数据库及实现 上机实验报告 4

邓淇升 大数据学院 16307110232

实验一：数据完整性

一、实验目的

1. 掌握 T-SQL 语句(CREATE RULE、DROP RULE)创建和删除规则的方法。

2. 掌握系统储存过程 sp_bindrule、sp_unbindrule 绑定和解除绑定规则的操作方法，以及 sp_help、sp_helptext 查询规则信息和 sp_rename 更名规则的方法。

3. 掌握 T-SQL 语句(CREATE DEFAULT、DROP DEFAULT)创建和删除默认对象的方法。

4. 掌握系统储存过程 sp_bindefault、sp_unbindefault 绑定和解除绑定默认对象的操作方法，以及 sp_helptext 查询规则信息。

5. 掌握 SQL Server 管理平台和 T-SQL 语句(CREATE TABLE、ALTER TABLE)定义和删除约束的方法，并了解约束的类型。

二、实验环境

软件配置：Microsoft SQL Server 2008 R2

操作平台：Windows 10

系统类型：64 位操作系统

三、实验内容

1. 创建规则，限制输入电话号码为 7 位 0-9 数字。为 studentsdb_dengqisheng 数据库创建一个规则 phone_rule，再用系统存储过程 sp_bindrule 将该规则绑定到 stu_phone_dengqisheng 表格的“电话号码”列上。

查询结果显示如图 1-1-1 至图 1-1-3。

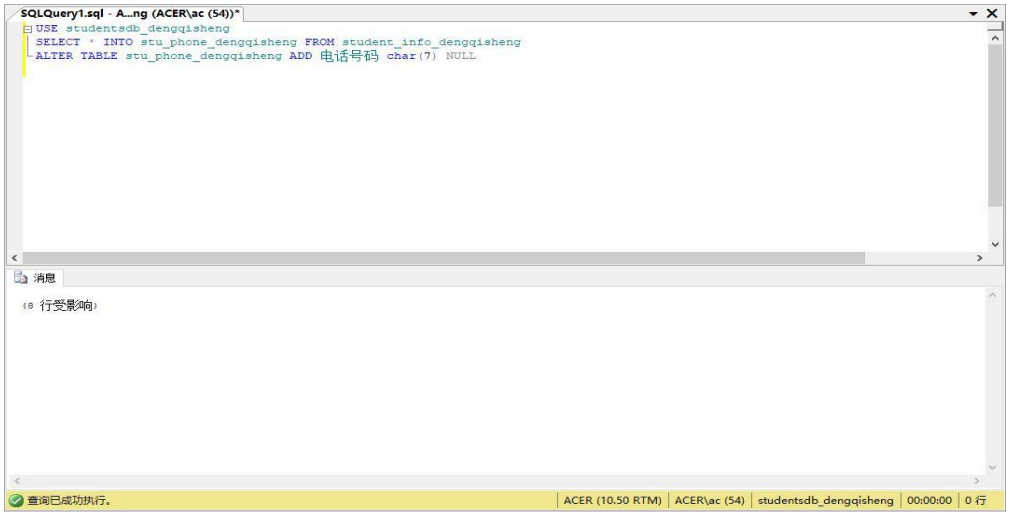

图 1-1-1　创建 stu_phone_dengqisheng 表

图 1-1-2　创建 phone_rule 规则

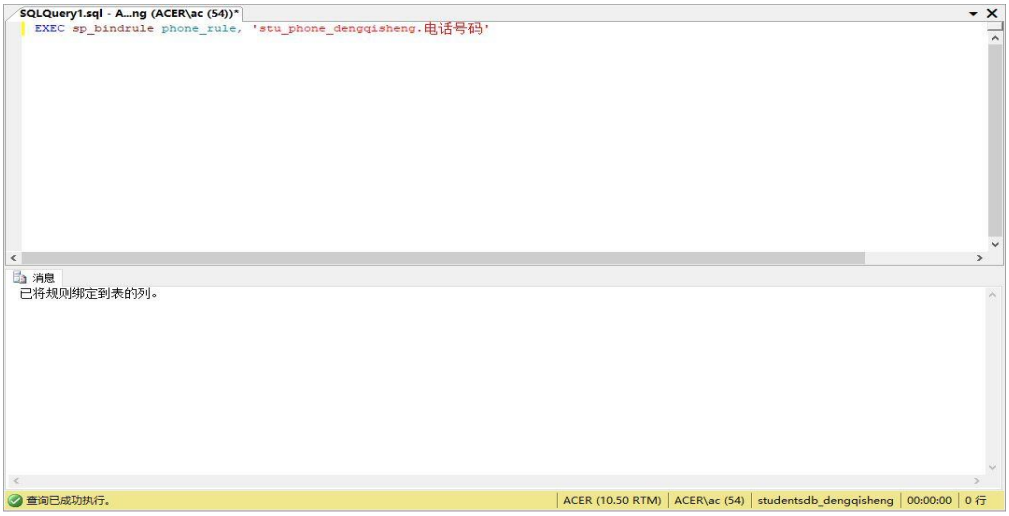

图 1-1-3　绑定 phone_rule 规则

2. 创建 stusex_rule，将其绑定到表格 stu_phone_dengqisheng 的“性别”列上，保证输入只能是男或女。

查询结果显示如图 1-2-1 和图 1-2-2。

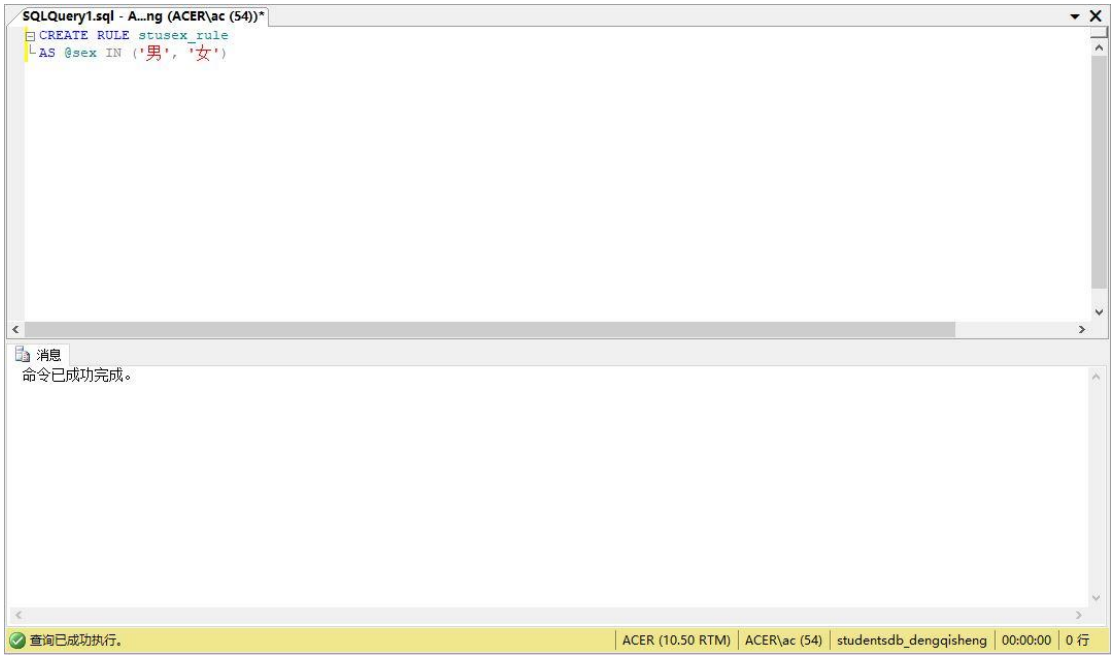

图 1-2-1　创建 stusex_rule 规则

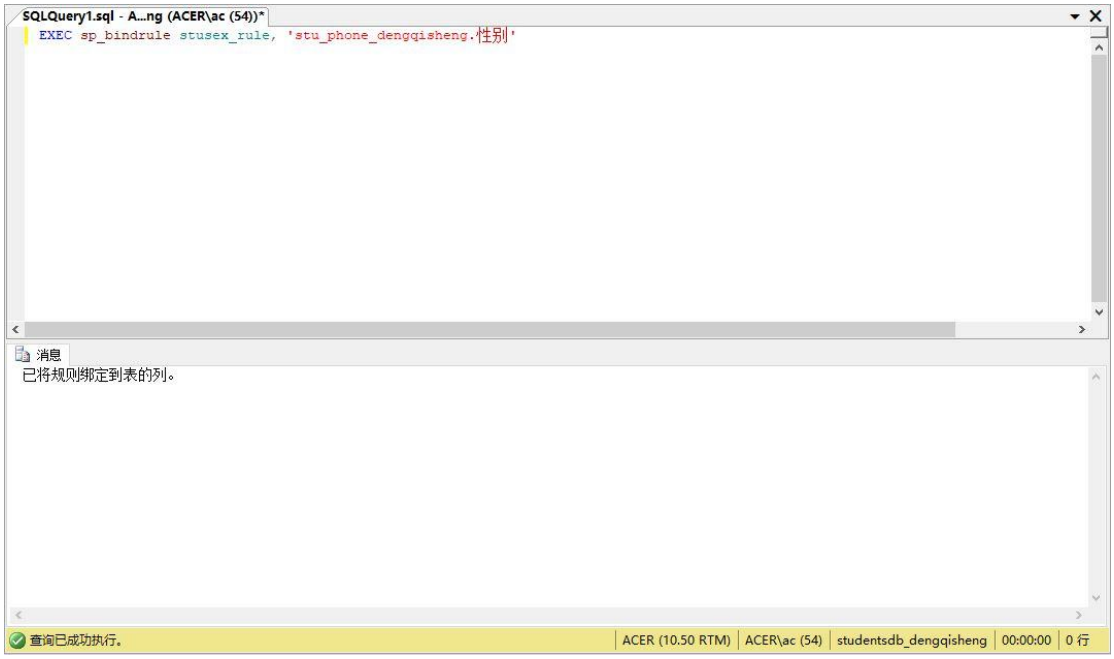

图 1-2-2　绑定 stusex_rule 规则

3. 使用系统存储过程 sp_help 查询 stusex_rule 规则列表，使用系统储存过程 sp_helptext 查询 stusex_rule 规则的文本，将 stusex_rule 规则更名为 stu_s_rule。

查询结果显示如图 1-3。

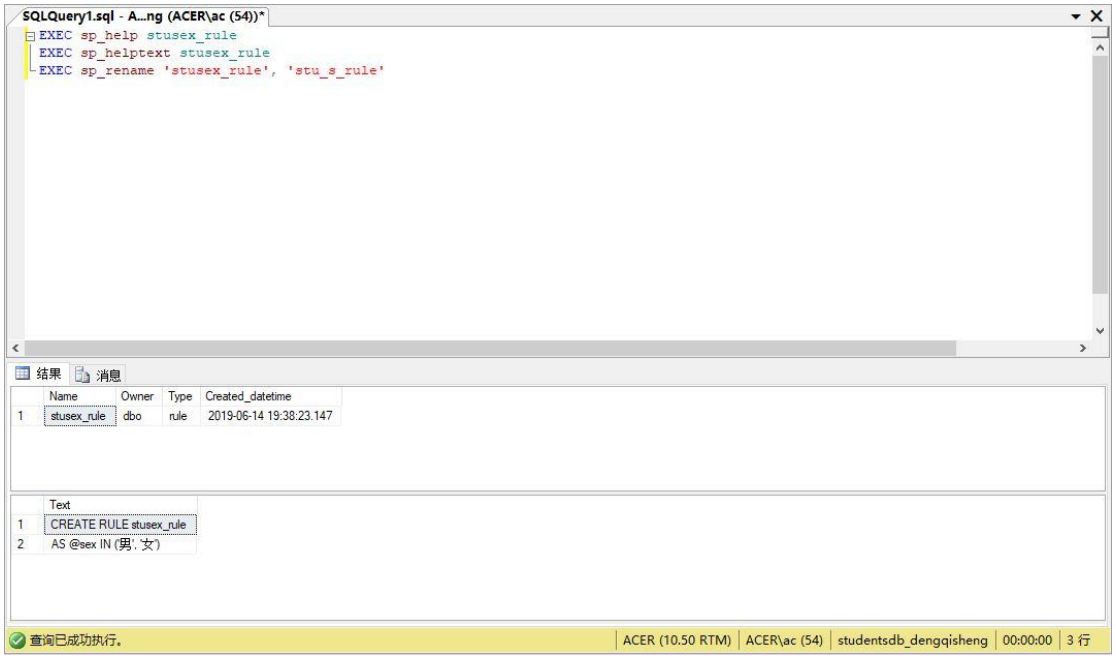

图 1-3　规则的查询与更名

4. 删除规则 stu_s_rule。注意被删除规则是否还被绑定，如果是，如何正确删除？

查询结果显示如图 1-4。

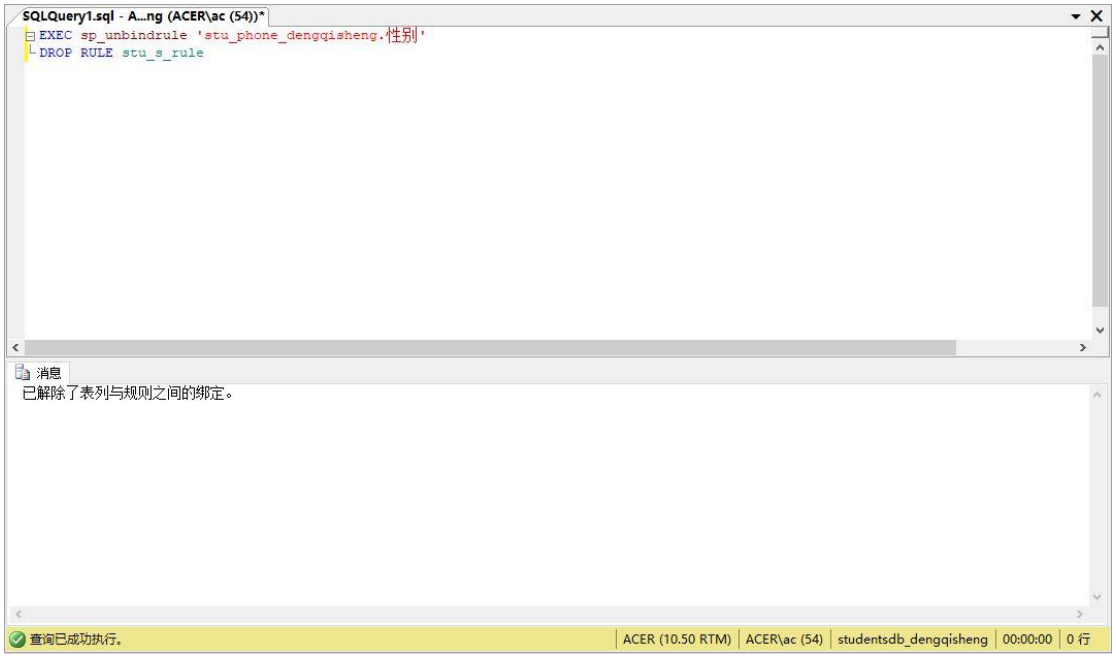

图 1-4　删除 stu_s_rule 规则

5. 为 studentsdb_dengqisheng 数据库建立（1）日期（2）字符（3）货币等类型的默认值对象。

① 在查询设计器中创建默认值对象 df_date, df_char, df_money；

② 在数据库中创建 stu_fee_dengqisheng 数据表；

```
CREATE TABLE stu_fee_dengqisheng
(学号  char(10) NOT NULL,
 姓名  char(8) NOT NULL,
 学费  money,
 交费日期  datetime,
 电话号码  char(7))
```

③ 使用 sp_bindefault 绑定 df_date, df_char, df_money 到 stu_fee_dengqisheng 表格的“学费”、“电话号码”、“交费日期”列上；

④ 输入以下代码，在 stu_fee_dengqisheng 表进行插入操作：

```
INSERT INTO stu_fee_dengqisheng (学号，姓名) VALUES ('0001', '刘卫平')
INSERT INTO stu_fee_dengqisheng (学号，姓名，学费) VALUES ('0001', '张卫民', $120)
INSERT INTO stu_fee_dengqisheng (学号，姓名，学费，交费日期) VALUES ('0001', '马东', $110, '2006-5-12')
```

分析 stu_fee_dengqisheng 表中插入记录的各列的值是什么？

⑤ 解除默认对象 df_char 的绑定并删除对象，类似地再把 df_date, df_money 对象删除。

查询结果显示如图 1-5-1 至图 1-5-5。

图 1-5-1　创建默认值对象

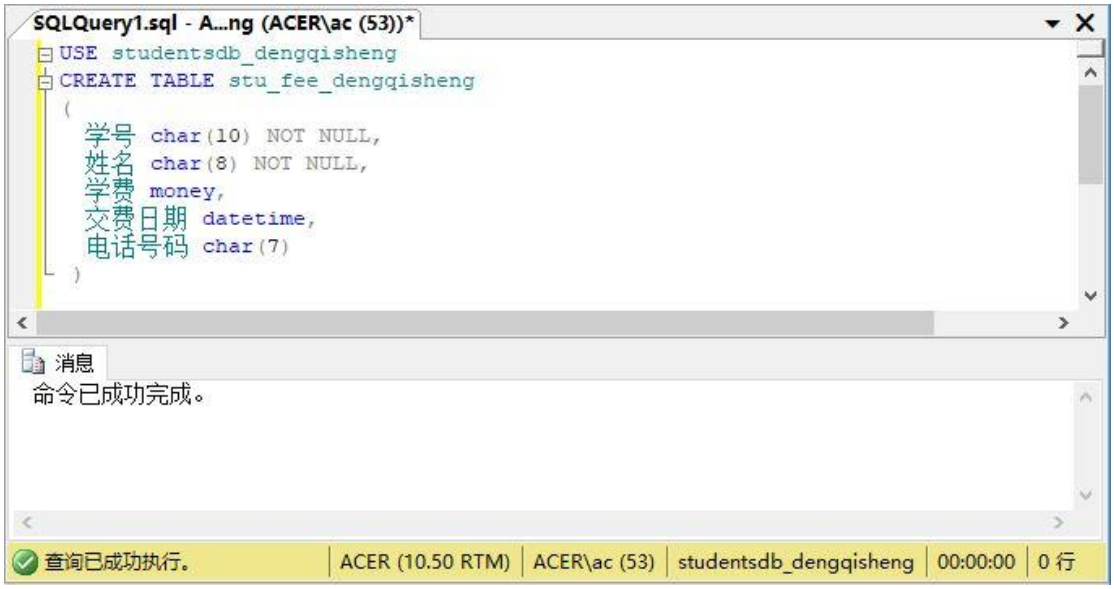

图 1-5-2　创建 stu_fee 数据表

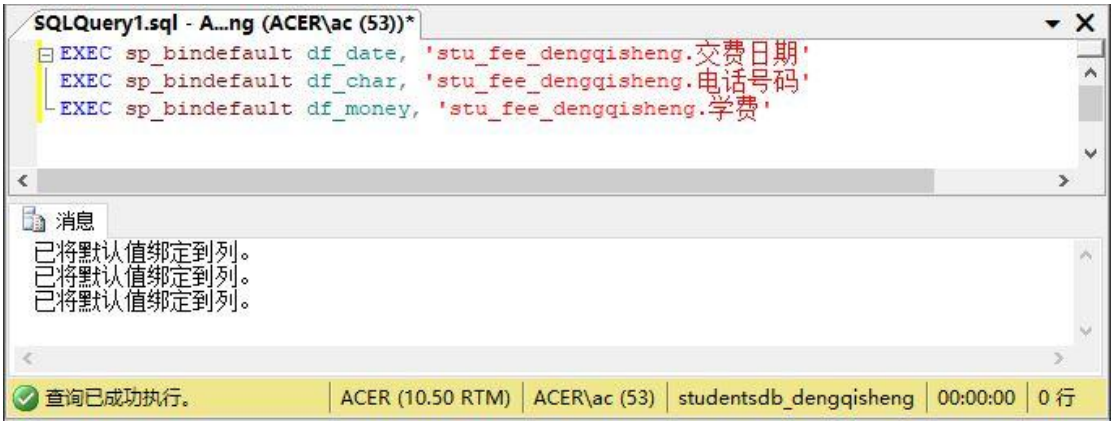

图 1-5-3　绑定默认值对象

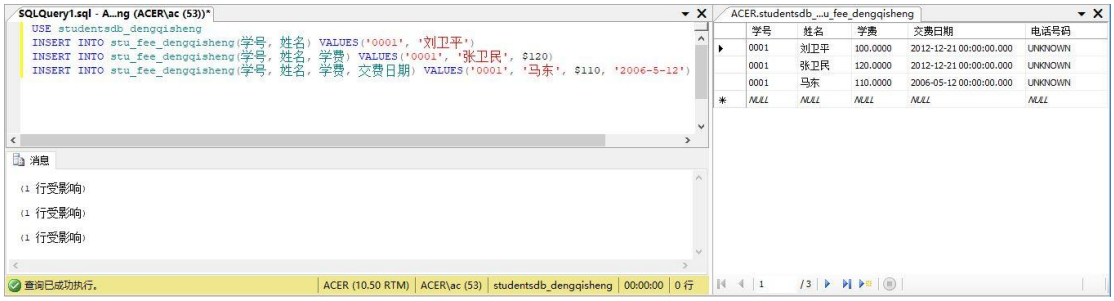

图 1-5-4　插入数据

分析：

由上图得知，插入数据时没有指定具体指的位置被默认值填充。如上所示，第一条记录的学费、交费日期和电话号码均空缺，其值为对应的默认值；第二条记录的交费日期和电话号码以及第三条记录的电话号码空缺，同样地也为默认值。

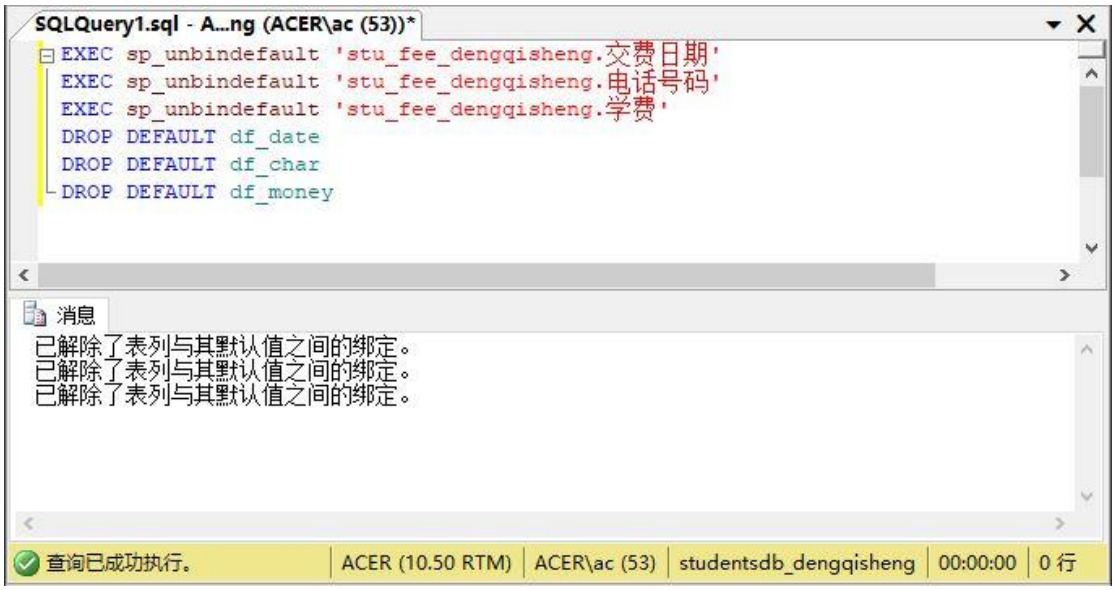

图 1-5-5　解绑并删除默认值对象

6. 为 student_info_dengqisheng 表添加一列，命名为“院系”，创建一个默认值对象 stu_d_df，将其绑定到 student_info_dengqisheng 表的“院系”列上，使其默认值为“信息工程学院”，对 student_info_dengqisheng 表进行插入操作，操作完成后，删除该默认对象。

查询结果显示如图 1-6。

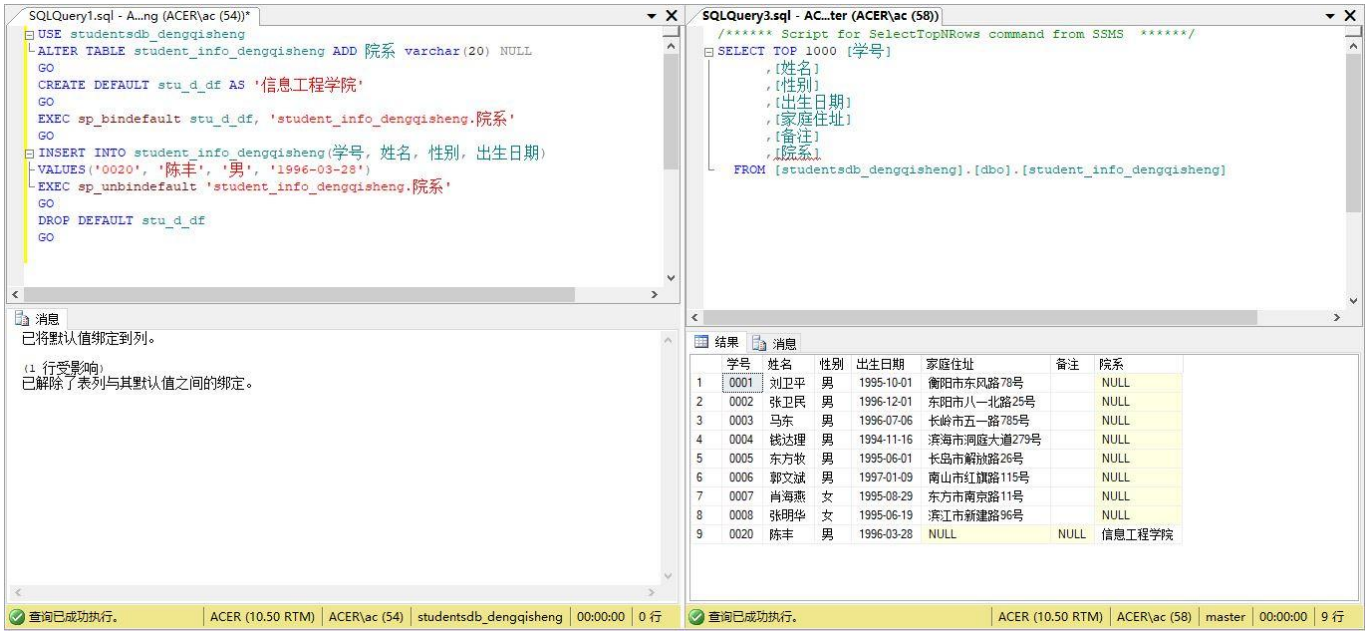

图 1-6　修改学生院系信息

7. 在 studentsdb_dengqisheng 数据库中用 CREATE TABLE 语句创建表 stu_con_dengqisheng，并同时创建约束。

① 创建表的同时创建约束。约束要求为：

将学号设置为主键(PRIMARY KEY)，主键名为 pk_sid；

为姓名添加唯一约束(UNIQUE)，约束名为 uk_name；

为性别添加默认约束(DEFAULT)，默认名称为 df_sex，其值为“男”；

为出生日期添加属性值约束(CHECK)，约束名为 ck_bday，其检查条件为：出生日期 >‘1998-1-1’。

② 在 stu_con_dengqisheng 表中插入数据。分析各约束在插入记录时所起的作用，查看插入记录后表中数据与所插入的数据是否一致？

③ 使用 ALTER TABLE 语句当中的 DROP CONSTRAINT 参数项删除为 stu_con_dengqisheng 表所建的约束。

查询结果显示如图 1-7-1 至图 1-7-3。

图 1-7-1　创建表及约束

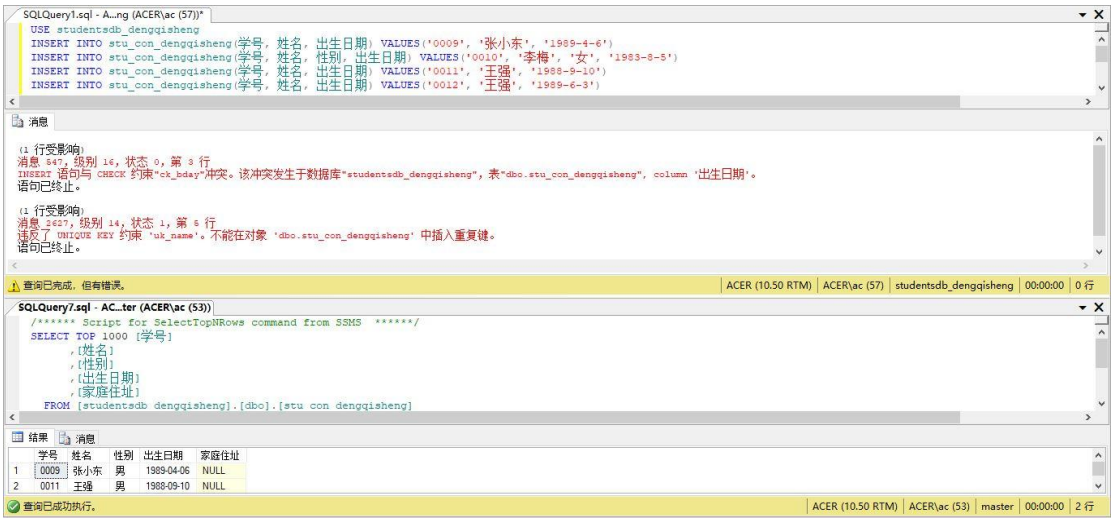

图 1-7-2　向 stu_con_dengqisheng 表中插入数据

分析：

查看查询结果可知，有两条语句违反了既定的约束，导致插入记录后表中的数据与待插入的数据不一致。其中，第二条数据的出生日期违反了检查约束，即出生日期小于“1988-1-1”；而第四条数据的姓名违反了唯一性约束，即姓名“王强”与第三条插入的数据中的姓名“王强”重复。另外，插入第一条和第三条数据时没有指定性别具体的值，故根据默认约束系统自动填充性别为“男”。

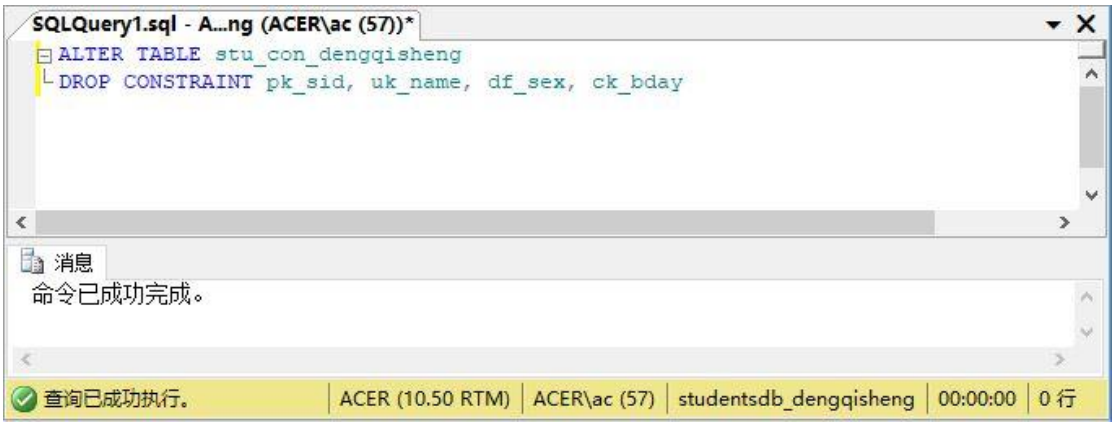

图 1-7-3　删除 stu_con_dengqisheng 表的所有约束

8. 为 studentsdb_dengqisheng 数据库的 grade_dengqisheng 表添加外键约束(FOREIGN KEY)，要求将“学号”设置为外键，参照表为 student_info_dengqisheng，外键名称为 fk_sid。

① 使用系统存储过程 sp_help 查看 grade_dengqisheng 表的外键信息。

② 在 grade_dengqisheng 表中插入一条记录，学号为 0100，课程编号为 0001，分数为 78。观察 SQL Server 会作何处理，为什么？如何解决所产生的问题？

③ 使用查询设计器删除 grade_dengqisheng 表的外键 fk_sid。

查询结果显示如图 1-8-1 至图 1-8-4。

图 1-8-1　添加外键约束

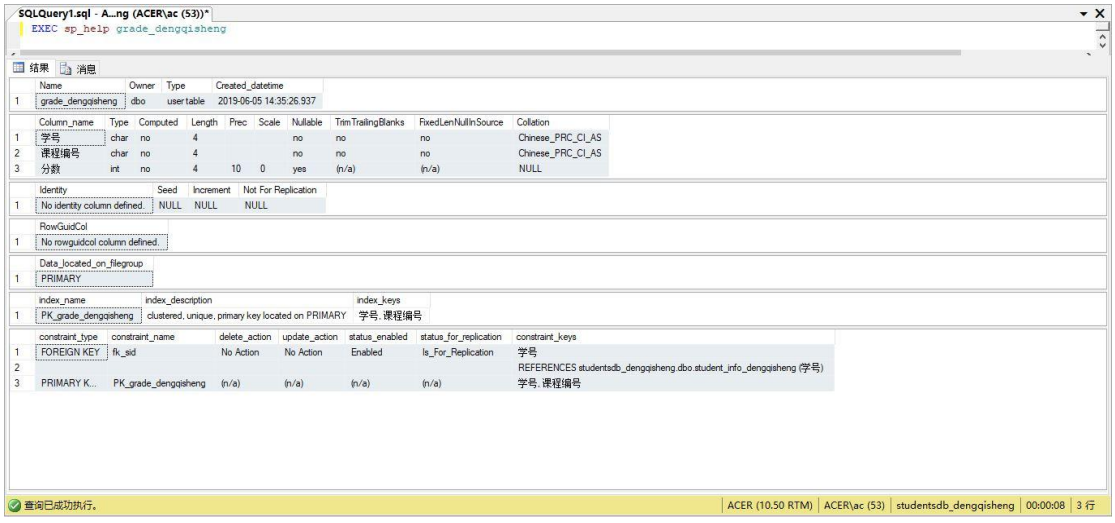

图 1-8-2　查看外键信息

图 1-8-3　插入数据记录

分析：

若在 grade_dengqisheng 表中插入学号为“0100”的记录，查询结果会显示与外键约束冲突，这是因为 student_info_dengqisheng 表中不存在学号为“0100”的学生，所以导致了外键约束冲突，故应将待插入记录中的学号值修改为已在 student_info_dengqisheng 表中存在的学号值，例如“0004”。修改后，查询结果显示插入成功。

图 1-8-4　删除外键约束

实验二：存储过程和触发器

一、实验目的

1. 掌握通过 SQL Server 管理平台和 T-SQL 语句 CREATE PROCEDURE 创建存储过程的方法和步骤。

2. 掌握使用 T-SQL 语句 EXECUTE 执行存储过程的方法。

3. 掌握通过 SQL Server 管理平台和 T-SQL 语句 ALTER PROCEDURE 修改存储过程的方法。

4. 掌握通过 SQL Server 管理平台和 T-SQL 语句 DROP PROCEDURE 删除存储过程的方法。

5. 掌握通过 SQL Server 管理平台和 T-SQL 语句 CREATE TRIGGER 创建触发器的方法和步骤。

6. 掌握引发触发器的方法。

7. 掌握使用 SQL Server 管理平台或 T-SQL 语句修改和删除触发器。

8. 掌握事务、命名事务的创建方法，了解不同类型的事务的处理情况。

二、实验环境

软件配置：Microsoft SQL Server 2008 R2

操作平台：Windows 10

系统类型：64 位操作系统

三、实验内容

1. 输入以下 T-SQL 代码,创建一个利用流控制语句的存储过程 letters_print，该存储过程能够显示 26 个小写字母。

```
CREATE PROCEDURE letters_print
AS
DECLARE @count int
SET @count = 0
```

```
WHILE @count < 26
BEGIN
    PRINT CHAR(ASCII('a') + @count)
    SET @count = @count + 1
END
```

使用 EXECUTE 命令执行 letters_print 存储过程。

查询结果显示如图 2-1-1 和图 2-1-2。

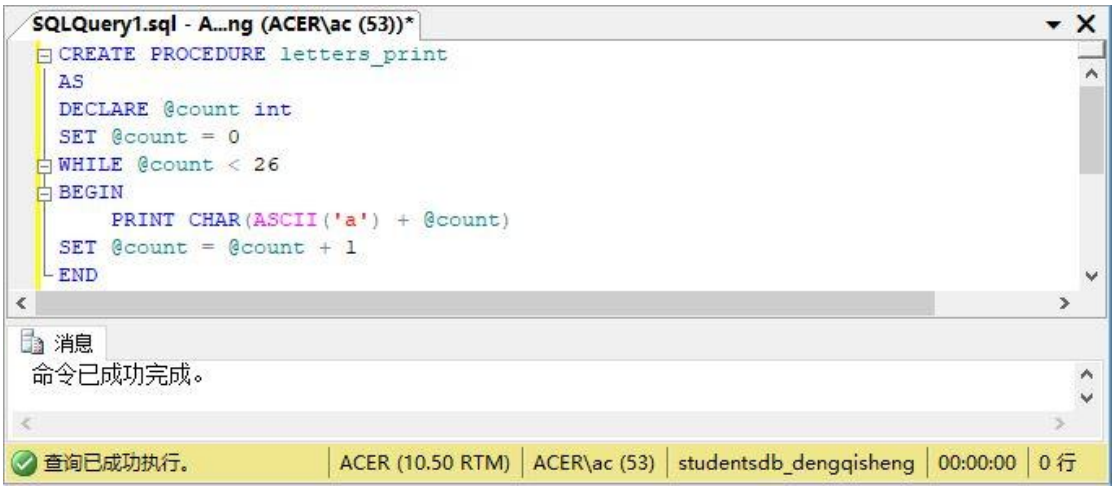

图 2-1-1　创建存储过程

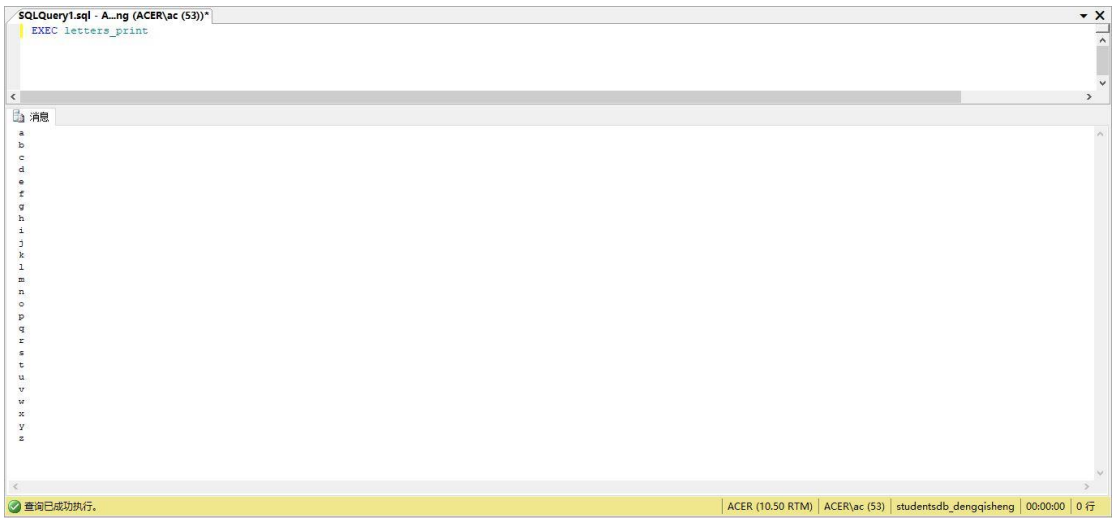

图 2-1-2　执行存储过程

2. 输入以下 T-SQL 代码，创建存储过程 stu_info，执行时通过输入姓名，可以查询该姓名对应的学生的各科成绩。

```
CREATE PROCEDURE stu_info @name varchar(40)
AS
SELECT a.学号，姓名，课程编号，分数
FROM student_info_dengqisheng a INNER JOIN grade_dengqisheng ta
ON a.学号 = ta.学号
WHERE 姓名 = @name
```

使用 EXECUTE 命令执行存储过程 stu_info，其参数值为“马东”。如果存储过程 stu_info 执行时没有提供参数，但要求能按默认值查询（设姓名为“刘卫平”），需要如何修改该过程的定义？

查询结果显示如图 2-2-1 至图 2-2-4。

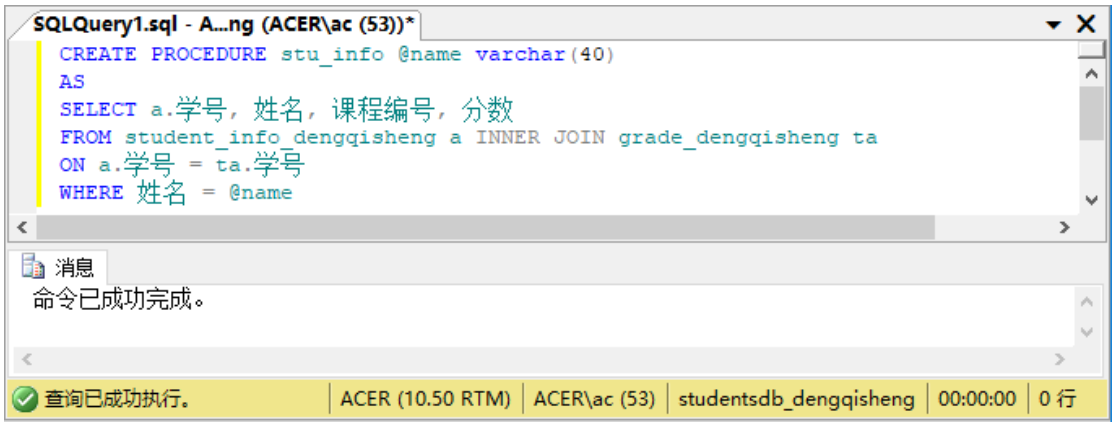

图 2-2-1　创建存储过程

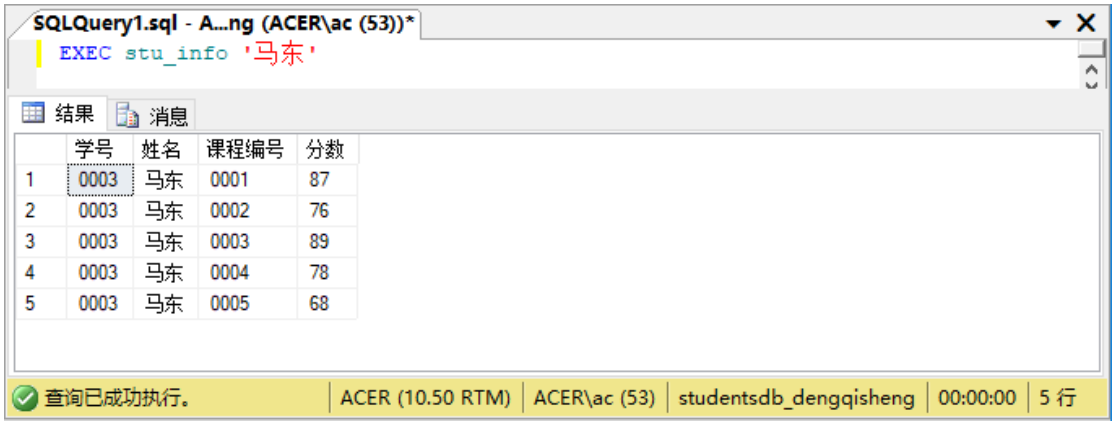

图 2-2-2　执行存储过程

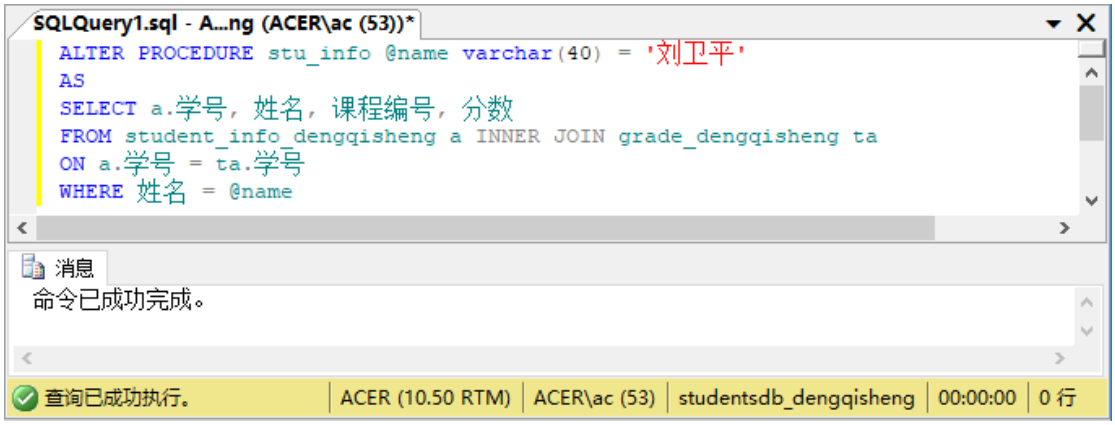

图 2-2-3　修改存储过程

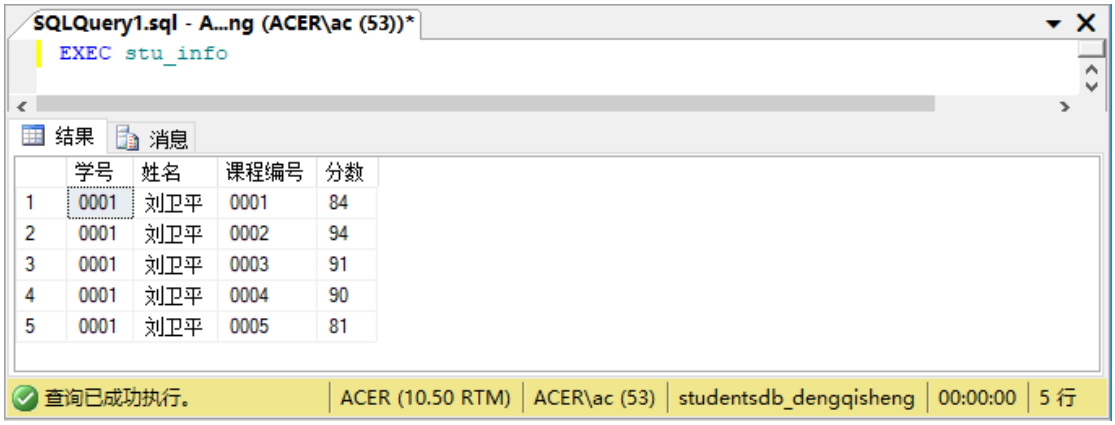

图 2-2-4　再次执行存储过程

3. 使用 studentsdb_dengqisheng 数据库中的 student_info_dengqisheng 表、curriculum_dengqisheng 表、grade_dengqisheng 表。

① 创建存储过程 stu_grade，查询学号为 0001 的学生姓名、课程名称、分数。

② 执行存储过程 stu_grade，查询 0001 学生的姓名、课程名称、分数。

③ 使用系统存储过程 sp_rename 将存储过程 stu_grade 更名为 stu_g。

查询结果显示如图 2-3-1 至图 2-3-3。

图 2-3-1　创建存储过程

图 2-3-2　执行存储过程

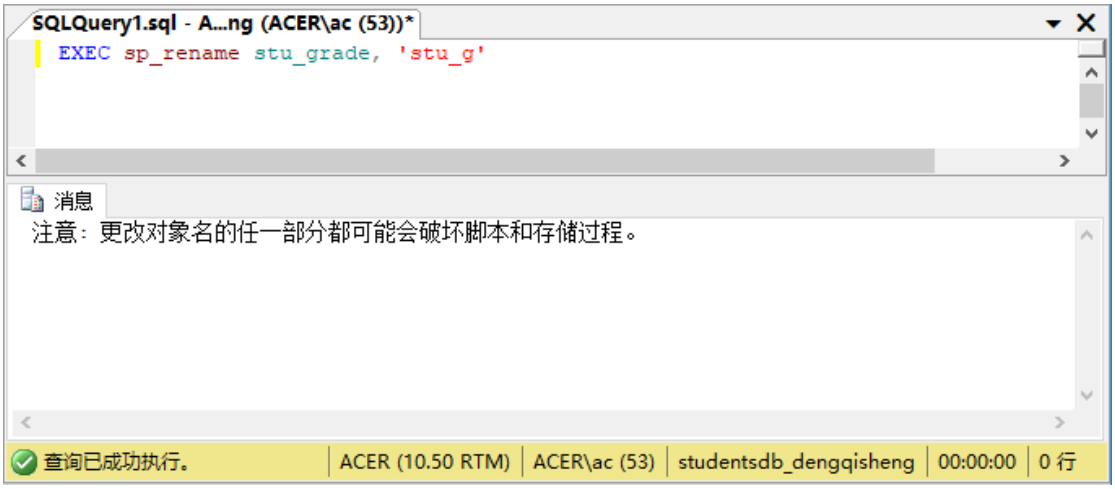

图 2-3-3　更名存储过程

4. 使用 studentsdb_dengqisheng 数据库中的 student_info_dengqisheng 表、curriculum_dengqisheng 表、grade_dengqisheng 表。

① 创建一个带参数的存储过程 stu_g_p，当任意输入一个学生的姓名时，将从 3 个表中返回该学生的学号、选修的课程名称和课程成绩。

② 执行存储过程 stu_g_p，查询“刘卫平”的学号、选修课程和课程成绩。

③ 使用系统存储过程 sp_helptext，查看存储过程 stu_g_p 的文本信息。

查询结果显示如图 2-4-1 至图 2-4-3。

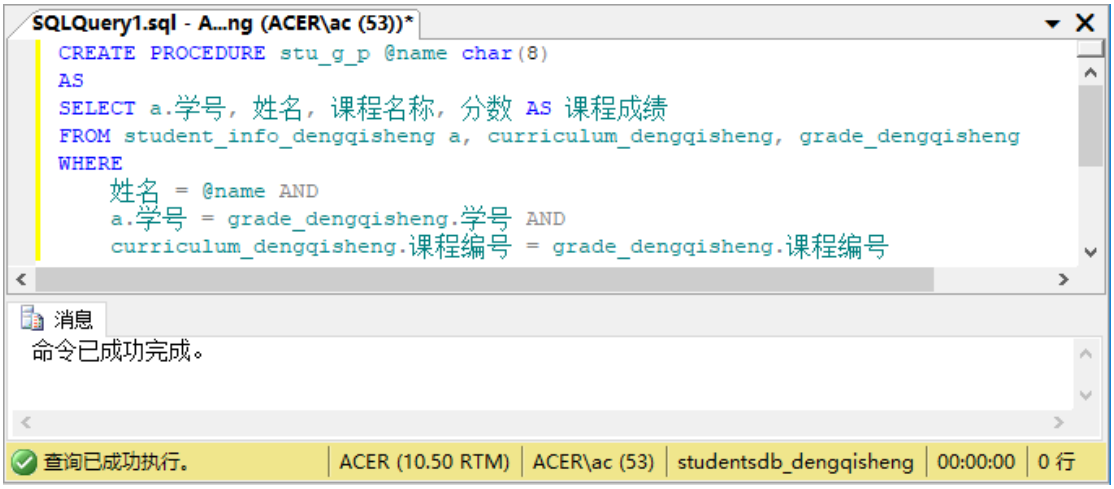

图 2-4-1　创建存储过程

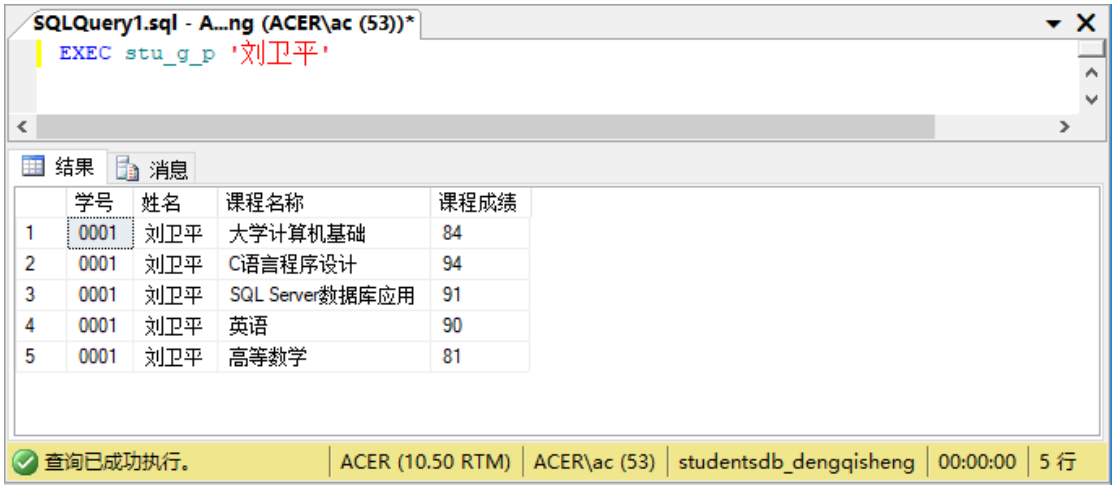

图 2-4-2　执行存储过程

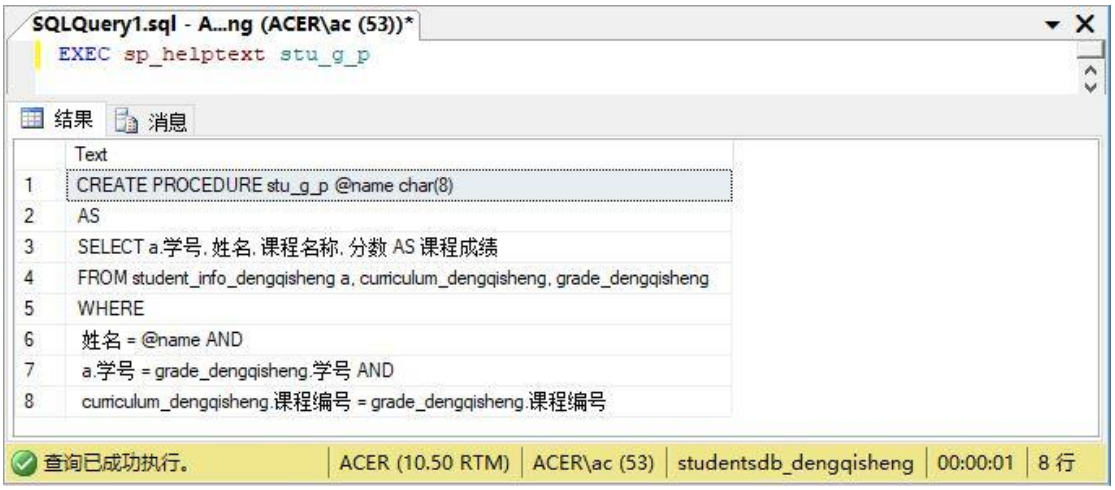

图 2-4-3　查看存储过程文本信息

5. 使用 studentsdb_dengqisheng 数据库中的 student_info_dengqisheng 表。

① 创建一个加密的存储过程 stu_en，查询所有男学生的信息。

② 执行存储过程 stu_en，查看返回学生的情况。

③ 使用 T-SQL 语句 DROP PROCEDURE 删除存储过程 stu_en。

查询结果显示如图 2-5-1 至图 2-5-3。

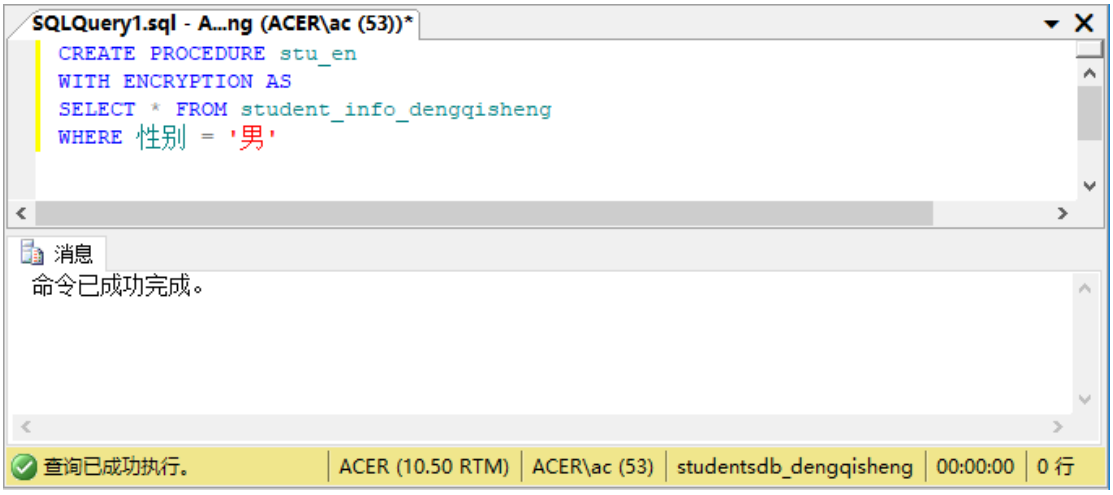

图 2-5-1　创建存储过程

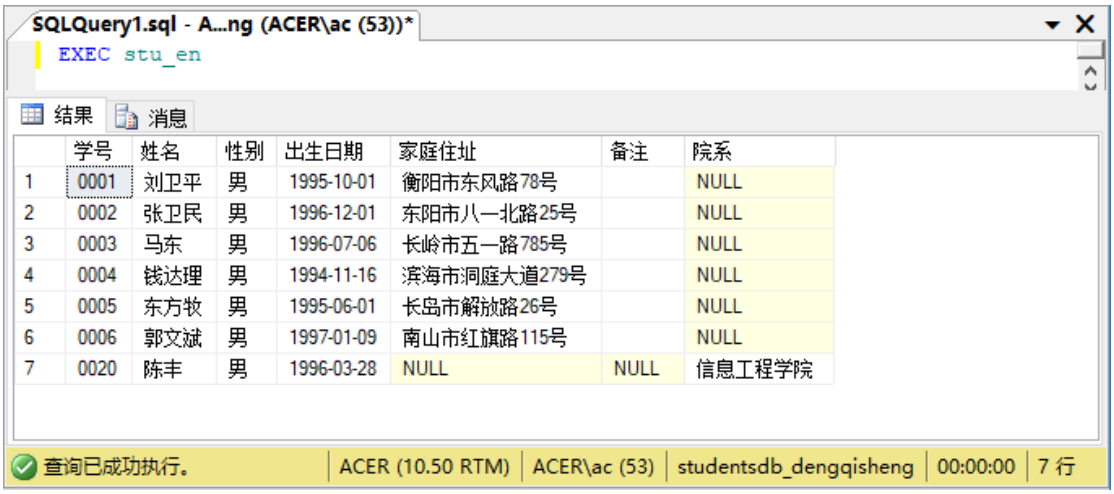

图 2-5-2　执行存储过程

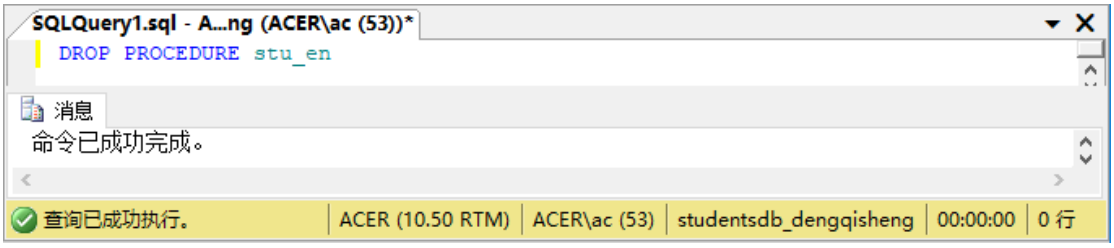

图 2-5-3　删除存储过程

6. 使用 studentsdb_dengqisheng 数据库中的 grade_dengqisheng 表。

① 创建一个存储过程 stu_g_r，当输入一个学生的学号时，通过返回输出参数获取该学生各门课程的平均成绩。

② 执行存储过程 stu_g_r，输入学号 0002。

③ 显示 0002 号学生的平均成绩。

查询结果显示如图 2-6-1 和图 2-6-2。

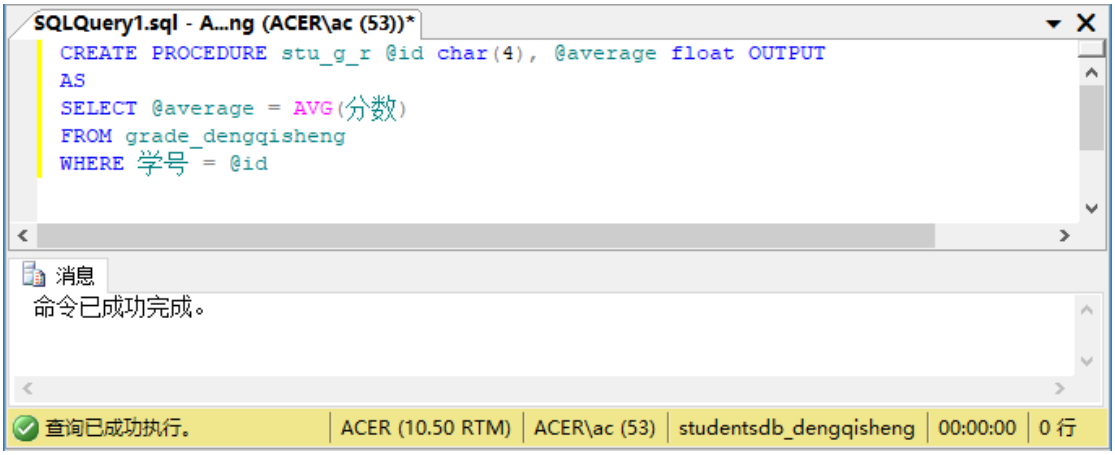

图 2-6-1　创建存储过程

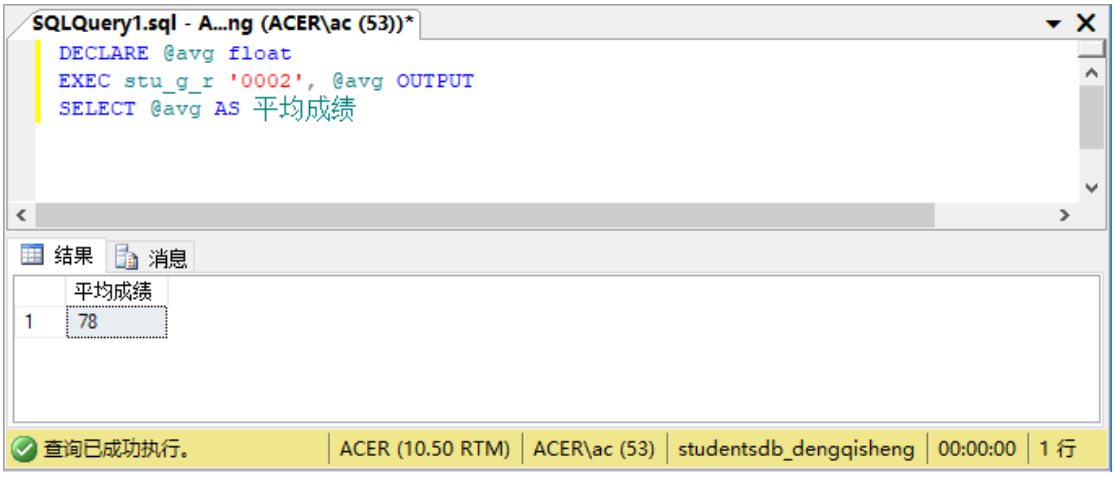

图 2-6-2　执行存储过程并显示平均成绩（注：成绩数据经过上机实验 3 修改）

7. 输入以下代码，复制 student_info_dengqisheng 表命名为 stu2_dengqisheng，为 stu2_dengqisheng 表创建一个触发器 stu_tr，当 stu2_dengqisheng 表插入一条记录时，为该记录生成一个学号，该学号为学号列数据的最大值加 1。

--复制 student_info_dengqisheng 表命名为 stu2_dengqisheng

SELECT * INTO stu2_dengqisheng FROM student_info_dengqisheng

GO

--为 stu2_dengqisheng 表创建一个 INSERT 型触发器 stu_tr

CREATE TRIGGER stu_tr

ON stu2_dengqisheng FOR INSERT

AS

DECLARE @max char(4)

SET @max = (SELECT MAX(学号) FROM stu2_dengqisheng)

SET @max = @max + 1

UPDATE stu2_dengqisheng SET 学号 = REPLICATE('0', 4 – LEN(@max)) + @max

FROM stu2_dengqisheng INNER JOIN inserted

ON stu2_dengqisheng.学号 = inserted.学号

执行以上代码，查看 studentsdb_dengqisheng 数据库中是否有 stu2_dengqisheng 表，展开 stu2_dengqisheng 表，查看其触发器项中是否有 stu_tr 触发器。在查询编辑窗口输入以下代码：

INSERT INTO stu2_dengqisheng(学号，姓名，性别) VALUES('0001', '张主', '女')

运行以上代码，查看 stu2_dengqisheng 表的变化情况，为什么插入记录的学号值发生了改变？

查询结果显示如图 2-7-1 至图 2-7-3。

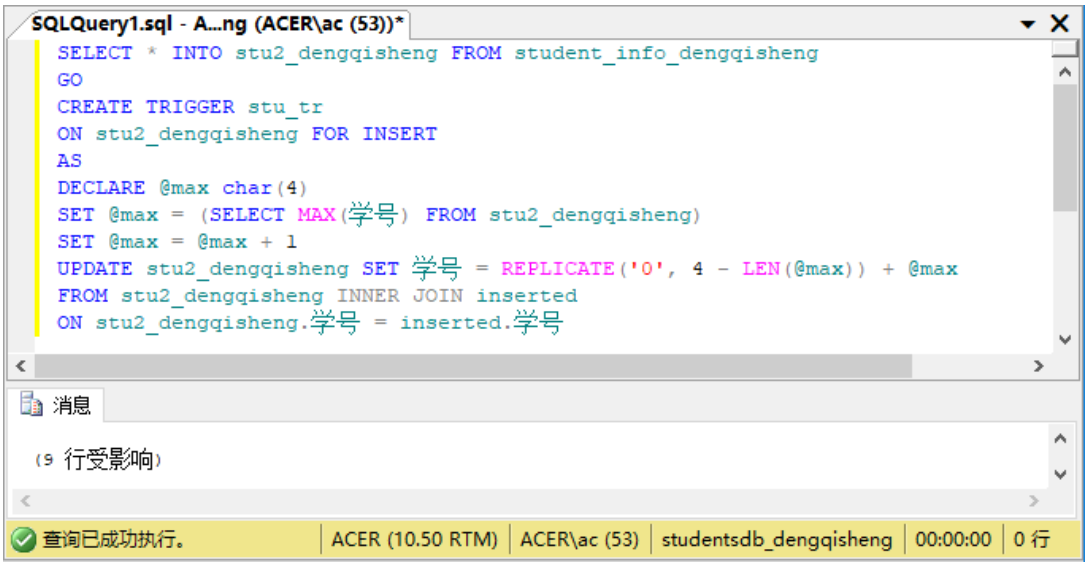

图 2-7-1　复制数据表并创建触发器

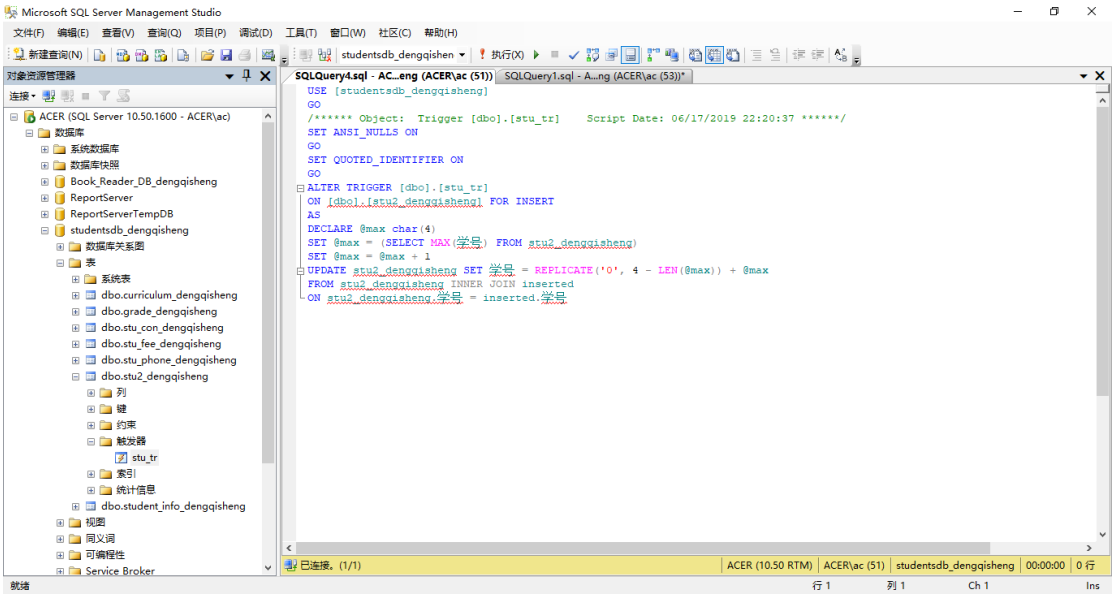

图 2-7-2　查看触发器

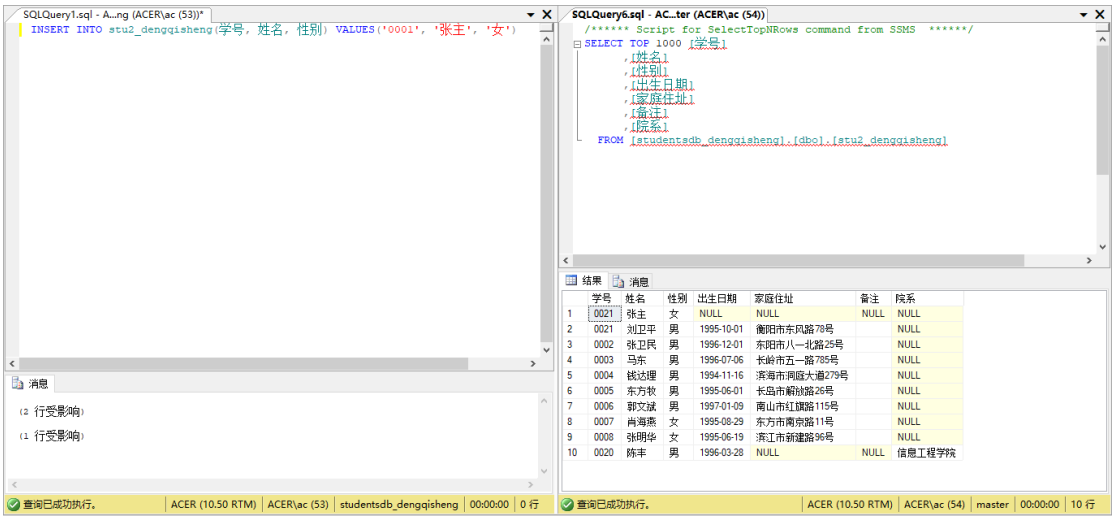

图 2-7-3　查看触发器

分析：

观察查询结果发现第一行和第二行的“学号”发生了改变，均从“0001”变为了“0021”。分析触发器的内容，由于学号均为 4 位，REPLICATE 函数返回 NULL 值，UPDATE 行将指定学号修改为@max 的值，而两行 SET 语句设定了@max 的值为学号最大值加 1，最大值为“0020”，故学号值应变更为“0021”。最后一行说明了指定的数据是 stu2_dengqisheng 表和 inserted 表中学号相同的记录，而 inserted 表的数据应只含有编辑器中待插入的新数据，故学号为“0001”的数据会被修改，所以最终导致学号为“0001”的数据学号被更改为“0021”。

8. 为 grade_dengqisheng 表建立一个名为 insert_g_tr 的 INSERT 触发器，当用户向 grade_dengqisheng 表中插入记录时，若插入的是在 curriculum_dengqisheng 表中没有的课程编号，则提示用户不能插入记录，否则提示记录插入成功。使用给定数据进行插入测试，观察插入数据时的运行情况，说明为什么？

查询结果显示如图 2-8-1 至图 2-8-3。

图 2-8-1　创建插入触发器

图 2-8-2　插入第一条数据

图 2-8-3　插入第二条数据

分析：

观察结果可知，插入不存在的课程编号会触发事务回滚导致插入失败。

9. 为 curriculum_dengqisheng 表创建一个名为 del_c_tr 的 DELETE 触发器，该触发器的作用是禁止删除 curriculum_dengqisheng 表中的记录。

查询结果显示如图 2-9。

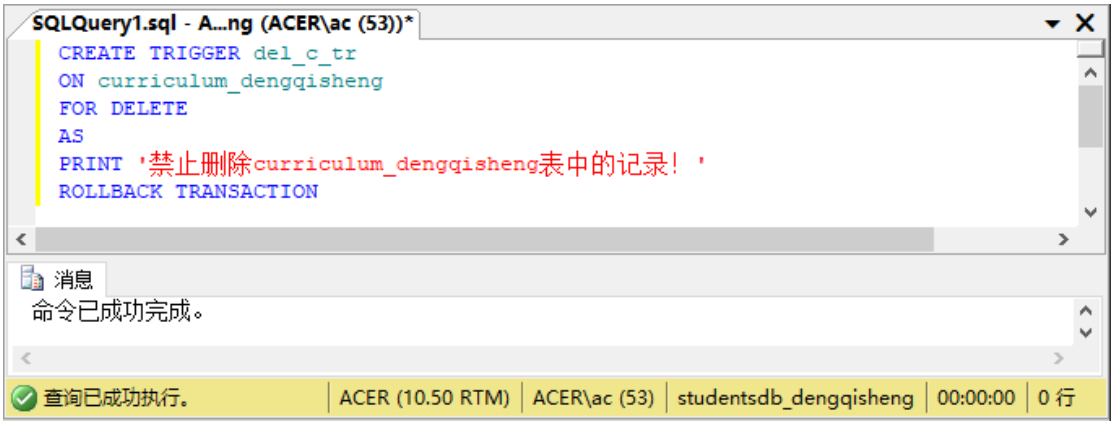

图 2-9　创建删除触发器

10. 为 student_info_dengqisheng 表创建一个名为 update_s_tr 的 UPDATE 触发器，该触发器的作用是禁止更新 student_info_dengqisheng 表中的“姓名”字段的内容。

查询结果显示如图 2-10。

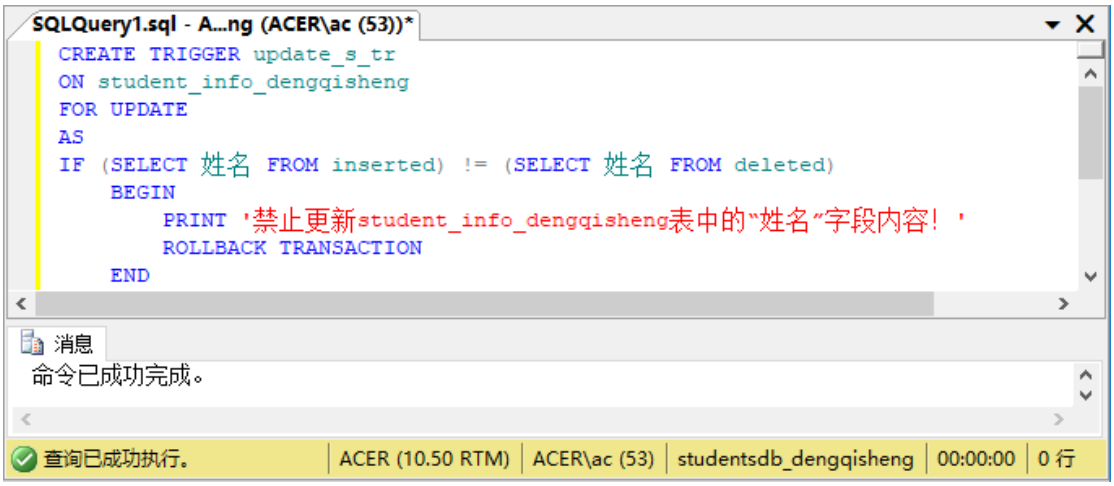

图 2-10　创建更新触发器

11. 使用 T-SQL 语句 DROP TRIGGER 删除 update_s_tr 触发器。

查询结果显示如图 2-11。

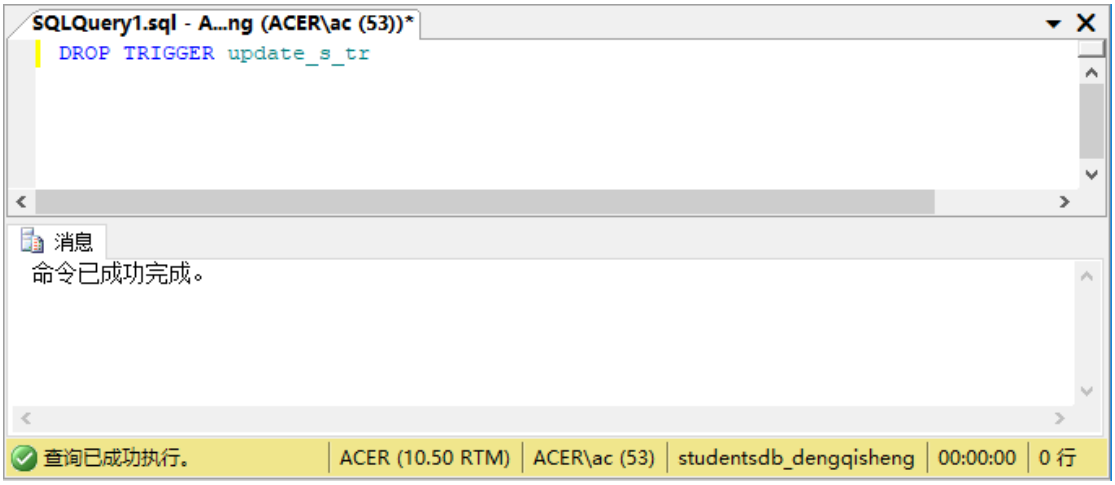

图 2-11　删除触发器

12. 为 student_info_dengqisheng 表建立删除触发器 del_s_tr，要求当 student_info_dengqisheng 表的记录被删除后，grade_dengqisheng 表中相应的记录也能自动删除。

查询结果显示如图 2-12。

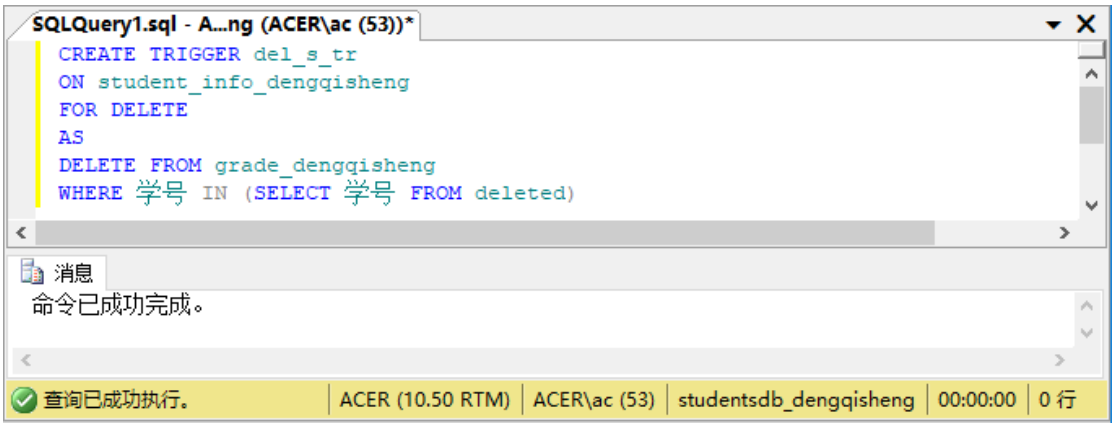

图 2-12　自动删除相应记录

13. 在 studentsdb_dengqisheng 数据库中，执行以下事务处理过程，说明这些事务属于哪一种事务类型（隐性事务、显性事务或自动式事务）。

①

```
BEGIN TRANSACTION
INSERT INTO student_info_dengqisheng(学号，姓名) VALUES('0009', '李青')
COMMIT TRANSACTION
```

②

```
SET IMPLICIT_TRANSACTIONS ON
GO
INSERT INTO grade_dengqisheng(学号，课程编号) VALUES('0005', '0007')
GO
IF((SELECT count(*)FROM curriculum_dengqisheng WHERE 课程编号='0007')=0)
ROLLBACK TRANSACTION
ELSE
COMMIT TRANSACTION
SET IMPLICIT_TRANSACTIONS OFF
```

SET IMPLICIT_TRANSACTIONS ON 的作用是什么？这里的事务由哪个语句启动？分析 IF 语句的功能，在 grade_dengqisheng 表中插入“课程编号”的值为“0007”时，执行哪个事务管理语句（ROLLBACK TRANSACTION 还是 COMMIT TRANSACTION）？如果“课程编号”的值为“0003”时，情况又如何？

③ 在①和②的基础上，执行以下事务：

```
INSERT INTO student_info_dengqisheng(学号，姓名) VALUES('0009', '王晶')
GO
```

该事务能否完成，为什么？

查询结果显示如图 2-13-1 至图 2-13-4。

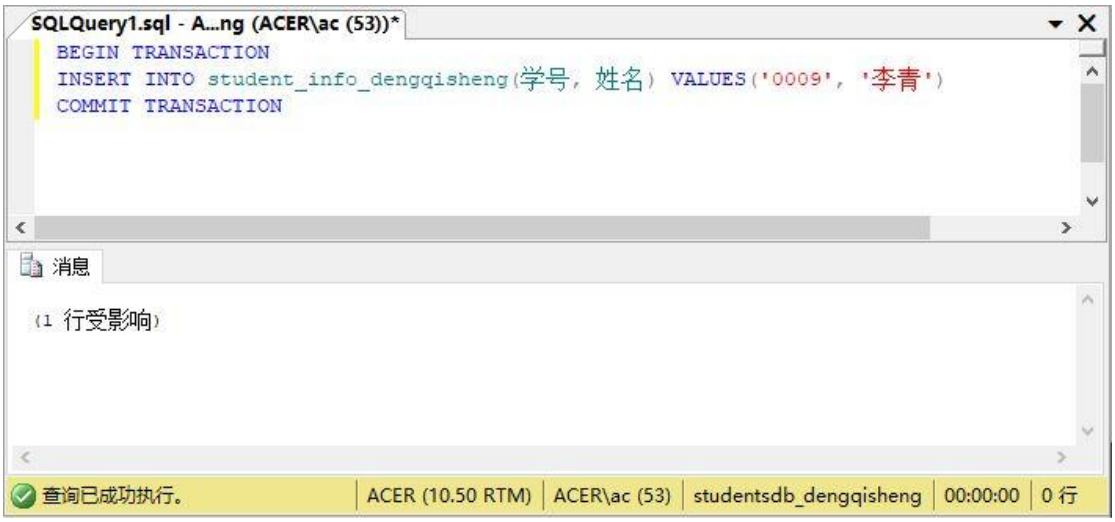

图 2-13-1　显性事务

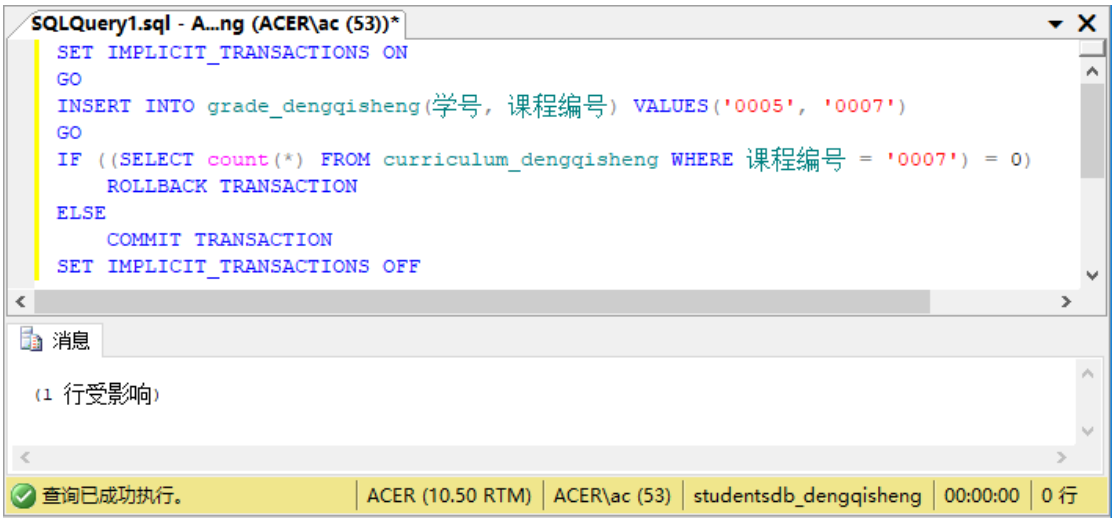

图 2-13-2　隐性事务 1

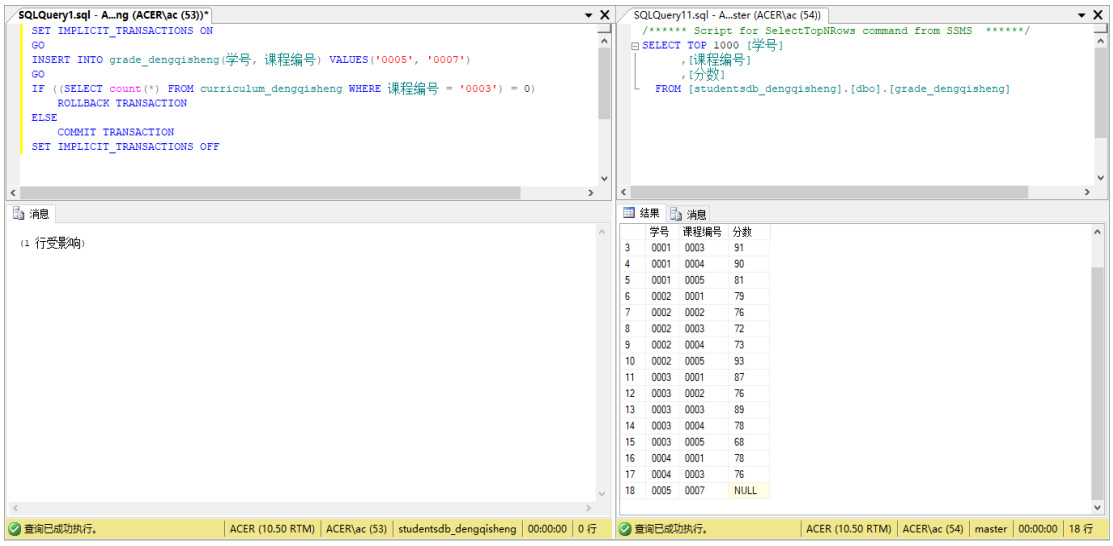

图 2-13-3　隐性事务 2

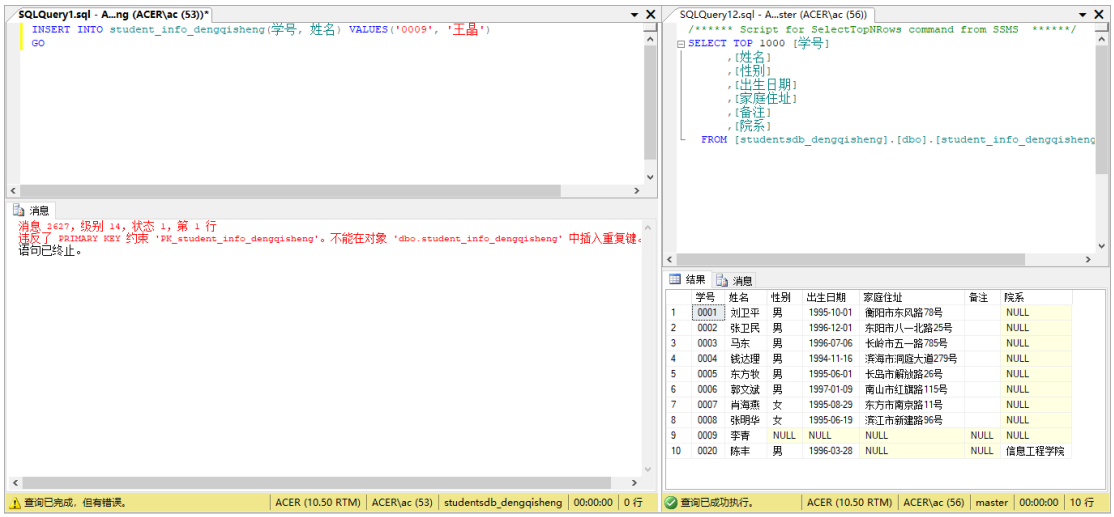

图 2-13-4　自动式事务

分析：

以上三题中，第 1 小题为显性事务，第 2 小题为隐性事务，第 3 小题为自动式事务。第 2 小题中的 SET IMPLICIT_TRANSACTIONS 起到启动隐性事务的作用，该隐性事务由插入语句 INSERT INTO 启动。分析 IF 语句的作用，当插入“课程编号”的值为“0007”时，由于 curriculum_dengqisheng 表中不存在“课程编号”为“0007”的记录，故必定会执行第一个分支，即执行事务管理语句 ROLLBACK TRANSACTION 进行事务回滚，使得插入失败。只要判断条件不改变，课程编号“0007”则一直不存在，向 grade_dengqisheng 表插入“课程编号”的值为“0003”的记录时仍然会执行 ROLLBACK TRANSACTION 语句。若更改判断条件为“0003”，插入数据“0007”时则会执行 COMMIT TRANSACTION 语句，使得插入成功。

第 3 小题中的事务如结果所示不能完成，这是因为待插入的数据违反了 student_info_dengqisheng 表的主键唯一性约束，故插入不能完成，事务自动回滚至执行语句前的状态。

14. 分析以下嵌套事务处理过程具有几级嵌套，每级嵌套的事务名称是什么？

```
BEGIN TRANSACTION outertran
INSERT INTO student_info_dengqisheng(学号，姓名) VALUES('0010', '王晶')
    BEGIN TRANSACTION innertran1
```

SELECT @@TRANCOUNT

INSERT INTO student_info_dengqisheng(学号，姓名) VALUES('0011', '张磊')

BEGIN TRANSACTION innertran2

SELECT @@TRANCOUNT

INSERT INTO student_info_dengqisheng(学号，姓名) VALUES('0012', '陈进')

COMMIT TRANSACTION innertran2

COMMIT TRANSACTION innertran1

COMMIT TRANSACTION outertran

说明在每个 BEGIN TRANSACTION 语句和 COMMIT TRANSACTION 语句块中当前事务数@@TRANCOUNT 的值是多少？

查询结果显示如图 2-14。

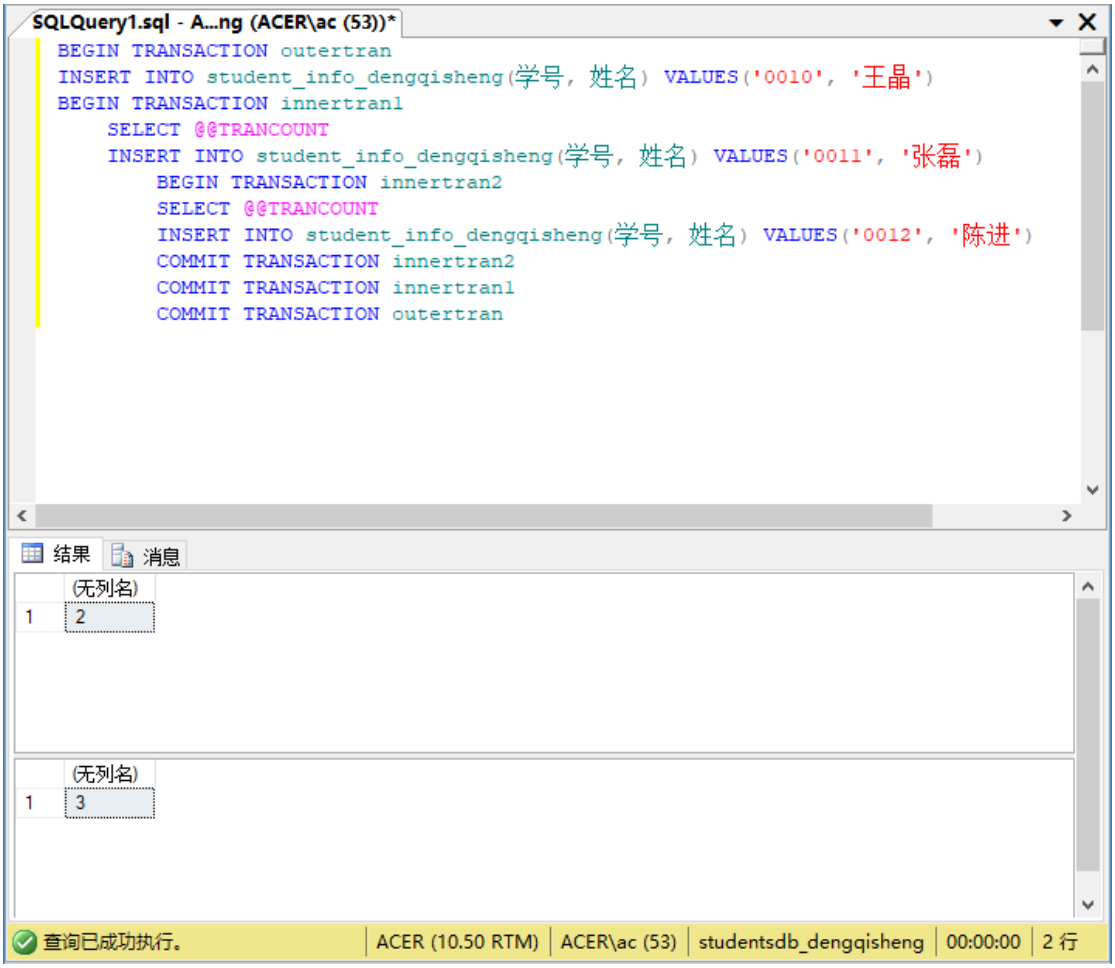

图 2-14　嵌套事务处理过程

分析：

该嵌套事务处理过程一共有三个 BEGIN TRANSACTION 语句和三个 COMMIT TRANSACTION 语句，其事务数@@TRANCOUNT 分别为：

```
BEGIN TRANSACTION outertran        -- @TRANCOUNT 为 1
BEGIN TRANSACTION innertran1       -- @TRANCOUNT 为 2
BEGIN TRANSACTION innertran2       -- @TRANCOUNT 为 3
COMMIT TRANSACTION innertran2      -- @TRANCOUNT 为 2
COMMIT TRANSACTION innertran1      -- @TRANCOUNT 为 1
COMMIT TRANSACTION outertran       -- @TRANCOUNT 为 0
```

15. 设计一个“选课”事务，每选一门课程，总学分增加该课程的学分数，如果所选课程的总学分>10，则选择的课程取消，即事务回滚。

① 在 studentsdb_dengqisheng 数据库中，从 student_info_dengqisheng 表中复制表，并从 stu_ch_dengqisheng 表添加一列，命名为“总学分”。

```
SELECT 学号, 姓名, 性别 INTO stu_ch_dengqisheng
FROM student_info_dengqisheng
ALTER TABLE stu_ch_dengqisheng
ADD 总学分 int
GO
```

② 建立一个命名事务 ch_c，当学号为@sid 的学生所选的课程（课程编号为@cid）的总学分没有超过 10 时，将学号和课程编号值（@sid, @cid）添加到 grade 表中，同时修改 stu_ch_dengqisheng 表中的总学分，使总学分为当前总学分加上所选课程的学分值（@c_h）。否则，取消该事务，实现回滚。

```
DECLARE @sid char(4), @cid char(4), @c_h int
SET @sid = '0004'   --学号为 0004
SET @cid = '0003'   --课程编号为 0003
SET @c_h = (SELECT 学分 FROM curriculum_dengqisheng
```

```
                WHERE 课程编号 = @cid)
--@c_h 是课程编号为@cid 的课程的学分值
BEGIN TRANSACTION ch_c
INSERT INTO grade_dengqisheng(学号，课程编号) VALUES(@sid, @cid)
UPDATE stu_ch_dengqisheng SET 总学分 = 总学分 + @c_h
WHERE 学号 = @sid
--使 stu_ch_dengqisheng 表的总学分列的值加@c_h
IF ((SELECT 总学分 FROM stu_ch_dengqisheng WHERE 学号 = @sid) > 10)
--判断学号为@sid 的学生总学分是否大于 10
BEGIN
    ROLLBACK TRANSACTION ch_c
    --是，则回滚，取消 INSERT 和 UPDATE 操作
    PRINT '总学分超过 10'
END
ELSE
COMMIT TRANSACTION ch_c
```

③ 连续执行事务 ch_c，每次执行给@cid 的赋值分别为'0003'，'0004'，'0005'，'0001'，观察事务 ch_c 处理结果及 stu_ch_dengqisheng 表与 grade_dengqisheng 表的变化。比较@cid 值为'0001'时，与其他取值执行时的不同结果，为什么？

查询结果显示如图 2-15-1 至图 2-15-5。

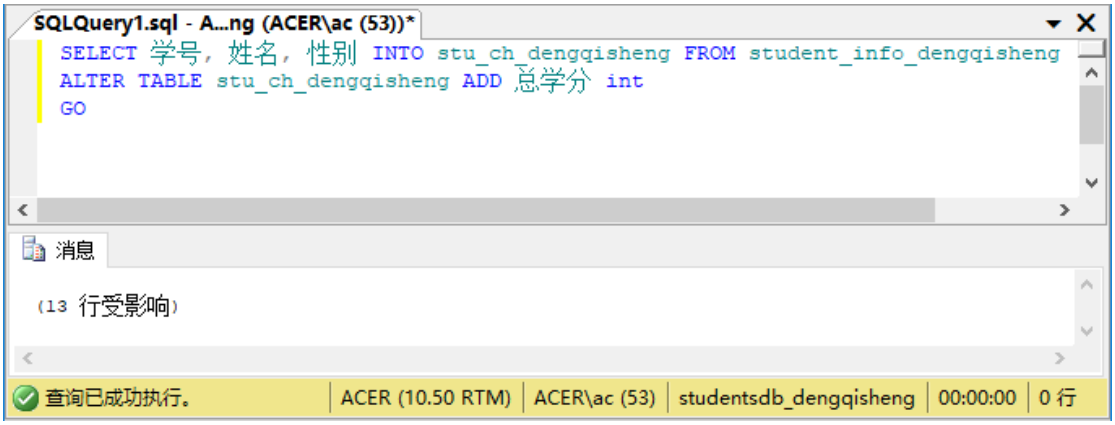

图 2-15-1　复制学生信息表并添加总学分列

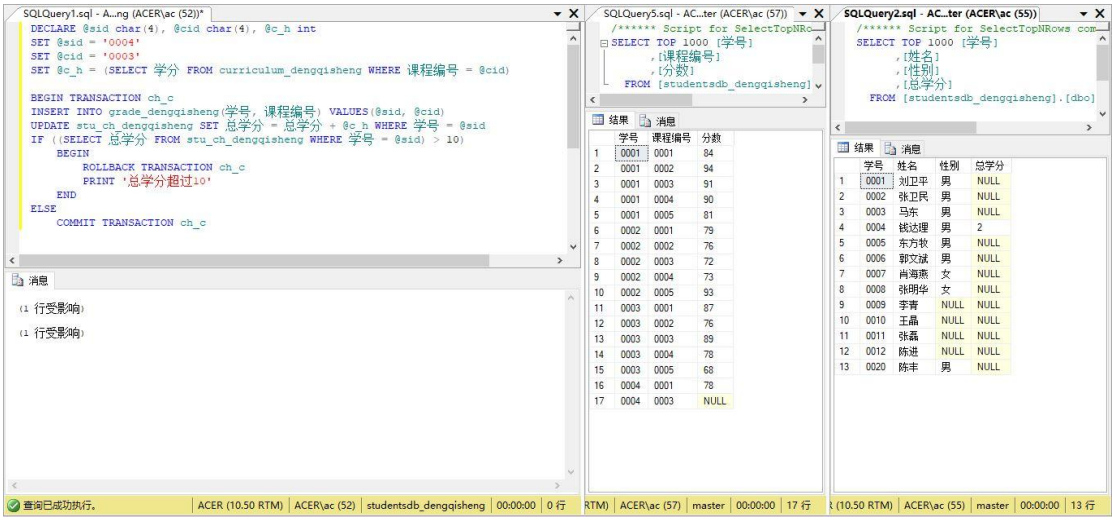

图 2-15-2　@cid 值为“0003”时的执行结果

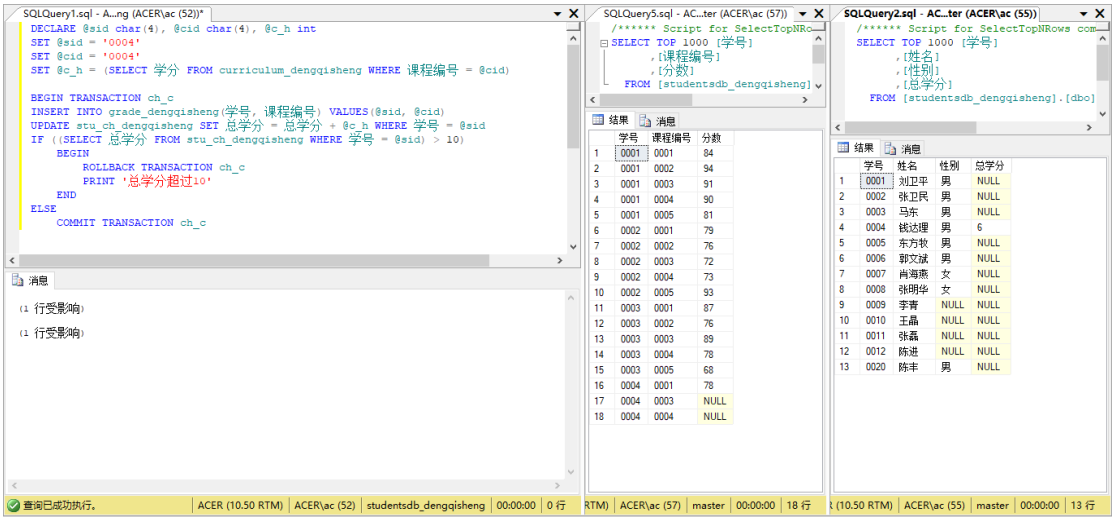

图 2-15-3　@cid 值为“0004”时的执行结果

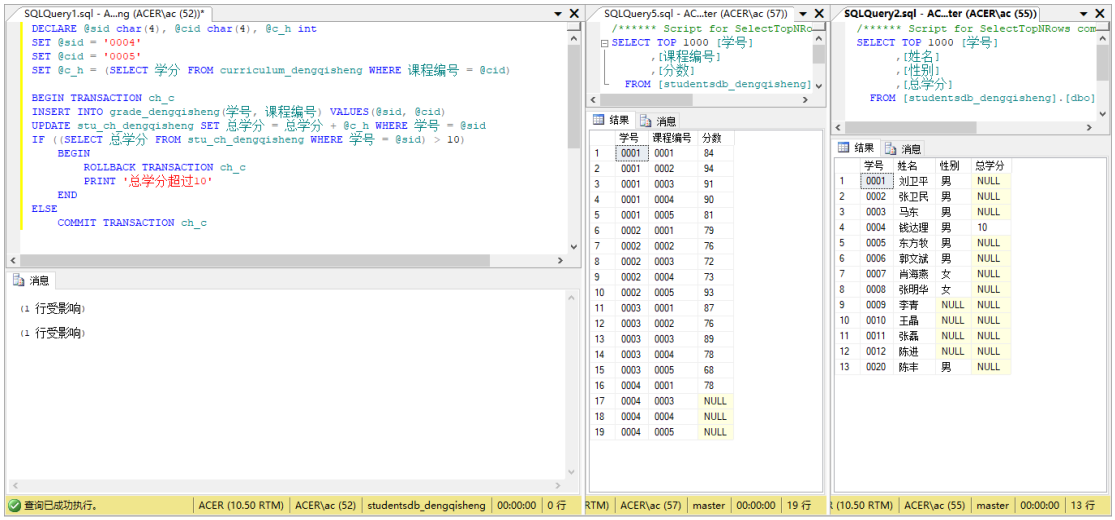

图 2-15-4　@cid 值为“0005”时的执行结果

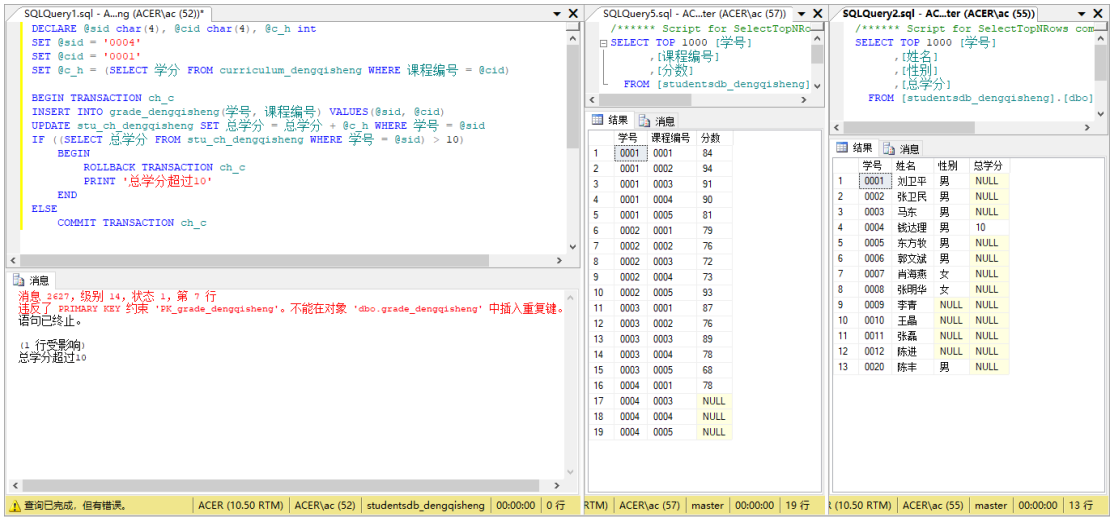

图 2-15-5　@cid 值为“0001”时的执行结果

分析：

观察以上执行结果，当课程编号@cid 赋值为“0003”、“0004”、“0005”时，grade_dengqisheng 表中学号值“0004”处依次对应增加课程编号为“0003”、“0004”、“0005”的记录，stu_ch_dengqisheng 表中总学分先后从 2 变为 6 再变为 10。当@cid 值为“0001”时，由于 grade_dengqisheng 表中已经存在学号为“0004”且课程编号为“0001”的记录，故 grade_dengqisheng 表不再发生变化，而 stu_ch_dengqisheng 表由于总学分超过了 10 导致了事务回滚语句的执行，同样不再发生变化，并且弹出警告“总学分超过 10”。

16. 使用 SELECT 语句为 student_info_dengqisheng 表添加表级锁定(NOLOCK)，通过系统存储过程 sp_lock 查看有关锁的信息，注意锁的信息存储的数据库。在 SQL Server 2008 企业管理器中，观察用户对资源的锁定。

查询结果显示如图 2-16-1 和图 2-16-2。

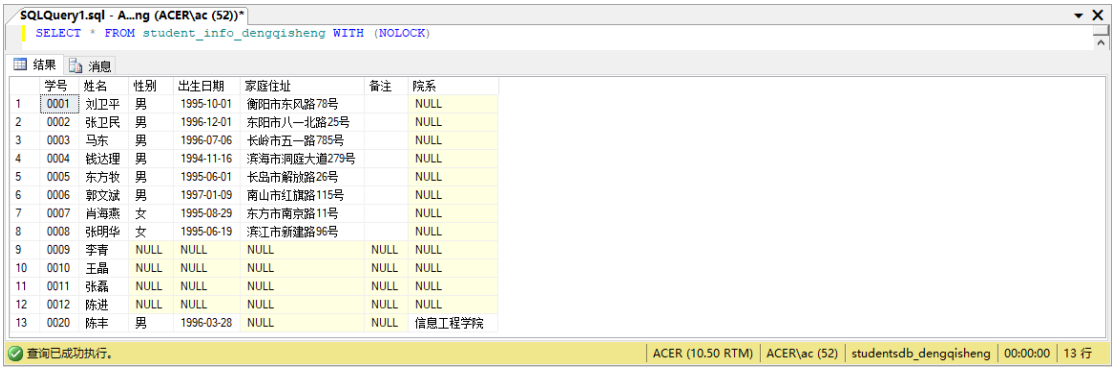

图 2-16-1　添加表级锁定

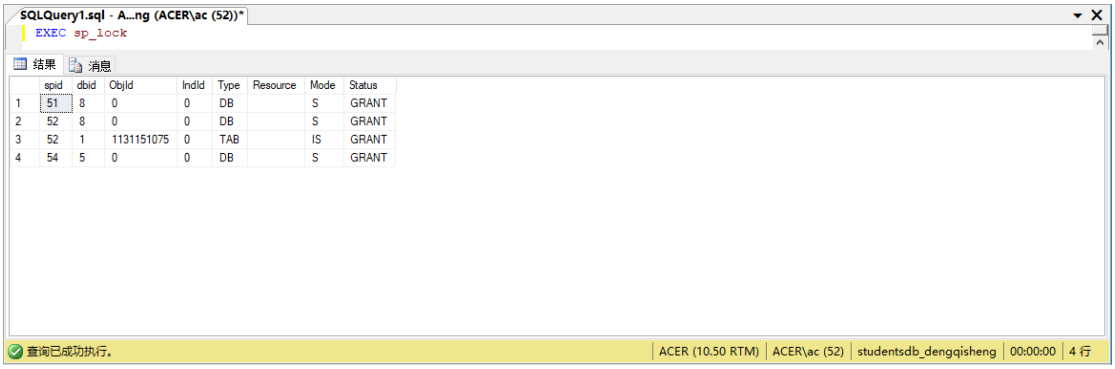
SQLQuery1.sql - A...ng (ACER\ac (52))*
EXEC sp_lock
结果
消息
spid dbid ObjId IndId Type Resource Mode Status
1 51 8 0 0 DB S GRANT
2 52 8 0 0 DB S GRANT
3 52 1 1131151075 0 TAB IS GRANT
4 54 5 0 0 DB S GRANT
查询已成功执行。
ACER (10.50 RTM)
ACER\ac (52)
studentsdb_dengqisheng
00:00:00
4 行

图 2-16-2　查看锁信息