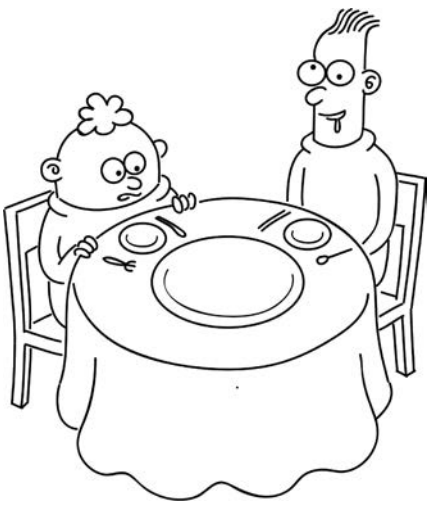
第 3 章

MySQL 表管理



本章首先介绍数据定义语言中的表管理。

3.1 关系模型的核心概念


微课视频

在学习创建表之前，首先要熟悉关系模型中的一些核心概念。

英国科学家埃德加·弗兰克·科德在 1976 年 6 月发表了《关于大型共享数据库数据的关系模型》论文，首先概述了关系数据模型及其原理，并把它用于数据库系统中。

关系模型的数据结构是二维表组成的集合，每个二维表又称为关系，因此可以说关系模型是关系组成的集合。假设要设计一个校园管理系统的数据库。校园管理系统中会有很多实体，如学生、课程和学生成绩等，这些实体构成的集合就是表，也称为关系。表 3-1 所示为校园管理系统中的一个学生表(关系)；表 3-2 所示为校园管理系统中的课程表(关系)；表 3-3 所示为校园管理系统中的学生成绩表(关系)。

表 3-1　学生表

学　　号	姓　　名	身份证号码
9904047	张烽	51 **********
9904048	朱强	51 **********
9904049	丁建辉	31 **********
9904050	陈丹	11 **********

表 3-2　课程表

编　　号	课　程　名	任 课 教 师	学　　时	学　　分
A06017	数据结构	管老师	51	5
A06021	操作系统	李老师	64	6

表 3-3　学生成绩表

学　　号	课 程 编 号	成　　绩
9904047	A06017	92
9904047	A06021	89
9904048	A06017	84
9904049	A06017	87
9904049	A06021	90
9904050	A06021	88

3.1.1　记录和字段

关系模型中的表是由若干行和列构成的，其中行称为元组，通常也称为记录；而列是记录中的数据项，通常也称为字段(Fields)。

3.1.2　键

关系模型的表结构中还有键(Key)的概念，也称为码。键又分为超键、候选键、主键和外键。

1. 超键

超键(Super Key，SK)也称为超码，能够唯一标识一行数据(记录)的字段或字段集。例如，表 1-1 所示的学生表中有 3 个字段，它们的组合有以下 6 种。

(1)(学号)；

(2)(学号，姓名)；

(3)(学号，身份证号码)；

(4)(姓名)；

(5)(姓名，身份证号码)；

(6)(身份证号码)。

由于超键要求能够唯一标识一行数据，且姓名字段是可以重复的，所以以下 5 种字段组

合可以作为超键。

(1)(学号);

(2)(学号,姓名);

(3)(学号,身份证号码);

(4)(姓名,身份证号码);

(5)(身份证号码)。

这5种组合中的任意一种都可以标识一行数据,因此它们中的任意一种都可以称为超键。

2. 候选键

候选键(Candidate Key,CK)也称为候选码。候选键也属于超键,且是包含最少字段的超键,所以候选键是不含多余字段的超键,超键组合中去掉任意字段,就不再是超键了。所以,在学生表的5种超键中,有以下两种属于候选键。

(1)(学号);

(2)(身份证号码)。

3. 主键

主键(Primary Key,PK)也称为主码。主键是从一组候选键中选择出来的,选择哪组候选键作为主键是由数据库设计人员决定的。在学生表中,推荐选择学号作为主键。

提示 主键和候选键的区别在于:一个表中只能有一个主键,但可以有多个候选键;主键不能为空值(NULL),而候选键可以包含空值,如学生表的学号或身份证号码都可以作为候选键,但如果选择学号作为主键,就不能再选择身份证号码作为主键了。

注意 主键和候选键有时可以是由多个字段组合而成的,如表1-3所示的学生成绩表,它的主键(学号,课程编号)是由两个字段组合而成的。

4. 外键

一个关系数据库可能包含多个表,可以通过外键(Foreign Key,FK)关联起来,外键也称为外码。例如,在校园管理系统中成绩表有以下两个外键。

(1) 学号:详细信息存储在学生表中,它是学生表中的主键;

(2) 课程编号:详细信息存储在课程表中,它是课程表中的主键。

3.1.3 约束条件

设计数据库表时,可以对表中的一个或多个字段的组合设置约束条件,检查该字段的输入值是否符合这个约束条件。约束分为表级约束和字段级约束,表级约束是对一个表中几个字段的约束,字段级约束则是对表中一个字段的约束。下面介绍几种常见的约束形式。

1. PRIMARY KEY 约束

PRIMARY KEY 即前面提到的主键，使用 PRIMARY KEY 约束可保证表中每条记录的唯一性。设计一个数据库表时，可以用一个或多个字段的组合作为这个表的主键。用单个字段作为主键时，使用了字段约束；用多个字段的组合作为主键时，则使用了表级约束。

主键的功能是保证某个字段或多个字段组合以后的值是唯一的。如果将多个字段的组合定义为主键，则包含在该组合中的个别字段的值允许重复，但是这些字段组合后的值必须是唯一的。在录入数据的过程中，必须在主键字段中输入数据，即主键字段不接受空值。

提示 空值(NULL)意味着用户还没有显式地为该字段输入数据，NULL 既不等价于数值型数据中的 0，也不等价于字符型数据中的空字符串或空格。不能把 NULL 视作大于、小于或等于任何其他值。

---

2. FOREIGN KEY 约束

FOREIGN KEY 即外键。外键字段的值必须在所引用的表中存在，所引用的表称为父表。父表通过主键字段或具有唯一性的字段与子表(包含外键表)的外键字段关联。

外键约束的主要作用是将彼此相关的表关联起来，以保证关联表之间的引用完整性。如果在外键字段中输入了一个非空值，但该值在所引用的表中并不存在，则这条记录也会被拒绝输入，因为这样输入将破坏关联表之间的引用完整性。

3. UNIQUE 约束

如果希望表中的一个字段值不重复，则应当对该字段添加 UNIQUE 约束。与 PRIMARY KEY 约束不同的是，一个表中可以有多个 UNIQUE 约束，且应用 UNIQUE 约束的单个或多个字段允许接受 NULL，候选键可以设置为 UNIQUE 约束。

4. CHECK 约束

CHECK 约束用于检查一个或多个字段的输入值是否满足指定的检查条件。在同一个字段上可以应用多个 CHECK 约束。在表中插入或修改数据时，CHECK 约束便会发生作用，如果插入或修改数据以后，字段中的数据不再符合该约束指定的条件，则数据不能被写入字段。例如，学生成绩表中的成绩如果采用百分制，则其取值范围是大于或等于 0 且小于或等于 100，那么就可以在学生成绩表中为成绩字段添加该约束。

5. DEFAULT 约束

DEFAULT(默认值)约束用于指定一个字段的默认值，当尚未在该字段中输入数据时，该字段中将自动填入这个默认值。例如，学生表中的成绩字段如果设置默认值为 0，那么在插入数据时，若不为成绩字段输入任何数据，则数据库系统会为该字段提供 0 值。


微课视频

3.2 管理数据库

模式(Schema)是数据库中所有对象的集合，包括表、字段、视图、索引和存储过程等。

注意 在 MySQL 中，模式就是数据库，即 Database，为了防止概念混淆，本书将模式统称为数据库。

3.2.1 创建数据库

在 2.3.3 节介绍了通过 MySQL Workbench 工具管理数据库(即 Schema)，这里不再赘述，本节重点介绍通过 SQL 代码创建数据库。通过 SQL 代码创建数据库的基本语法格式如下。

```
CREATE DATABASE db_name
```

创建数据库时，上面代码中的 DATABASE 可以换成 SCHEMA。

创建一个学校数据库(school_db)的示例代码如下。

```
-- 创建 school_db 数据库
CREATE DATABASE school_db ;
```

执行代码后会创建 school_db 数据库。SQL 代码中的“--”表示注释，它与注释内容之间会隔一个空格。

SQL 代码执行过程可以参考 2.3.5 节，这里不再赘述。

3.2.2 删除数据库

有时候还需要删除数据库，删除数据库的基本语法格式如下。

```
DROP DATABASE db_name
```

删除学校数据库(school_db)的示例代码如下。

```
-- 删除数据库 school_db
DROP DATABASE school_db ;
```

上述代码执行后，会删除 school_db 数据库，但是如果数据库不存在，则会发生如下错误。

```
Error Code: 1008. Can't drop database 'school_db'; database doesn't exist
```

使用 MySQL Workbench 工具执行上述 SQL 代码，结果如图 3-1 所示。

为了防止删除不存在的数据库而引发的错误，可以使用 IF EXISTS 子句判断数据库是否存在，修改代码如下。

```
-- 删除数据库 school_db
DROP DATABASE IF EXISTS school_db;
```

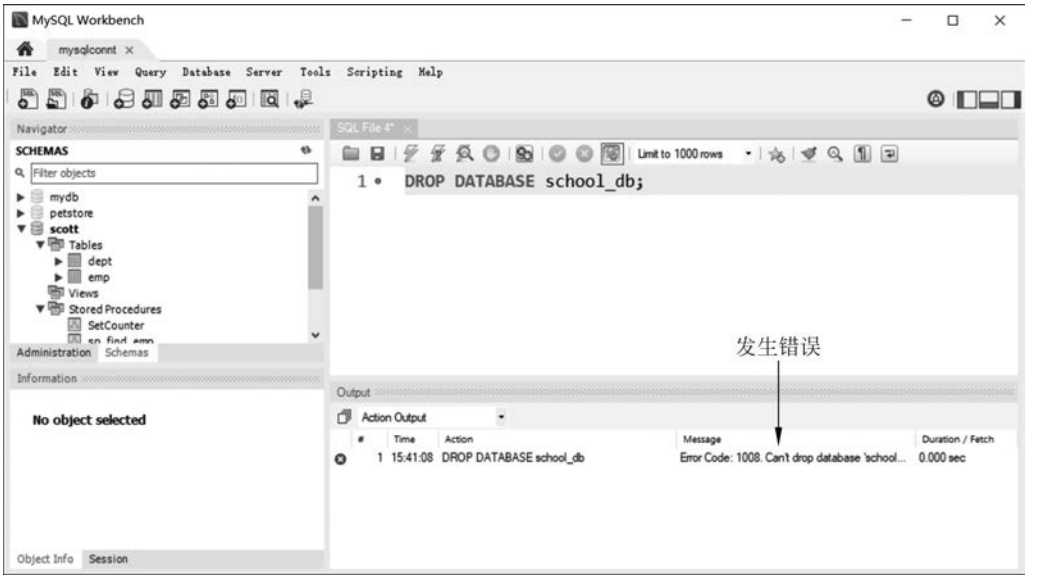


图 3-1　删除数据库执行结果

3.2.3　选择数据库

由于可以有多个数据库，不同的数据库中又有很多不同的对象，因此选择数据库也是非常重要的，选择数据库可以使用 USE 语句实现。

选择 school_db 数据库示例代码如下。

```
-- 选择使用 school_db 数据库
USE school_db;
```

如图 3-2 所示的 3 个数据库都未被选中，现使用 USE 语句选择某个数据库，界面如图 3-3 所示，被选中的数据库会用加粗字体显示。

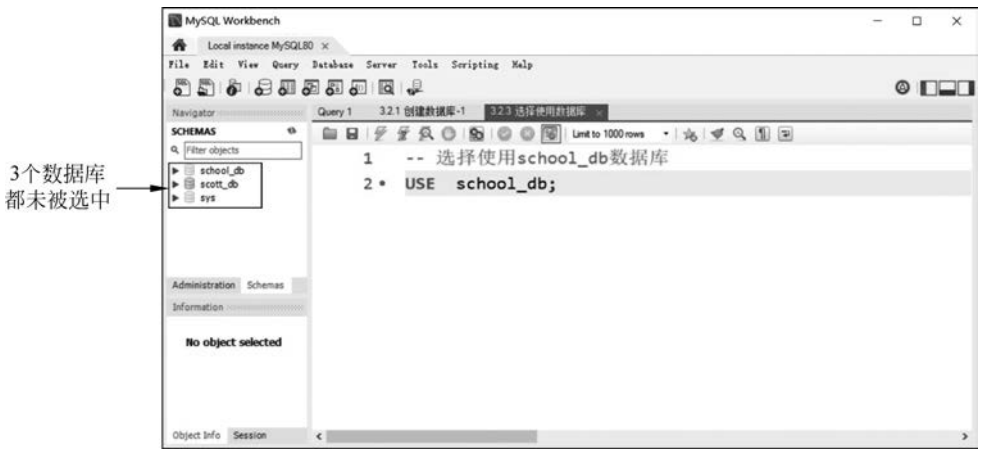


图 3-2　3 个数据库都未被选中

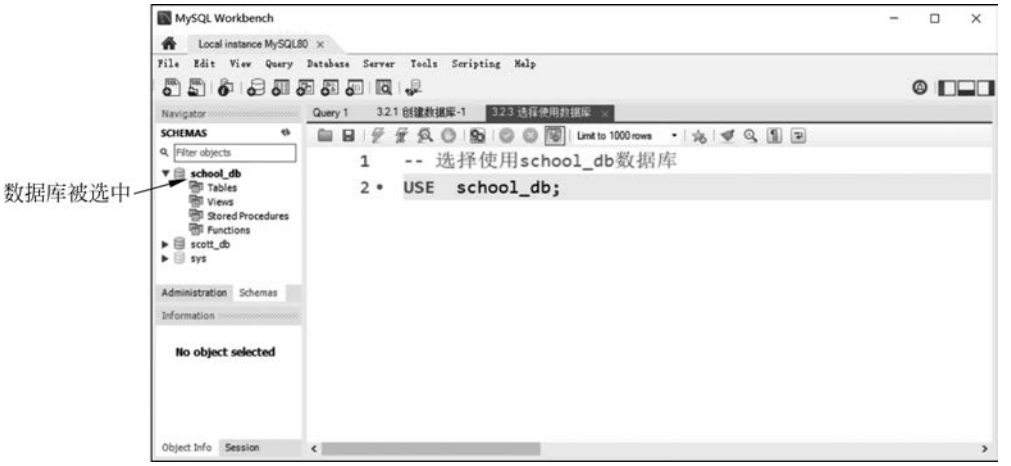


图 3-3 使用 USE 语句选中数据库后的界面

3.3 创建表


微课视频

表管理包括创建表、修改表和删除表操作，本节介绍创建表。

在数据库中创建表可以使用 CREATE TABLE 语句。CREATE TABLE 语句的基本语法格式如下。

```
CREATE TABLE table_name (
table_field1 datatype[(size)],
table_field2 datatype[(size)],
    …
)
```

其中，table_name 是表名，table_field1 和 table_field2 等是表中的字段。表名和字段名是开发人员自定义的名称，但一般不推荐中文名称，如果有多个英文单词，推荐使用下画线分隔，如 s_id 和 s_name。

语法结构中的 datatype 是字段的数据类型；size 指定数据类型所占用的内存空间。注意语法结构中括号“[]”中的内容可以省略，因此[(size)]表示 size 是可以省略的。如果要定义多个字段，则字段之间要用逗号“,”分隔，但是最后一个字段之后要省略逗号。

下面通过示例介绍 CREATE TABLE 语句的使用。创建学生表，结构如表 3-4 所示。

表 3-4 学生表结构

字 段 名	数 据 类 型	长 度	备 注
s_id	INTEGER		学号
s_name	VARCHAR(20)	20	姓名
gender	CHAR(1)	1	性别，F 表示女，M 表示男
PIN	CHAR(18)	18	身份证号码

创建学生表的示例代码如下。

```
-- 3.3 创建表
-- 创建学生表的语句
CREATE TABLE student(
    s_id INTEGER,           -- 学号                          ①
    s_name  VARCHAR(20),    -- 姓名                          ②
    gender  CHAR(1),        -- 性别,'F'表示女,'M'表示男      ③
    PIN     CHAR(18)        -- 身份证号码                    ④
)
```

由于创建表的语句属于DDL，因此建表的SQL文件扩展名可以为.ddl或.sql。它是一个文本文件，可以通过任意文本编辑工具进行编辑。这种文件通常可以通过数据库管理工具执行，因此也称为脚本文件。

代码第①行定义s_id字段，其中INTEGER指定字段为整数类型。

代码第②行定义s_name字段，其中VARCHAR(20)表示长度可变，且最大长度为20字节的字符串类型。

代码第③行定义gender字段，其中CHAR(1)表示固定长度为1字节的字符串类型，其取值为'F'或'M'。

代码第④行定义PIN字段，目前身份证号码为18位字符串，因此该字段数据类型设置为CHAR(18)，表示固定长度为18位的字符串类型。

注意 上述代码运行后会创建student表，但是如果没有使用USE语句选中数据库，也没有设置默认数据库，则会发生Error Code：1046. No database selected错误。


微课视频

3.4 字段数据类型

在创建表时，要求为每个字段指定具体的数据类型。关系数据共分为4种类型：字符串数据、数值数据、日期和时间数据及大型对象数据。

3.4.1 字符串数据

多数数据库都提供以下两种类型字符串数据。

(1) 固定长度(CHAR)字符串：总是占据等量的内存空间，不管实际上它们存储的数据量有多少。

(2) 可变长度(VARCHAR)：可变长度的字符串只占据它们的内容所消耗的内存量。

例如，CHAR(2)表示固定两个字节长度的字符串，当只输入一个字节时，对于未占用的空间，数据库会用空格补位，使之能够始终保持两个字节的占位，这就是所谓的固定长度的字符串；VARCHAR(2)表示两个字节可变长度的字符串，当输入的字符串不足两个字节时，数据库不会补位。

提示 如果不能确定字符串的长度，则可以使用 TEXT 类型，该类型可以存储大量文字数据。

3.4.2 数值数据

多数数据库都提供至少两种类型数值数据：整数（INTEGER）和浮点数（FLOAT 或 REAL）。

整数和浮点数可以统一用 numeric[（p[,s] ）]表示。其中，numeric 表示十进制数值数据；p 为精度，即整数位数与小数位数之和；s 为小数位数。此外，还有一些数据库提供更加独特的数字类型。

3.4.3 日期和时间数据

多数关系数据库支持的另一种独特的数据类型是日期和时间数据。数据库处理日期和时间数据的方式有很多种，日期和时间数据中日期的存储和显示方法都可以变化，有些数据库还支持更多类型的时间数据。本质上，关系数据库所支持的 3 种类型日期和时间数据为日期、时间、日期＋时间组合。

3.4.4 大型对象

大多数数据库为字段提供大型对象类型数据，大型对象主要分为以下两种类型数据。

（1）大文本（CLOB）：保存大量的文本数据。有的数据库中大文本可以容纳高达 4GB 的数据，有的数据库提供使用 TEXT 类型作为大文本数据类型。

（2）大二进制（BLOB）：保存大量的二进制数据，如图片、视频等二进制文件数据。

3.5 指定键

键是数据库的一种约束行为，它对于防止数据重复、保证数据的完整性是非常重要的。在定义表时可以指定键，包括候选键（CK）、主键（PK）和外键（FK）。

3.5.1 指定候选键


微课视频

指定表的候选键使用 UNIQUE 关键字实现，语法格式有两种。

1. 在定义字段时指定

示例代码如下。

```
-- 指定候选键
-- 创建学生表语句
CREATE TABLE student(
    s_id INTEGER,                -- 学号
    s_name  VARCHAR(20),         -- 姓名
    gender  CHAR(1) ,            -- 性别, 'F'表示女, 'M'表示男
```

```
    PIN      CHAR(18) UNIQUE      -- 身份证号码                    ①
)
```

代码第①行定义 PIN 字段，可见在定义 PIN 字段后面使用 UNIQUE 关键字，这样就将 PIN 字段指定为候选键了。

2. 在 CREATE TABLE 语句结尾处添加 UNIQUE 子句指定

示例代码如下。

```
-- 指定候选键
-- 创建学生表语句
CREATE TABLE student(
    s_id     INTEGER,           -- 学号
    s_name   VARCHAR(20),       -- 姓名
    gender   CHAR(1),           -- 性别, 'F'表示女,'M'表示男
    PIN      CHAR(18),          -- 身份证号码
    UNIQUE   (PIN)              -- 定义身份证号码为候选键          ①
)
```

代码第①行在 CREATE TABLE 语句结尾处添加 UNIQUE 子句(单独一行)，注意它与其他字段定义语句用逗号分隔。

学生表创建完成后，可以使用 MySQL Workbench 工具测试候选键，如图 3-4 所示。试图通过 INSERT 语句插入两条数据，注意它们的 PIN 字段数据是重复的 51 ********** 3，会引发违反候选键约束错误 Error Code：1062. Duplicate entry '51 ********** 3' for key 'student.PIN'。

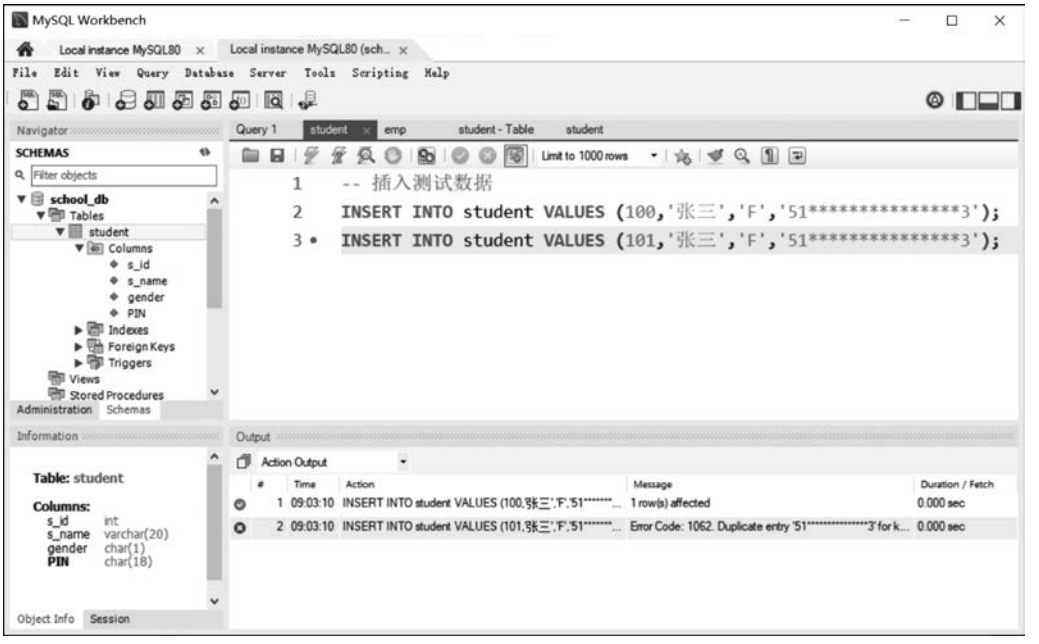


图 3-4　测试候选键 1

候选键可以是一个字段，也可以是多个字段的组合，上述示例介绍的是一个字段作为候选键的情况，下面再介绍多个字段组合作为候选键的示例，该示例是创建一个学生成绩表(student_score)。学生成绩表的相关信息如表 3-5 所示。

表 3-5 学生成绩表

字 段 名	数 据 类 型	长 度	是否候选键	备 注
s_id	INTEGER		是	学号
c_id	INTEGER		是	课程编号
score	INTEGER		否	成绩

创建学生成绩表的示例代码如下。

```
-- 指定多个字段组合候选键
-- 创建学生成绩表语句
CREATE TABLE student(
    s_id     INTEGER,          -- 学号
    c_id     INTEGER,          -- 课程编号
    score    INTEGER,          -- 成绩
    UNIQUE   (s_id,c_id)       -- 定义多字段组合候选键          ①
)
```

代码第①行指定 s_id 和 c_id 字段为组合候选键。

学生成绩表创建完成之后，可以测试候选键，使用 MySQL Workbench 工具测试候选键，试图通过 INSERT 语句插入数据，如图 3-5 所示，如果候选键有重复数据，则会引发错误 Error Code：1062. Duplicate entry '100-2' for key 'student_score. s_id'。

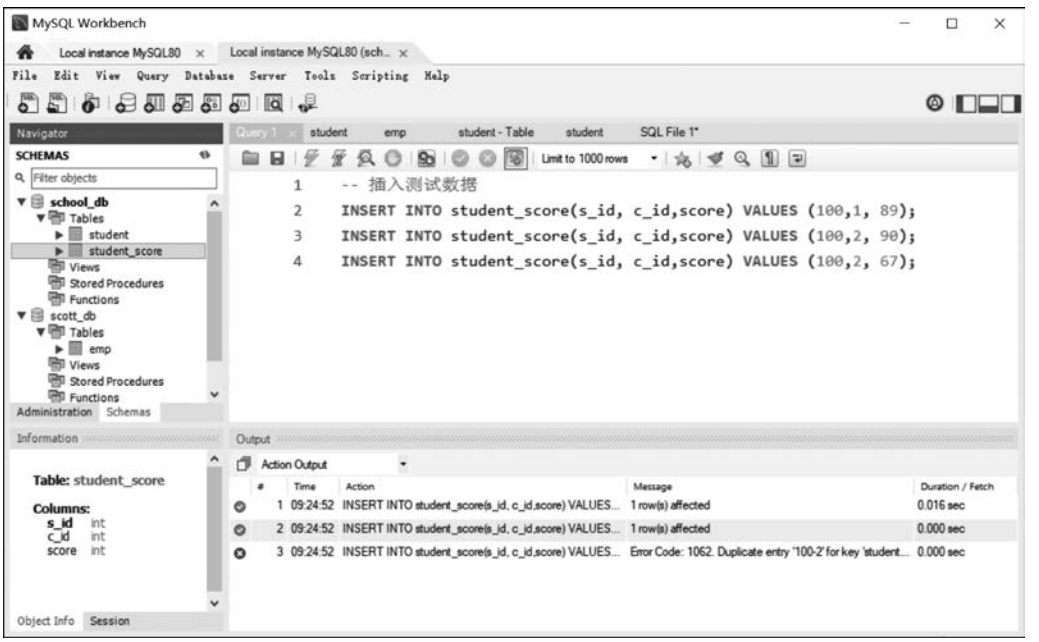


图 3-5 测试候选键 2


微课视频

3.5.2 指定主键

可以使用 PRIMARY KEY 关键字指定主键，它可以与 UNIQUE 关键字一起用在 CREATE TABLE 语句中。指定主键的方法也有两种。

1. 定义字段时指定

示例代码如下。

```
-- 指定主键
-- 创建学生表语句
CREATE TABLE student(
    s_id INTEGER PRIMARY KEY,        -- 学号                    ①
    s_name  VARCHAR(20),

    s_name  VARCHAR(20),
    gender  CHAR(1),
    PIN     UNIQUE CHAR(18)) UNIQUE
)
```

代码第①行定义 s_id 字段，可见在定义 s_id 字段后面使用 PRIMARY KEY 关键字，就可以将 s_id 字段指定为主键了。

2. 在 CREATE TABLE 语句结尾处添加 PRIMARY KEY 子句指定

示例代码如下。

```
-- 指定主键
-- 创建学生表语句
CREATE TABLE student (
    s_id    INTEGER ,      -- 学号
    s_name  VARCHAR(20),
    gender  CHAR(1),
    PIN  CHAR(18) UNIQUE,                          ①
    PRIMARY KEY(s_id)                              ②
)
```

代码第①行指定候选键，代码第②行指定主键。主键和候选键都可以防止数据重复，读者可以参考候选键测试方法测试，这里不再赘述。

主键也可以是一个字段或多个字段的组合。修改学生成绩表，如表 3-6 所示，学生成绩表的主键是由 s_id 和 c_id 两个字段组合而成的。

表 3-6 学生成绩表

字 段 名	数 据 类 型	长 度	是 否 主 键	备 注
s_id	INTEGER		是	学号
c_id	INTEGER		是	课程编号
score	INTEGER		否	成绩

创建学生成绩表代码如下。

```
-- 指定主键
-- 创建学生成绩表语句
CREATE TABLE student_score(
    s_id     INTEGER,              -- 学号
    c_id     INTEGER,              -- 课程编号
    score    INTEGER,              -- 成绩
    PRIMARY KEY (s_id,c_id)        -- 定义多字段组合主键                    ①
)
```

代码第①行指定 s_id 和 c_id 字段为主键。

3.5.3 指定外键


微课视频

指定外键使用 REFERENCES 关键字实现。将表 3-3 所示的学生成绩表中的学号字段(s_id)引用到表 3-1 所示的学生表中的学号字段(s_id)。学生成绩表称为子表,学生表称为父表。

提示 这种表之间的外键关联关系通过文字描述不够形象,在数据库设计中,这种关系可以通过 ER(实体关系)图描述,如图 3-6 所示,学生成绩表有两个外键(学号和课程编号),学生成绩表通过学号关联到学生表。另外,学生成绩表通过课程编号关联到课程表。

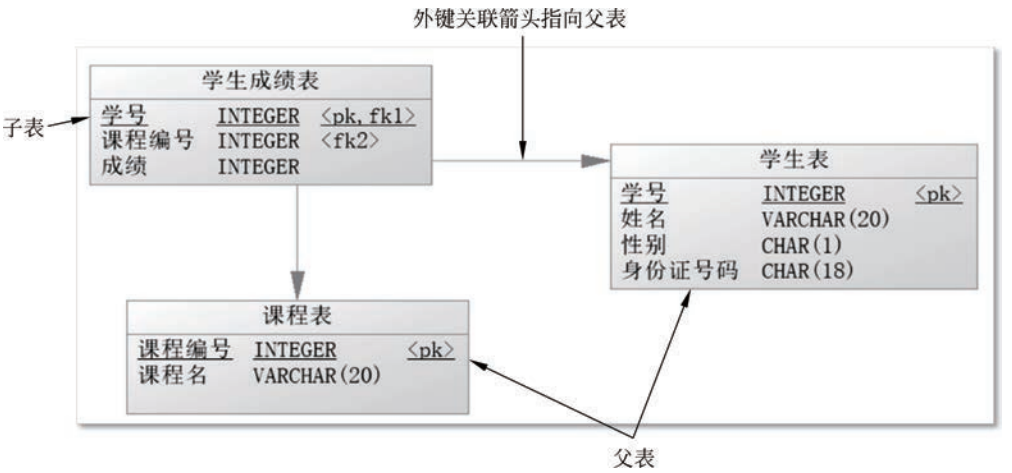


图 3-6 ER 图

指定外键的方法也有两种。

1. 在定义字段时通过 REFERENCES 关键字指定

示例代码如下。

```
-- 指定外键
-- 创建学生成绩表语句
```

```
CREATE TABLE student_score(
    s_id     INTEGER REFERENCES student(s_id),      -- 学号              ①
    c_id     INTEGER,                               -- 课程编号
    score    INTEGER,                               -- 成绩
    PRIMARY KEY (s_id,c_id)
)
```

代码第①行定义 s_id 字段。在定义 s_id 字段时，后面使用 REFERENCES 关键字指定外键关联的父表及字段，这里的 s_id 字段就是外键。

2. 在 CREATE TABLE 语句结尾处添加 FOREIGN KEY 子句指定

示例代码如下。

```
-- 指定外键
-- 创建学生成绩表语句

CREATE TABLE student_score(
    s_id     INTEGER ,                  -- 学号
    c_id     INTEGER,                   -- 课程编号
    score    INTEGER,                   -- 成绩
    PRIMARY KEY (s_id,c_id),
    FOREIGN KEY (s_id) REFERENCES student(s_id)                    ①
)
```

代码第①行是 FOREIGN KEY 子句，FOREIGN KEY 关键字后面的(s_id)是指定表的外键。


微课视频

3.6 其他约束

除了指定键约束外，还有指定默认值、禁止空值和设置 CHECK 约束等。

3.6.1 指定默认值

在定义表时可以为字段指定默认值，使用 DEFAULT 关键字实现。例如，在定义学生表时，可以为性别字段设置默认值'F'。示例代码如下。

```
-- 创建学生表
-- 指定默认值
CREATE TABLE student(
    s_id INTEGER,                  -- 学号
    s_name   VARCHAR(20),          -- 姓名
    gender   CHAR(1) DEFAULT 'F',  -- 性别,'F'表示女,'M'表示男,默认值为'F'      ①
    PIN      CHAR(18) UNIQUE       -- 身份证号码
)
```

代码第①行为性别(gender)字段设置默认值'F'，表示默认为女性。当没有给性别字段提供数据时，数据库系统会为其提供默认值'F'。如图 3-7 所示，插入数据时，没有为性别(gender)字段提供数据，则系统会为其提供默认值'F'。

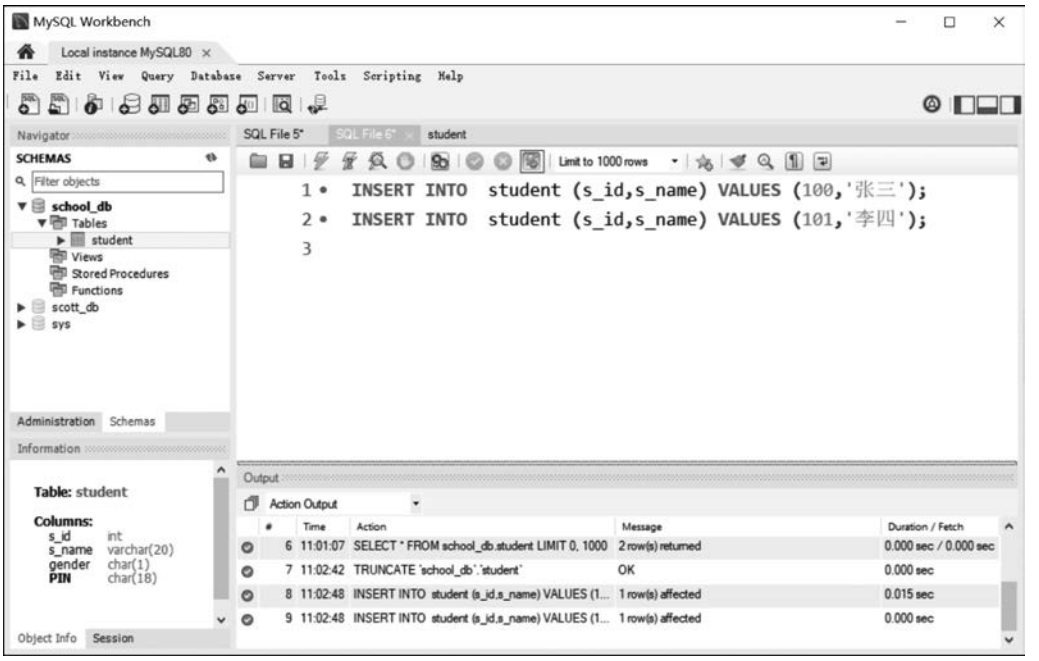


图 3-7 指定默认值执行结果

3.6.2 禁止空值

有时输入空值会引起严重的程序错误，在定义字段时，可以使用 NOT NULL 关键字设置字段禁止输入空值。

示例代码如下。

```
-- 创建学生表
-- 禁止空值
CREATE TABLE student(
    s_id INTEGER,                  -- 学号
    s_name  VARCHAR(20) NOT NULL,  -- 姓名                ①
    gender  CHAR(1) DEFAULT 'F',   -- 性别,'F'表示女,'M'表示男,默认值为'F'
    PIN     CHAR(18) UNIQUE        -- 身份证号码
)
```

代码第①行为姓名(s_name)字段设置禁止空值。插入数据时，如果没有为姓名(s_name)字段提供数据，则会引发错误 Error Code：1364. Field 's_name' doesn't have a default value。

3.6.3 设置 CHECK 约束

CHECK 关键字用来限制字段所能接收的数据。例如，在学生成绩表中可以限制成绩(score)字段的值为 0～100。示例代码如下。

```
-- 指定外键
-- 创建学生成绩表语句
CREATE TABLE student_score(
    s_id     INTEGER REFERENCES student(s_id),                    -- 学号
    c_id     INTEGER,                                             -- 课程编号
    score    INTEGER CHECK (score >= 0 AND score <= 100),         -- 成绩          ①
    PRIMARY KEY (s_id,c_id)
)
```

代码第①行在定义 score 字段时设置对该字段的限制，CHECK 关键字后面的表达式“(score >= 0 AND score <= 100)”是限制条件，其中>= 和<= 为条件运算符，AND 为逻辑运算符，表示“逻辑与”，类似的还有 OR 表示“逻辑或”，NOT 表示“逻辑非”。有关条件运算符和逻辑运算符将在第 7 章详细介绍。

学生成绩表创建完成后，可以测试 CHECK 约束，如图 3-8 所示。通过 INSERT 语句插入数据时，试图为 score 字段输入值－20，则会引发 Error Code：1136. Column count doesn't match value count at row 1 错误。

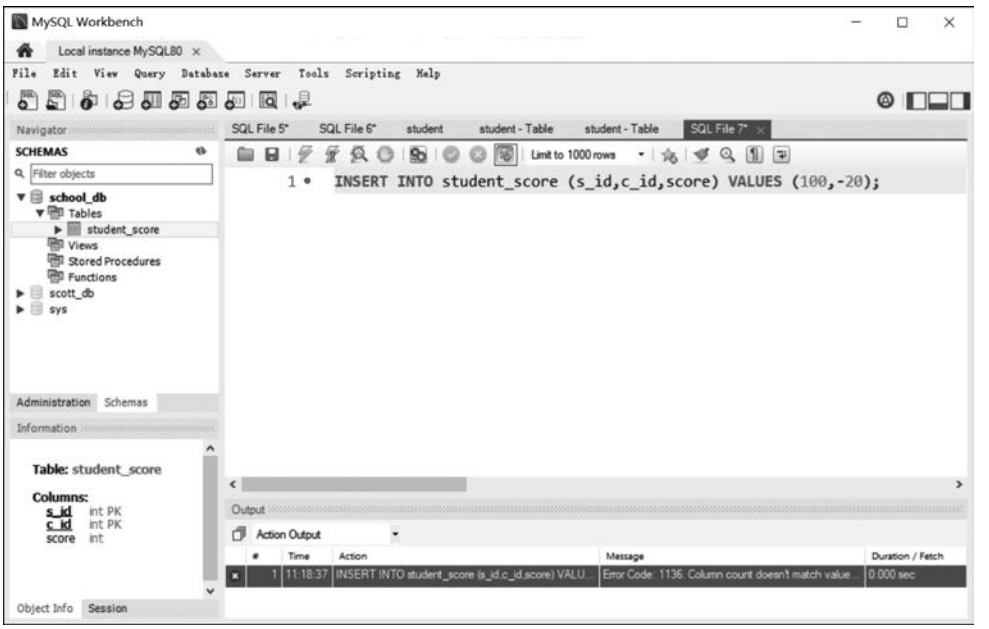


图 3-8　CHECK 约束执行结果


微课视频

3.7　修改表

有时表建立后，由于某种原因需要修改表的结构或字段的定义。使用 ALTER TABLE 语句可以修改表的结构。下面介绍如何通过 ALTER TABLE 语句实现修改表名、修改字段名、添加字段和删除字段等操作。

3.7.1 修改表名

修改表名的 ALTER TABLE 语句基本语法格式如下。

```
ALTER TABLE table_name
RENAME TO new_table_name
```

其中，table_name 为要修改的表名，new_table_name 为修改后的表名。

> 注意 不同数据库中的 ALTER TABLE 语句的语法格式有很大的不同，上述 ALTER TABLE 语句语法格式主要支持 Oracle 和 MySQL 数据库。

示例代码如下。

```
-- 修改表名
-- 将表名 student 修改为 stu_table
ALTER TABLE student RENAME TO stu_table;                ①
```

代码第①行将 student 表名修改为 stu_table，如图 3-9 所示。

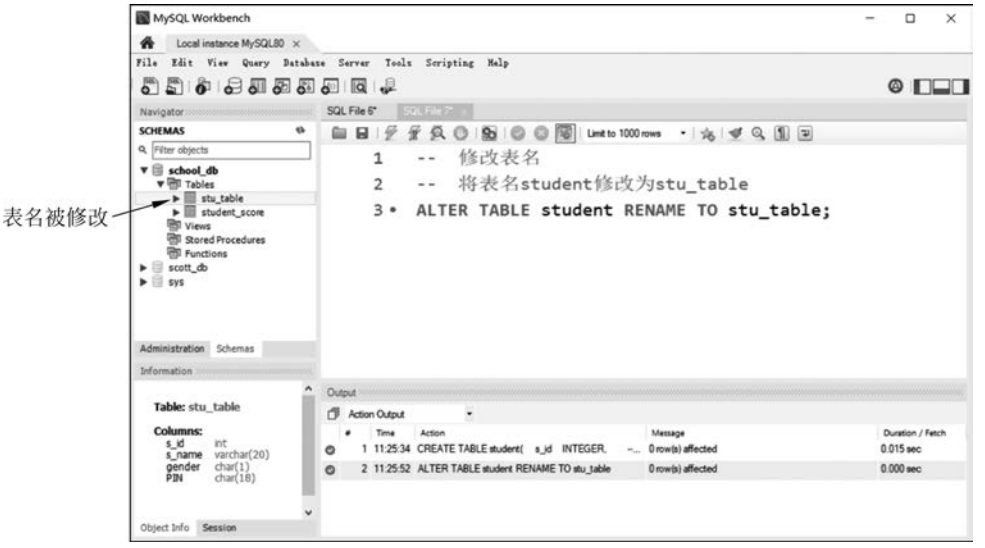


图 3-9 修改表名执行结果

3.7.2 添加字段

有时表已经创建好了，甚至使用了一段时间，表中有了一些数据，这时如果删除表再新建，代价就很大。此时，可以使用 ALTER TABLE 语句中的 ADD 语句在现有表中添加字段，语法格式如下。

```
ALTER TABLE table_name ADD field_name datatype[(size)]
```

其中，table_name 为要修改的表名，field_name 为要添加的字段。

以下代码是在 student 表中添加两个字段。

```
-- 在 student 表中添加 birthday(生日)和 phone(电话)字段
ALTER TABLE student ADD     birthday CHAR(10);        ①
ALTER TABLE student ADD     phone VARCHAR(20);        ②
```

代码第①行在 student 表中添加 birthday 字段，代码第②行在 student 表中添加 phone 字段。在 MySQL Workbench 工具中执行上述 SQL 语句，结果如图 3-10 所示。

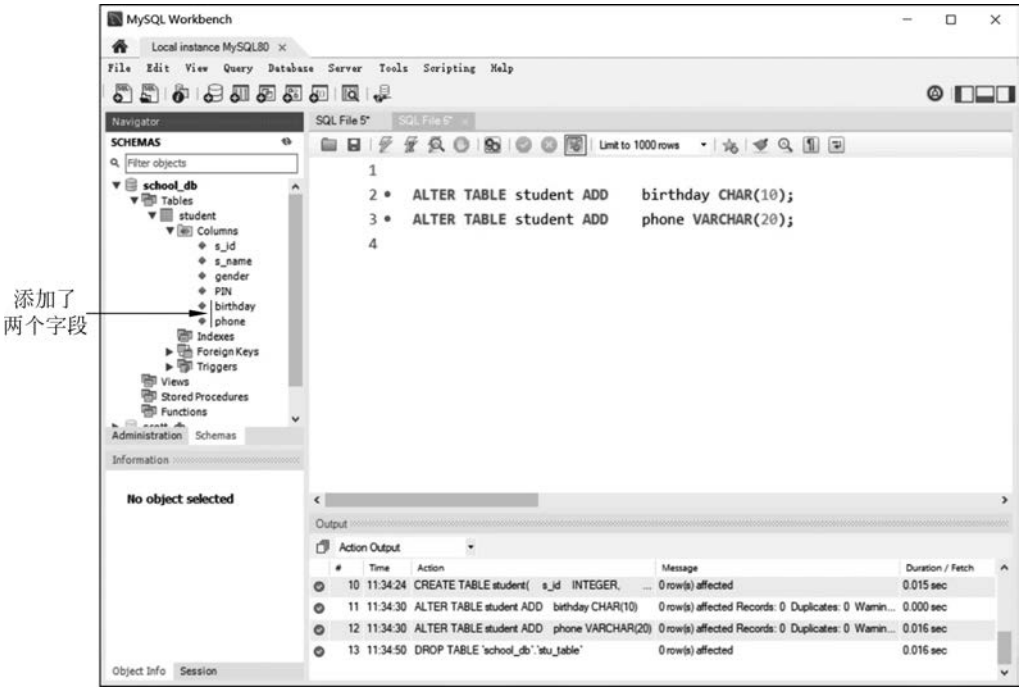


图 3-10　添加字段执行结果

3.7.3　删除字段

既然可以在现有表中添加字段，当然也可以删除字段。可以使用 ALTER TABLE 语句的 DROP COLUMN 语句在现有表中删除字段，语法格式如下。

```
ALTER TABLE table_name DROP COLUMN field_name
```

以下代码是从 student 表中删除 birthday 字段。

```
-- 从 student 表中删除 birthday 字段
ALTER TABLE student DROP COLUMN birthday;                ①
```

代码第①行从 student 表中删除 birthday 字段。在 MySQL Workbench 工具中执行上述 SQL 语句，结果如图 3-11 所示。

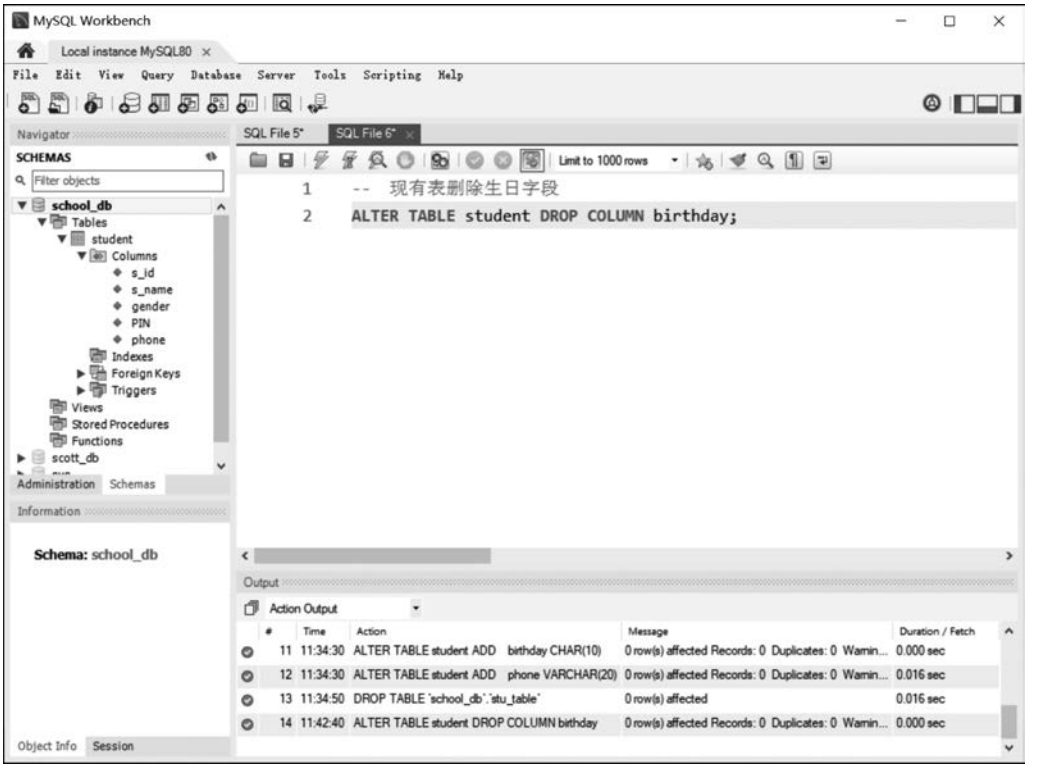


图 3-11 删除字段执行结果

3.8 删除表

通过 DROP TABLE 语句实现删除表，语法格式如下。

```
DROP TABLE [IF EXISTS] table_name
```

注意，中括号“[]”中的内容是可以省略的。

删除 student 表的示例代码如下。

```
-- 删除 student 表
DROP TABLE student;                    ①
```

上述代码执行结果如图 3-12 所示，可见 student 表被删除了。

但是如果要删除的表不存在，则将发生 Error Code：1051. Unknown table 'school_db.student'错误。

为了防止错误发生，可以增加 IF EXISTS 子句，示例代码如下。

```
DROP TABLE IF EXISTS student;
```

如果表不存在，上述示例代码执行时不会发生错误，但会出现警告：1051 Unknown table 'school_db.student'，如图 3-13 所示。

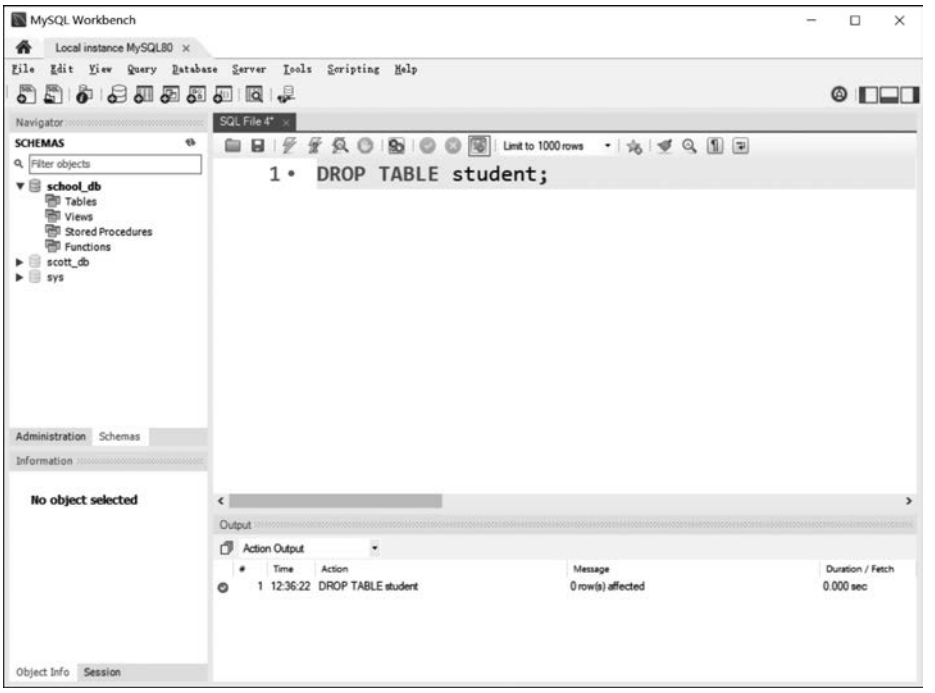


图 3-12　删除表执行结果

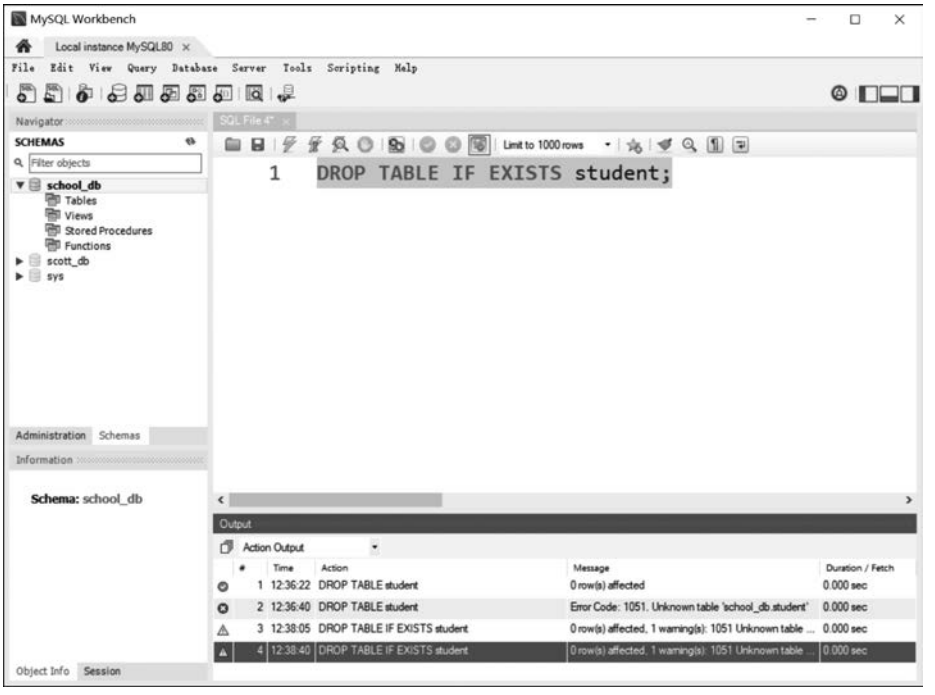


图 3-13　表不存在时的执行结果

3.9 动手练一练

1. 简答题

（1）请简述表记录和字段。

（2）请说明主键、候选键和外键的区别。

2. 选择题

下列哪些约束可以防止数据重复？（　　）

A. UNIQUE 约束　　B. FOREIGN KEY 约束

C. PRIMARY KEY 约束　　D. CHECK 约束

3. 操作题

（1）使用命令提示符工具登录 MySQL 数据库服务器，并创建 MyDB 数据库。

（2）使用命令提示符工具登录 MySQL 数据库服务器，并在 MyDB 数据库中创建 teacher 表。