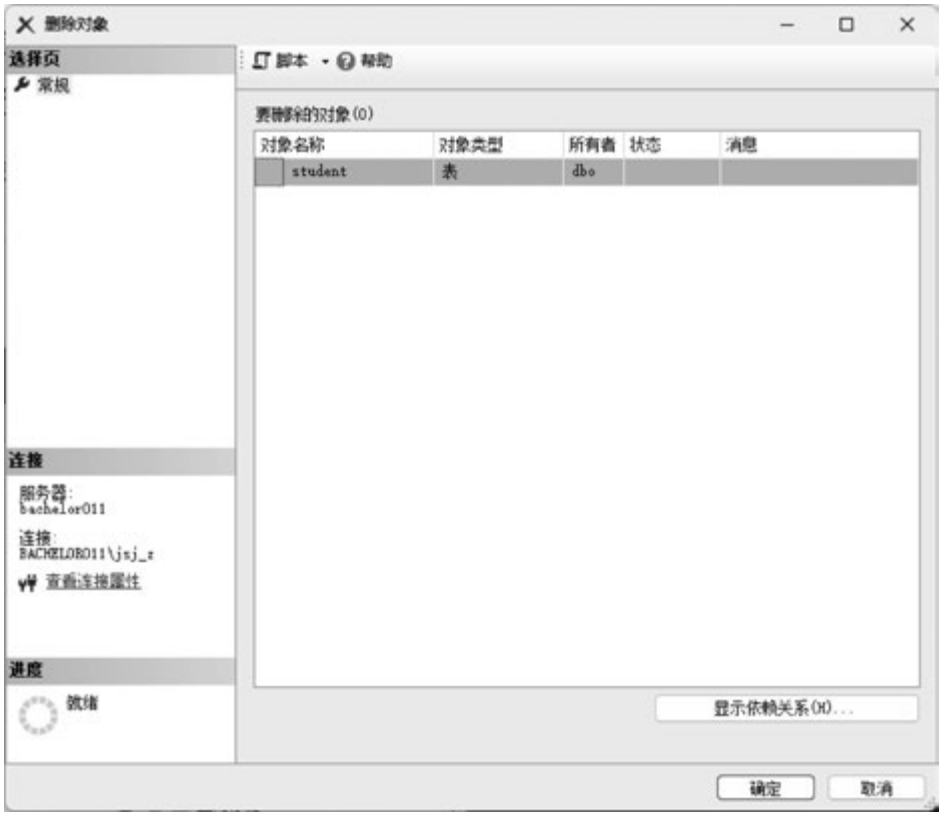

图 3-7 删除数据表

2. 使用 DROP TABLE 语句

使用 DROP TABLE 语句的语法格式如下：

```
DROP TABLE <表名>
```

【例 3.12】 删除课程表。

在查询编辑器中执行如下 SQL 语句：

```
USE 学生选课
GO
DROP TABLE course
GO
```

说明：为保持数据库的延续性，练习完例 3.11 和例 3.12 后请还原数据库。

3.3.4 重命名数据表

1. 使用 Management Studio

使用 Management Studio 修改表名的方法是：在指定数据库中展开表，右击指定表，在弹出的快捷菜单中选择“重命名”命令，输入新表名即可。

2. 使用系统存储过程 sp_rename

使用系统存储过程 sp_rename 修改表名的语法格式如下：

```
sp_rename '原表名','新表名'
```

【例 3.13】 将学生表 student 更名为 Newstudent。

```
USE 学生选课
GO
sp_rename 'student','Newstudent'
GO
```

说明：更改列名时必须加引号，更改表名时可加引号，也可不加，测试完后请恢复表名。

3.4 学生选课数据库数据的完整性


观看视频

数据完整性是指数据的精确性和可靠性，主要用于保证数据库中数据的质量。它是为防止数据库中存在不符合语义规定的数据和防止因错误信息的输入输出造成无效操作或报错而提出的。

例如，如果输入了学号值为 95001 的学生，则在该数据库中不应允许其他学生使用具有相同值的学号。如果将学生性别列的取值范围设置为“男”或“女”，则对于该列，数据库不应接收其他信息。如果课程表中存储了课程编号，则学生选课时，只能选择课程表中存在的课程编号。

3.4.1 数据完整性的分类

数据完整性分为 3 类：实体完整性(Entity Integrity)、用户定义完整性(User-defined Integrity)、参照完整性(Referential Integrity)，如图 3-8 所示。

1. 实体完整性

实体完整性用于保证表中的每一行数据在表中是唯一的。

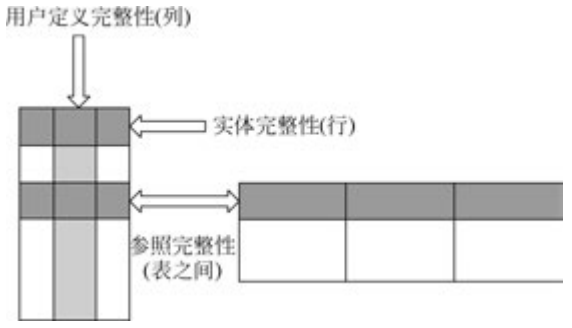

图 3-8　数据完整性

2. 用户定义完整性

用户定义完整性也称为域完整性或语义完整性，是指数据库应用系统根据应用环境的不同，需要的一些约束条件，如表中的列的数据类型、限制格式或限制可能值的范围。它反映某一具体应用的数据必须满足应用语义的要求。

3. 参照完整性

参照完整性是指在输入或删除记录时，包含主关键字的主表和包含外关键字的从表的数据应对应一致，保证了表之间数据的一致性，防止数据丢失或无意义的数据在数据库中扩散。SQL Server 将防止用户执行下列操作。

（1）当主表中没有关联的记录时，将记录添加或更改到相关表中。

（2）更改主表中的值，但会导致在相关表中生成孤立的记录。

（3）从主表中删除记录，但仍存在与该记录匹配的相关记录。

例如，对于学生选课数据库中的选课表（SC）和课程表（course），参照完整性基于 sc 表中的课程编号（cno）和 course 表中的课程编号（cno）的关联关系，即选修的课程参照课程表中的课程，如图 3-9 所示。

Sno	Cno	Grade
95001	1	87
95001	2	76
95001	3	79
95001	4	80
95001	5	81
95001	6	82
95001	7	67
95002	1	89
95002	2	81
95004	1	83
95004	2	56
95005	1	89
95006	1	54
95006	2	77
95010	1	56
95013	3	80
95013	5	90
95014	2	NULL
95015	2	NULL

Cno	Cname	Cpno	Credit	Semester
1	数据库	5	5	4
10	C++	NULL	3	4
11	网络编程	NULL	2	5
2	高等数学	NULL	1	1
3	信息系统	1	1	3
4	操作系统	6	1	2
5	数据结构	7	1	3
6	数据处理	NULL	1	2
7	C语言	6	3	1
8	Java	NULL	3	3
9	网页制作	NULL	2	5

图 3-9　选课表和课程表之间的参照关系

3.4.2　约束概述

1. 约束定义

约束(Constraint)是 Microsoft SQL Server 2022 提供的自动保持数据库完整性的一种方法。约束就是限制,定义约束就是定义可输入表或表的单个列中的数据的限制条件。

2. 约束分类

在 SQL Server 中有 5 种约束:主键约束(Primary Key Constraint)、外键约束(Foreign Key Constraint)、唯一约束(Unique Constraint)、检查约束(Check Constraint)、默认值约束(Default Constraint)。

约束与完整性之间的关系如表 3-7 所示。

表 3-7　约束与完整性之间的关系

完整性类型	约束类型	描　　述	约束对象
实体完整性	Primary Key	每行记录的唯一标识符,确保用户不能输入重复值,并自动创建索引,提高性能,该列不允许使用空值	行
参照完整性	Foreign Key	定义一列或几列,其值与本表或其他表的主键或 UNIQUE 列相匹配	表与表之间
用户定义完整性	Unique	在列集内强制执行值的唯一性,防止出现重复值,表中不允许有同一列或多列包含相同两行非空值	列
	Check	指定某一列可接收的值	
	Default	当使用 INSERT 语句插入数据时,若已定义默认值的列没有提供指定值,则将该默认值插入记录中	

3.4.3　主键约束

主键约束用于指定表的一列或几列的组合唯一标识表,即能在表中唯一地指定一行记录,这样的一列或列的组合称为表的主键。定义主键约束的列,其值不可为空、不可重复;每个表中只能有一个主键。主键约束也称实体完整性约束。

1. 使用 Management Studio 创建主键约束

【例 3.14】　在“学生选课”数据库中创建主键约束。

分析:“学生选课”数据库目前有 3 张表,根据主键约束的定义,可以确定,学生表的主键是(sno),课程表的主键是(cno),选课表的主键是(sno,cno),这里可以通过 Management Studio 为选课表创建主键约束。

具体操作步骤如下。

(1) 启动 Management Studio。

(2) 在“对象资源管理器”窗格中,展开“数据库”→“学生选课”→“表”选项,右击选课表 sc,在弹出的快捷菜单中选择“设计”命令,打开表设计器。

(3) 将光标定位到 sno 行,同时按住 Ctrl 键,单击 cno 行。

(4) 单击 Management Studio 工具栏上的 按钮,设置主键。此时 sno 和 cno 行会显示钥匙图标,如图 3-10 所示。

(5) 选择“文件”→“保存”命令或单击工具栏上的 按钮。

(6) 仿照步骤(1)～(5),为其他表设置主键。

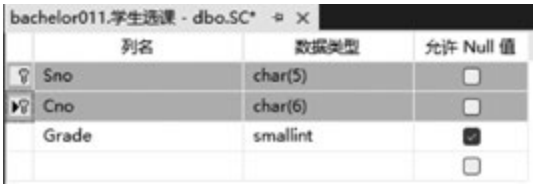

图 3-10　设置主键

2. 在创建表的同时创建主键约束

（1）创建单个列的主键可采用列级约束，它的语法格式如下：

```
CREATE TABLE<表名>
(<列名><列属性>[CONSTRAINT 约束名]
PRIMARY KEY [CLUSTERED | NONCLUSTERED])
```

（2）创建多个列组合的主键约束可采用表级约束，其语法格式如下：

```
CONSTRAINT<约束名>
PRIMARY KEY [CLUSTERED | NONCLUSTERED](列名 1[,…列名 16])
```

其中，约束名在数据库中必须是唯一的；CLUSTERED | NONCLUSTERED 表示在创建主键时自动创建的索引类别，CLUSTERED 为默认值；主关键字最多由 16 个列组成。

【例 3.15】 在“学生选课”数据库中，创建课程表 course 的同时创建主键。

当课程表不存在时，可在查询编辑器中执行如下 Transact-SQL 语句：

```
USE  学生选课
GO
CREATE TABLE  course(
  cno       char(6) NOT NULL  primary key,
  cname     char(20) NOT NULL,
  credit    tinyint,
  semester  tinyint)
```

说明：在此处创建的约束包含在列的定义中，不用指定约束名，系统自动分配名称。这类约束称为列级完整性。

Sno	Cno	Grade
95001	1	87
95001	2	76
95001	3	79
95001	4	80
95001	5	81
95001	6	82
95001	7	67
95002	1	89
95002	2	81
95004	1	83
95004	2	56
95005	1	89
95006	1	54
95006	2	77
95010	1	56
95013	3	80
95013	5	90
95014	2	NULL
95015	2	NULL

图 3-11　选课表

【例 3.16】 在“学生选课”数据库中，创建如图 3-11 所示的选课表，并设置主键。

分析：在例 3.2 中创建表时没有创建表中的主键约束，在此重新建表。在主键只有一列的情况下，可以采用表级约束或列级约束；若主键包含两列及以上，则必须采用表级约束，即在所有的列定义后再定义约束。若创建一个名称为 PK_sc 的主键约束，则代码为“CONSTRAINT PK_sc PRIMARY KEY (sno,cno)”，放在所有列定义之后。

在查询编辑器中执行如下 SQL 语句：

```
CREATE TABLE  sc
(sno     CHAR(5) NOT NULL,
cno     CHAR(6) NOT NULL,
```

```
grade  TINYINT
CONSTRAINT Pk_sc PRIMARY KEY(sno,cno))
GO
```

说明：采用表约束时，最好指明约束名称，表级约束与列定义相互独立。

3. 在一张现有表上添加主键约束

1）使用 Management Studio

在 Management Studio 中，右击要添加约束的表，在弹出的快捷菜单中选择“修改”命令，利用表设计器添加约束。

2）利用 ALTER TABLE 语句

使用 ALTER TABLE 语句不仅可以修改列的定义，而且可以添加和删除约束。语法格式如下：

```
ALTER TABLE <表名>
ADD CONSTRAINT 约束名 PRIMARY KEY(列名,[,…n])
```

例如：如果 sc 表创建时没有添加主键，则可以执行如下语句添加：

```
ALTER TABLE sc
ADD CONSTRAINT Pk_Snocno PRIMARY KEY (sno,cno)
```

3.4.4　外键约束

两个表中如果有共同列，则可以利用外关键字与主关键字将两个表关联起来。例如，学生表和选课表可以通过它们的共同列 sno 关联起来，在学生表中将 sno 列定义为主关键字，在选课表中通过定义 sno 列为外关键字将选课表和学生表关联起来。当向含有外关键字的选课表中插入数据时，如果选课表的 sno 列中插入的列值在 student 表的 sno 列中不存在，则系统会拒绝插入数据。外键约束也称参照完整性约束。

1. 使用 Management Studio 创建外键约束

【例 3.17】 在学生选课数据库的选课表中创建外键约束。

分析：在选课表的结构中，有主键(sno,cno)列，有外键 sno 列和学生表的 sno 列对应，有外键 cno 列与课程表的 cno 列对应，可以使用 Management Studio 来实现外键约束。

具体操作步骤如下。

（1）启动 Management Studio，在“对象资源管理器”窗格中展开“数据库”→“学生选课”→“表”选项。

（2）右击选课表 sc，在弹出的快捷菜单中选择“设计”命令，打开表设计器。

（3）将光标定位到 sno 行并右击，在弹出的快捷菜单中选择“关系”命令或单击工具栏上的 按钮，如图 3-12 所示。

（4）在弹出的“外键关系”对话框中，单击左下角的“添加”按钮后，在右侧单击“表和列规范”右端的按钮，如图 3-13 所示。

图 3-12　设置外键约束 1

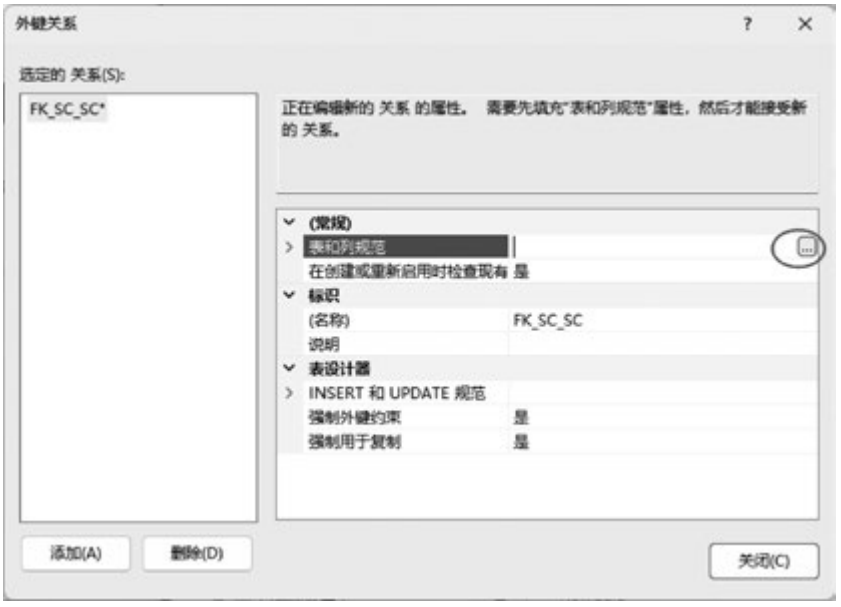

图 3-13　设置外键约束 2

（5）打开“表和列”对话框，选择 student 表作为主键表，其主键为 sno，系统默认选择 sc 表作为外键表，把多余的键去掉，选择 sc 中的 sno 列作为外键，单击“确定”按钮，如图 3-14 所示。

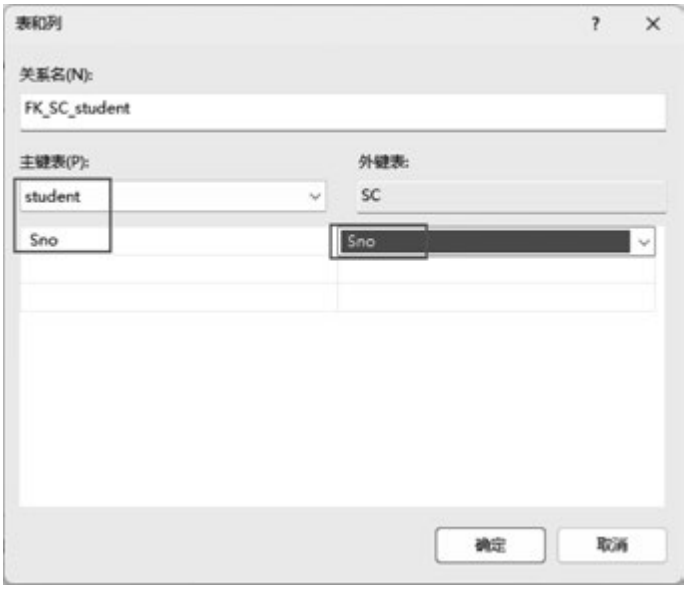

图 3-14　设置外键约束 3

（6）单击工具栏上的“保存”按钮，在弹出的对话框中，单击“是”按钮，如图 3-15 所示。

说明：若设置无法保存，请单击“工具”选项卡，在弹出的下拉菜单中单击“选项”命令，并在“设计器”→“表选项”中取消勾选“阻止保存要求重新创建表的更改”复选框，如图 3-16 所示。

图 3-15　保存外键

图 3-16　设置“阻止保存要求重新创建表的更改”

2. 使用 Transact-SQL 语句定义外键

(1) 创建表时,在定义列的同时定义外键,其语法格式如下:

```
CREATE TABLE<表名>
(列名 数据类型 为空性
  FOREIGN KEY REFERENCES ref_table(ref_column)
  )
```

参数说明如下。

REFERENCES：参照。

ref_table：主键表名，要建立关联的被参照表的名称。

ref_column；主键列名。

【例 3.18】 在“学生选课”数据库中，重新创建选课表。

分析：选课表中的 sno 列参照学生表的 sno 列，选课表中的 cno 列参照课程表的 cno 列。

在查询编辑器中执行如下 Transact-SQL 语句：

```
USE 学生选课
GO
CREATE TABLE  sc
(sno  CHAR(5) NOT NULL FOREIGN KEY References student(sno),
cno  CHAR(6) NOT NULL FOREIGN KEY References course(cno),
grade  TINYINT,
PRIMARY KEY(sno,cno)
)
GO
```

（2）创建表时，定义与列定义无关的表级外键约束，其语法格式如下：

```
CONSTRAINT 约束名
    FOREIGN KEY column_name1[,column_name2,…,column_name16]
    REFERENCES ref_table[ref_column1[,ref_column2,…,ref_column16]]
```

参数说明如下。

Column_name：外键列名。

REFERENCES：参照。

ref_table：主键表名，要建立关联的被参照表的名称。

ref_column：主键列名。

【例 3.19】 利用表级约束形式创建例 3.18 中的选课表。

在查询编辑器中执行如下 Transact-SQL 语句：

```
USE 学生选课
GO
CREATE TABLE  sc
(sno  Char(5) NOT NULL,
cno  Char(6) NOT NULL,
grade  TINYINT,
PRIMARY KEY(sno,cno),
FOREIGN KEY (sno) REFERENCES student(sno),
FOREIGN KEY (cno) REFERENCES course(cno)
)
GO
```

3.4.5 唯一约束

唯一约束用于指定非主键的一个列或多个列的组合值具有唯一性，以防止在列中输入重复的值。也就是说，如果一个数据表已经设置了主键约束，但该表中还包含其他的非主键列，也必须具有唯一性，则为了避免该列中的值出现重复输入的情况，必须使用唯一约束（一

个数据表不能包含两个或两个以上的主键约束)。

唯一约束与主键约束的区别如下。

(1) 唯一约束指定的列可以为 NULL,但主键约束所在的列不允许为 NULL。

(2) 一个表中可以包含多个唯一约束,而主键约束则只能有一个。

若在创建表的同时创建唯一约束,其语法格式如下:

```
CREATE TABLE 表名
(列名 列属性 UNIQUE[,…n])
```

定义唯一约束的语法格式如下:

```
CONSTRAINT 约束名 UNIQUE[CLUSTERED | NONCLUSTERED]
column_name1[,colun_name2,…,column_name16]
```

【例 3.20】 在学生选课数据库的学生表中,为 sname 列添加唯一约束,保证姓名不重复。创建后使用 Transact-SQL 语句删除此约束。

在查询编辑器中执行如下 Transact-SQL 语句:

```
USE 学习选课
GO
ALTER TABLE student
ADD CONSTRAINT Un_sname UNIQUE(sname)
GO
ALTER TABLE student
DROP CONSTRAINT Un_sname
GO
```

3.4.6　检查约束

检查约束(CHECK 约束)实际上是验证字段输入内容的规则,表示一个字段的输入内容必须满足检查约束的条件。若不满足,则数据无法正常输入。用户可以对每个列设置检查约束。

1. 使用 Management Studio

【例 3.21】 在“学生选课”数据库的学生表中,为性别列(ssex)添加检查约束,保证性别列的输入值为“男”或“女”。

具体操作步骤如下。

(1) 启动 Management Studio,在“对象资源管理器”窗格中展开“数据库”→“学生选课”→“表”选项。

(2) 右击 student 表,在弹出的快捷菜单中选择“设计”命令,打开表设计器。

(3) 将光标定位到 ssex 字段。

(4) 右击并在弹出的快捷菜单中选择“CHECK 约束”命令,如图 3-17 所示。

(5) 打开“检查约束”对话框,单击“添加”按钮,在右侧表达式框中输入逻辑表达式“ssex='男' or ssex='女'”,如图 3-18 所示。

(6) 单击“关闭”按钮,关闭“检查约束”对话框。

(7) 单击工具栏上的 按钮,保存设置。

2. 使用 Transact-SQL 语句

创建检查约束的语法格式如下:

图 3-17　设置 CHECK 约束 1

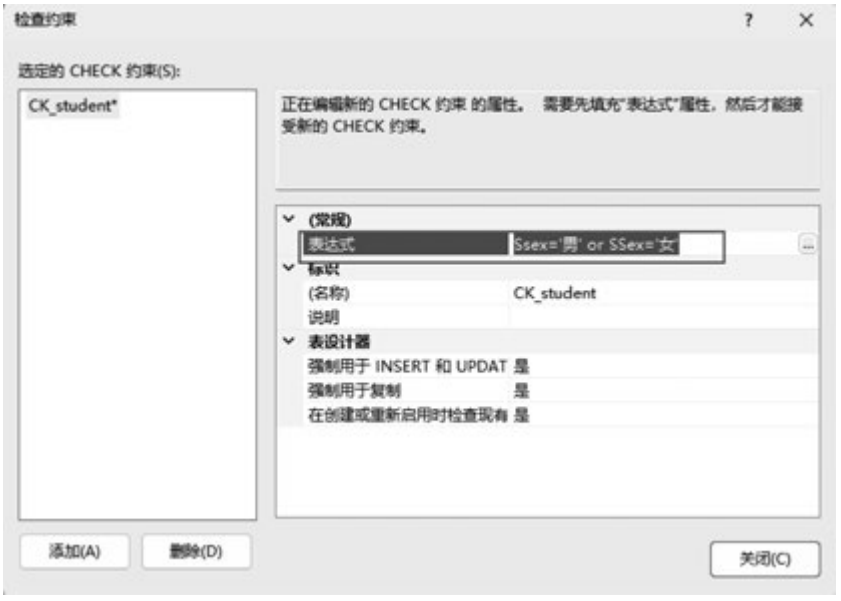

图 3-18　设置 CHECK 约束 2

```
CONSTRAINT  约束名  CHECK ( logical_expression)[,…n]
```

【例 3.22】 在"学生选课"数据库中，为了保证输入数据的质量，确保学生的年龄大于或等于 18 岁。

分析：在已经创建的学生表中，要保证年龄大于或等于 18 岁，可以在表中添加检查约束。

在查询编辑器中执行如下 Transact-SQL 语句：

```
ALTER TABLE student
ADD CONSTRAINT Ck_sage CHECK(sage > = 18)
```

3.4.7　默认值约束

默认值约束用于确保域完整性，它提供了一种为数据表中任何一列提供默认值的手段。默认值约束是指使用 INSERT 语句向数据表中插入数据时，如果没有为某一列指定数据，则将默认值随新记录一起存储到数据表的该列中。例如，在学生表 student 的 ssex 列定义了一个默认值约束，默认值为“男”，那么当添加新学生时，如果没有为其指定性别，则默认为“男”。

使用默认值约束时，用户需注意以下几点。

（1）默认值约束只能应用于 INSERT 语句，且定义的值必须与该列的数据类型和精度一致。

（2）在每一列上只能有一个默认值约束。如果有多个默认值约束，则系统将无法确定在该列上使用哪一个约束。

（3）默认值约束不能定义在指定了 IDENTITY 属性或数据类型为 timestamp 的列上，因为对于这些列，系统会自动提供数据，使用默认值约束是没有意义的。

（4）默认值约束允许使用一些系统函数提供的值。

1. 使用 Management Studio 创建默认值

【例 3.23】 在“学生选课”数据库的学生表中，为性别列设定默认值“男”。

具体操作步骤如下。

（1）启动 Management Studio，在“对象资源管理器”窗格中展开“数据库”→“学生选课”→“表”选项。

（2）右击 student 表，在弹出的快捷菜单中选择“设计”命令，打开表设计器。

（3）将光标定位到 ssex 字段。

（4）在“列属性”区域中，将“默认值或绑定”选项设为“男”，如图 3-19 所示。

图 3-19　默认值设置

（5）单击工具栏上的“保存”按钮，保存设置。

2. 使用 Transact-SQL 语句创建默认值

创建默认值约束的语法格式如下：

```
CONSTRAINT 约束名 DEFAULT constant_expression FOR 列名
```

其中，DEFAULT 为默认值；constant_expression 是用作列的默认值的常量、NULL 或系统函数。

【例 3.24】 使用 Transact-SQL 语句，实现例 3.23 中的默认值约束。

在查询编辑器窗口执行如下 Transact-SQL 语句：

```
USE 学生选课
GO
ALTER TABLE student
ADD CONSTRAINT Def_ssex DEFAULT '男' FOR ssex
```

3.4.8 约束禁用和启用

在一些特殊的情况下有禁用约束的需求，只有 FOREIGN KEY 和 CHECK 约束可被禁用。禁用的方法是在 ALTER TABLE 语句中使用 NOCHECK 并指定约束名，如果要禁用表的所有 CHECK 约束和 FOREIGN KEY 约束，则使用 ALL，方法是使用 NOCHECK。参考下面的语句。

禁用和启用 sc 表中所有的 CHECK 约束和 FOREIGN KEY 约束：

```
ALTER TABLE sc NOCHECK CONSTRAINT ALL
ALTER TABLE sc CHECK CONSTRAINT ALL
```

禁用和启用某张表的某个 CHECK 约束或外键约束：

```
ALTER TABLE sc NOCHECK CONSTRAINT FK_sc_student
ALTER TABLE sc CHECK CONSTRAINT FK_sc_student
```

可以通过系统存储过程 sp_helpconstraint 来查看表中的约束状态。例如查看 sc 表中的约束，结果如图 3-20 中的 status_enabled 列所示。当 CHECK 约束或外键约束被禁用时显示 disabled，启用时显示 enabled，其他约束为[n/a]（无相关内容）。

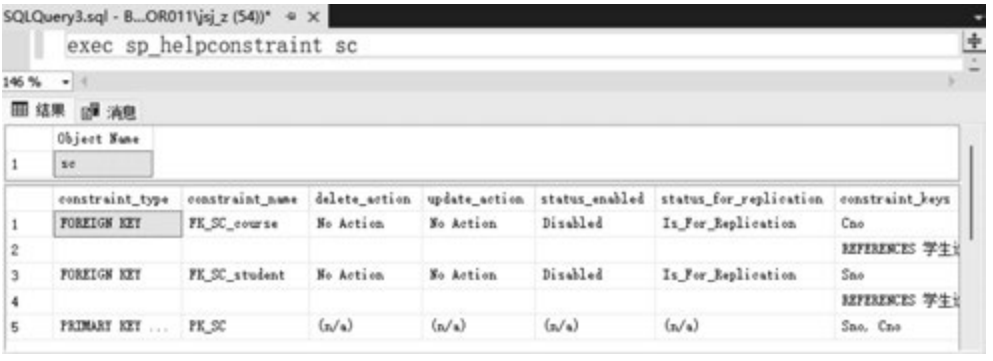

图 3-20 查看表中约束的信息

3.5　数据表中数据的操作

在 SQL Server 2022 中，表的基本结构确定以后，接着就是表中数据的处理：添加、修改和删除数据。数据操作有两种方法：一种是使用 Management Studio 操作表中数据；另一种是使用 Transact-SQL 语句。

【例 3.25】　在“学生选课”数据库的学生表 student 中添加、修改和删除数据。

具体操作步骤如下。

(1) 打开“对象资源管理器”窗格，展开“数据库”→“学生选课”→“表”选项。右击 dbo.student 表，在弹出的快捷菜单中选择“编辑前 200 行”命令，如图 3-21 所示。

(2) 打开的表数据编辑界面如图 3-22 所示，在其中可以添加、修改和删除记录。

图 3-21　打开表

	Sno	Sname	Ssex	Sage	sdept
▶	95001	刘超华	男	22	计算机系
	95002	刘晨	女	21	信息系
	95003	王敏	女	20	数学系
	95004	张海	男	23	数学系
	95005	陈平	男	21	数学系
	95006	陈斌斌	男	28	数学系
	95007	刘德虎	男	24	数学系
	95008	刘宝祥	男	22	计算机系
	95009	吕翠花	女	26	计算机系
	95010	马盛	男	23	数学系
	95011	吴鑫	男	22	计算机系
	95012	马伟	男	22	数学系
	95013	陈冬	男	18	信息系
	95014	李小鹏	男	22	计算机系
	95015	王娜	女	23	信息系
	95016	胡萌	女	23	计算机系
	95017	徐晓兰	女	21	计算机系
	95018	牛川	男	22	信息系
	95019	孙晓慧	女	23	信息系
*	NULL	NULL	NULL	NULL	NULL

图 3-22　表数据编辑界面

3.5.1　插入记录

INSERT 语句提供了添加数据的功能。INSERT 语句通常有两种形式：一种是插入一条记录；另一种是插入子查询的结果，一次可以插入多条记录。

INSERT 语句的语法格式如下：

```
INSERT[INTO]  表名  [(column_list)]
VALUES({DEFAULT | NULL | expression}[,…n])
```

参数说明如下。

INTO：用在 INSERT 关键字和目标表之间的可选关键字。

column_list：指定要插入数据的列，列名之间用逗号隔开。

DEFAULT：表示使用为此列指定的默认值。

Expression：指定一个常数变量或表达式。

1. 插入一条记录

【例 3.26】 在“学生选课”数据库中，向学生表 student 中插入一条记录。

具体操作步骤如下。

（1）在“对象资源管理器”中展开“数据库”→“学生选课”→“表”选项，右击 dbo.student 表，在弹出的快捷菜单中选择“编写表脚本为”→“INSERT 到”→“新查询编辑器窗口”命令，如图 3-23 所示。此时在窗口右侧的“查询编辑器”窗口中，提供了使用 INSERT 语句插入记录代码的基本框架。

图 3-23 插入记录 1

（2）修改代码。修改 VALUE 部分的语句，如图 3-24 所示，单击工具栏上的 ! 执行(X) 按钮，添加记录。

说明：默认显示的代码可以分为两部分，前半部分（INSERT INTO 部分）显示的是要插入的列名，后半部分（VALUES 部分）是要插入的具体列值，它们与前面的列一一对应，如

图 3-24　插入记录 2

果该列为空值，则可使用“,,”来表示，但不能删除。在 VALUES 代码部分还具有该列的数据属性，提示用户输入合适的数据。

（3）打开 student 表，验证插入的结果，如图 3-25 所示。若没有新记录显示，请先关闭打开的表，然后刷新并重新打开表。

Sno	Sname	Ssex	Sage	sdept
95001	刘丽华	男	22	计算机系
95002	刘晨	女	21	信息系
95003	王敏	女	20	数学系
95004	张海	男	23	数学系
95005	陈平	男	21	数学系
95006	陈斌斌	男	28	数学系
95007	刘晓虎	男	24	数学系
95008	刘宝祥	男	22	计算机系
95009	吕翠花	女	26	计算机系
95010	马盛	男	23	数学系
95011	吴霜	男	22	计算机系
95012	马伟	男	22	数学系
95013	陈冬	男	18	信息系
95014	李小鹏	男	22	计算机系
95015	王娜	女	23	信息系
95016	胡萌	女	23	计算机系
95017	徐晓兰	女	21	计算机系
95018	牛川	男	22	信息系
95019	孙晓慧	女	23	信息系
95020	王俊涛	男	23	信息系
NULL	NULL	NULL	NULL	NULL

图 3-25　插入记录 3

【例 3.27】 在“学生选课”数据库中，向学生表中插入一条记录：学号为 95021，姓名为“苏子墨”。

分析：插入一条记录，使用 INSERT INTO 语句。由于学生表 student 有 5 个字段，此处只给出两个字段的值，所以给出的值的顺序必须与列名的排列顺序相同。

在查询编辑器中执行如下 Transact-SQL 语句：

```
INSERT INTO student(sno,sname)  VALUES('95021','苏子墨')
```

2. 插入多行记录

在 INSERT 语句中使用 SELECT 子查询可以同时插入多行记录。INSERT 语句结合 SELECT 子查询可用于将一个或多个表或视图中的值添加到另一个表中。插入 SELECT 子查询的 INSERT 语句的语法格式如下：

```
INSERT  [INTO]  表名  [(column_list)]
SELECT column_list  FROM  table_list  WHERE search_condition
```

【例 3.28】 在“学生选课”数据库中，建立信息系学生表。

分析：在“学生选课”数据库中，没有信息系学生表，因此可以先建立表结构，然后通过查询学生表中的数据为新表插入数据。

在查询编辑器中执行如下 Transact-SQL 语句：

```
USE 学生选课
GO
CREATE TABLE Is_student(
sno Char(5),
sname Char(20),
ssex Char(2),
sage Smallint,
sdept Char(20))
GO
INSERT INTO Is_student
SELECT *
FROM student
WHERE  sdept = '信息系'
GO
```

说明：此处创建信息系学生表仅用于练习，用完请删除。

3.5.2 修改记录

修改表中数据可以用 UPDATE 语句来实现，其语法结构如下：

```
UPDATE  表名
SET  column_name = value [,column_name = value]
    [FROM  table_name]
[WHERE  condition ]
```

参数说明如下。

column_name：指定要修改的列名。

value：指出要更新的表的列应取的值。其有效值可以是表达式、列名和变量。

FROM table_name：指出 UPDATE 语句使用的表。

condition：指定修改行的条件。

【例 3.29】 在“学生选课”数据库中，将选课表中的所有成绩上调 5 分。

分析：对选课表中数据的修改使用 UPDATE 语句实现。

在查询编辑器中执行如下 Transact-SQL 语句：

```
USE 学生选课
    GO
    UPDATE sc SET grade = grade + 5
```

说明：执行完请还原数据。

3.5.3　删除记录

删除表中的数据时，可以用 DELETE 语句来实现，其语法格式如下：

```
DELETE  [FROM]   表名
[WHERE  condition]
```

其中，condition 指定删除行的条件。

【例 3.30】 将学生表 student 中学号为 95021 的学生删除。

在查询编辑器中执行如下 Transact-SQL 语句：

```
USE 学生选课
GO
DELETE student
WHERE sno = '95021'
GO
```

习题 3

扫描下方二维码在线答题。

在线习题