


学号：031602523 姓名：刘宏岩 学院：数计学院 专业：计算机类


## 《数据库应用实践》实验二：数据库管理系统的维护与管理

- 实验目的：

掌握 DBMS 提供的数据库用户和权限管理机制；理解存储过程概念，掌握存储过程与触发器的使用；掌握数据库备份与恢复方法。

- 实验环境：


 操作系统：Windows 10

 数据库管理系统：SQL sever 2017

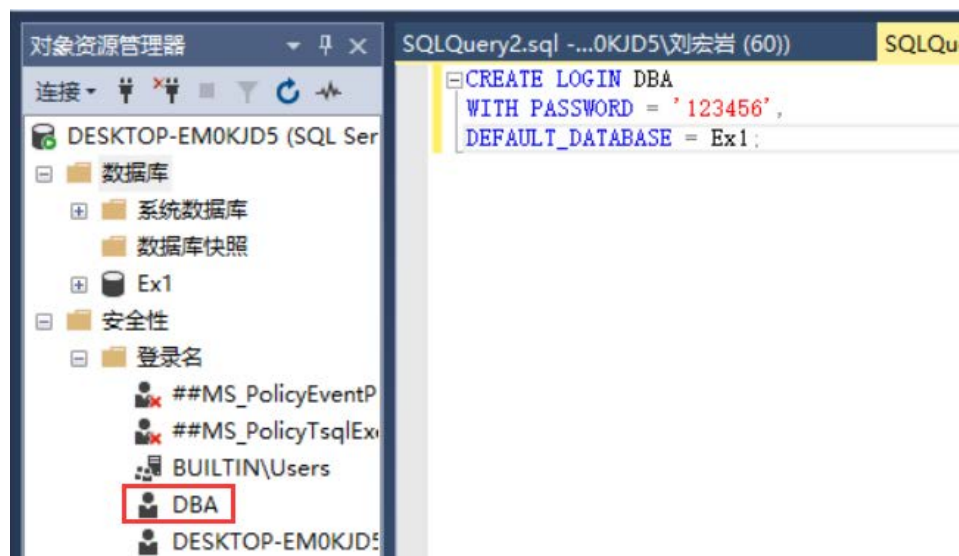
- 实验内容：

- 1) 数据库安全性：

1. DBMS 登录帐号管理：

 创建登陆账户 (create login)

```
CREATE LOGIN DBA  
WITH PASSWORD = '123456',  
DEFAULT_DATABASE = Ex1;
```



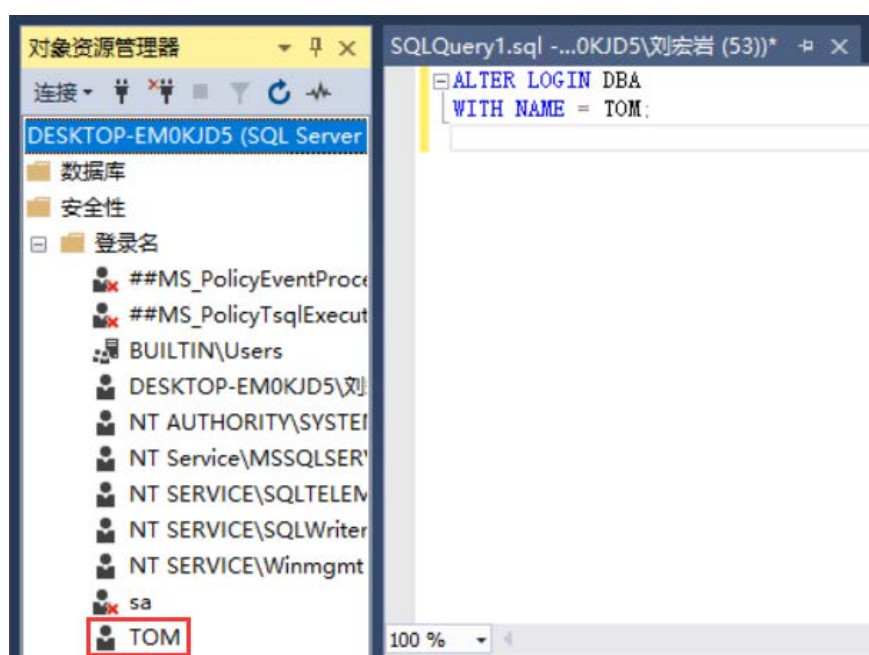
🚦 登陆帐户名为：“DBA”，登陆密码：123456，默认连接到的数据库：

“Ex1”。这时候，DBA 帐户就可以连接到 SQL Server 服务器上了。但是此时还不能访问数据库中的对象。

🚦 要使 DBA 帐户能够在 Ex1 数据库中访问自己需要的对象。需要在数据库 Ex1 中建立一个“数据库用户”，赋予这个“数据库用户”某些访问权限，并且把登陆帐户“DBA”和这个“数据库用户”映射起来。习惯上，“数据库用户”的名字和“登陆帐户”的名字相同，即：“DBA”。

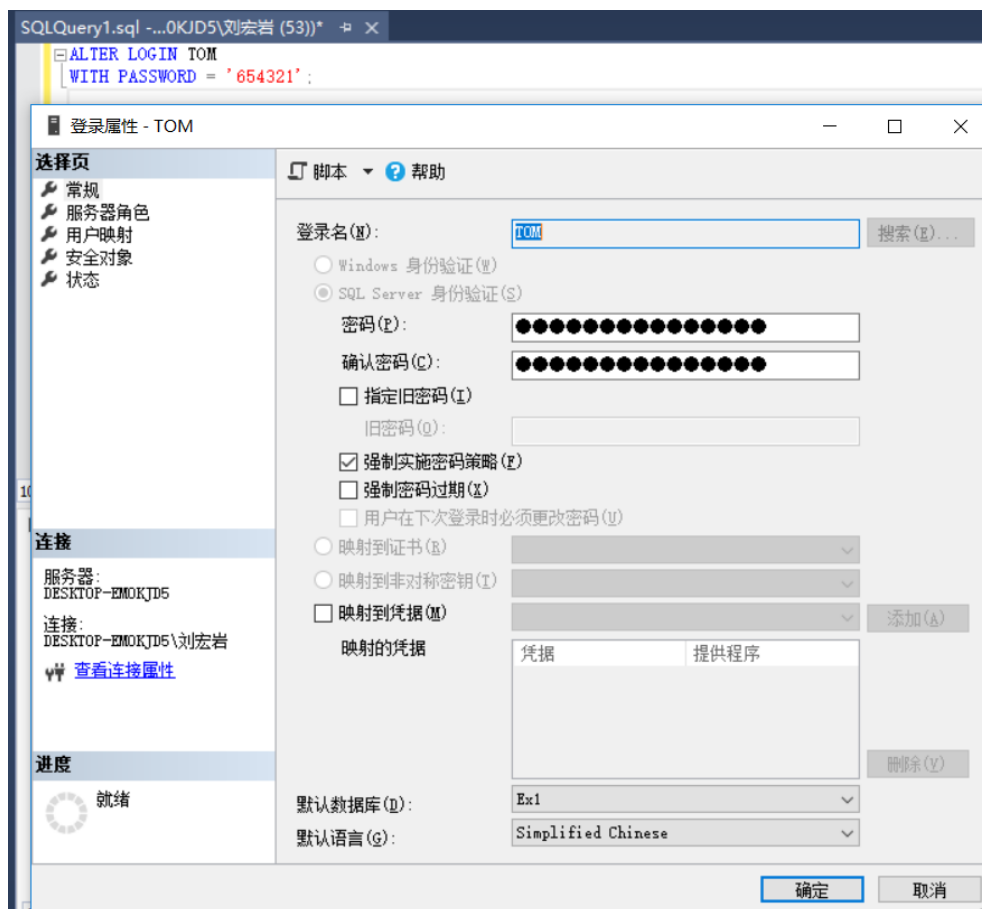
🚦 修改登陆用户名：

```
ALTER LOGIN DBA  
WITH NAME = TOM;
```



🚦 修改用户登陆密码：

```
ALTER LOGIN TOM  
WITH PASSWORD = '654321';
```



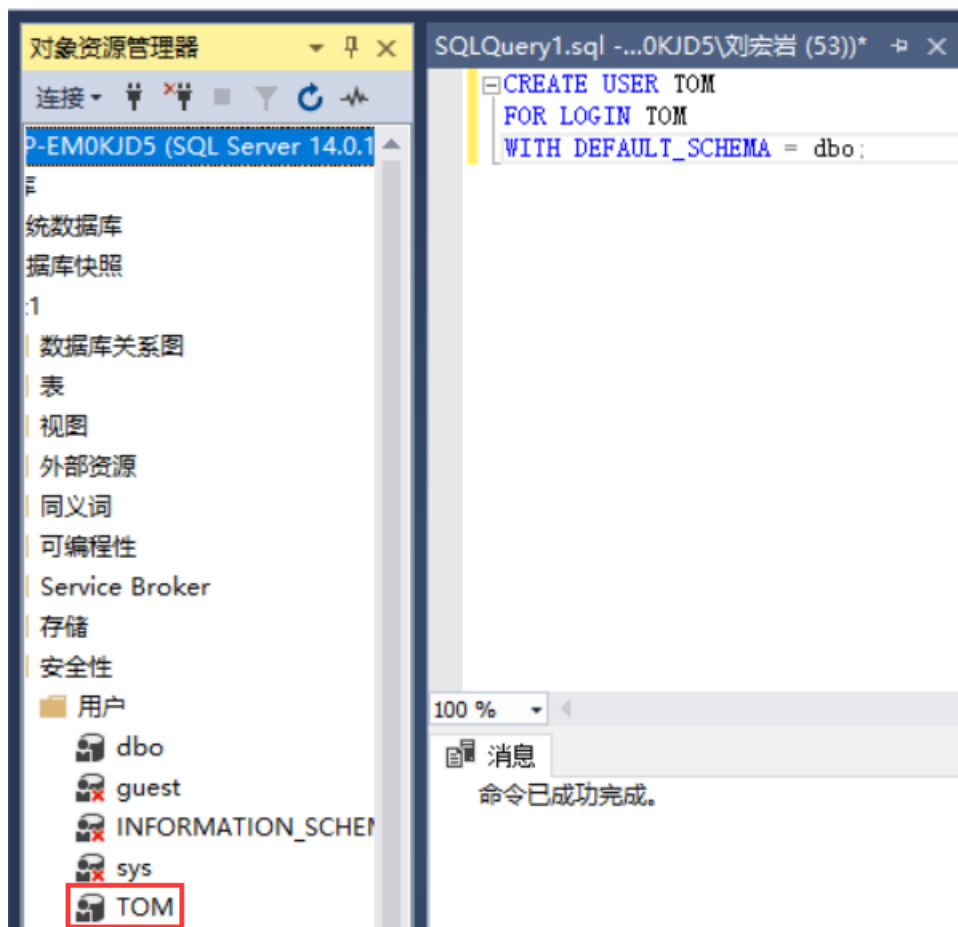
删除登陆用户：(这里就不做演示了)

```
DROP LOGIN TOM;
```

## 2. 数据库用户管理：

创建数据库用户 (create user)：

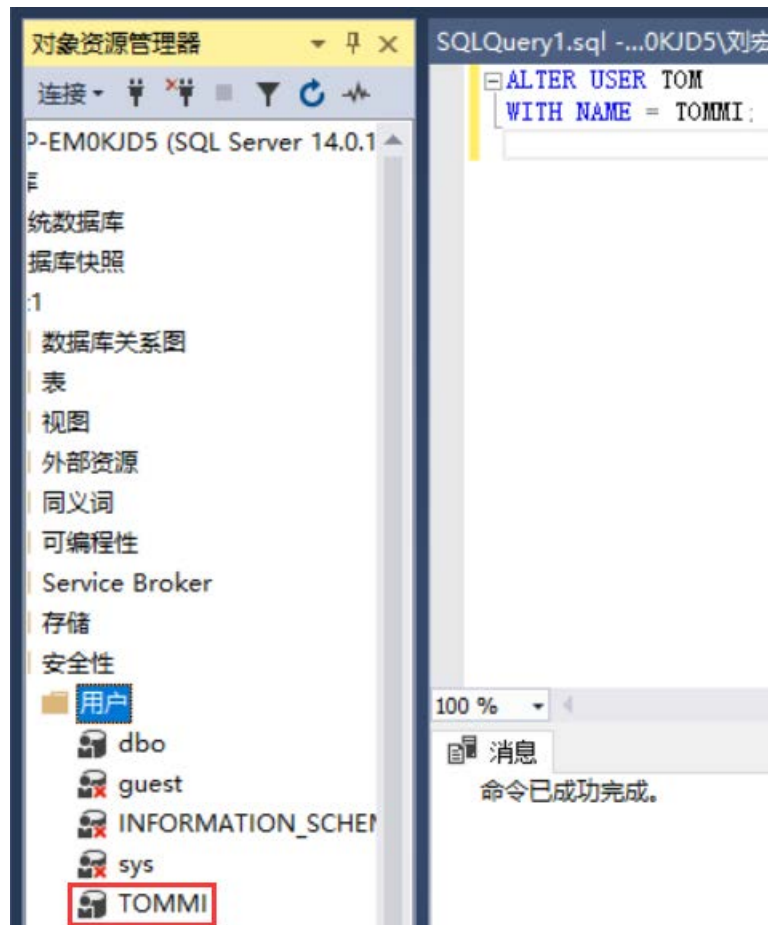
```
CREATE USER TOM  
FOR LOGIN TOM  
WITH DEFAULT_SCHEMA = dbo;
```



为登陆账户创建数据库用户 TOM, 并指定数据库用户 “TOM” 的默认 schema 是 “dbo”。这意味着：用户 “TOM” 在执行 “select \* from t”，实际上执行的是 “select \* from dbo.t”。

用户改名：

```
ALTER USER TOM  
WITH NAME = TOMMI;
```



🚧 用户删除：(这里就不做演示了)

```
DROP USER TOMMI
```

### 3. 对数据库用户进行授予、收回权限：

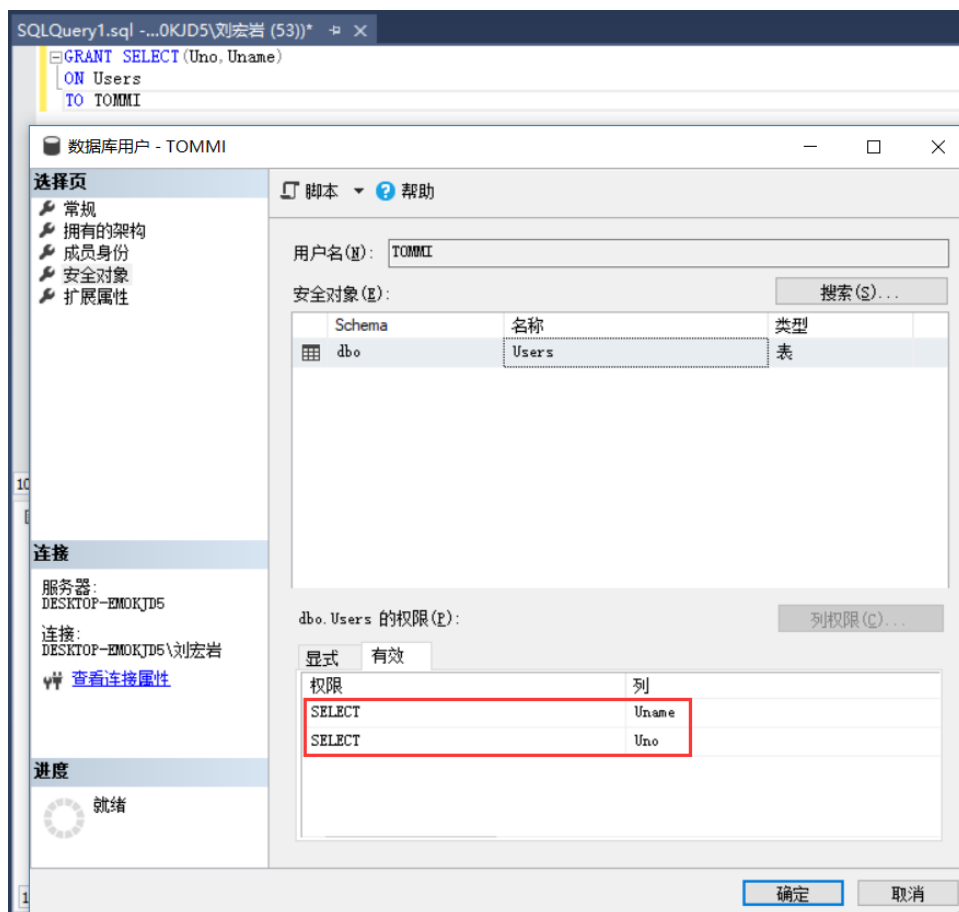
🚧 角色授权，赋予数据库用户“db\_owner”权限。

🚧 由于某些原因 GRANT 语句不能使用，要使用 EXEC 语句：

```
exec sp_addrolemember 'db_owner', 'TOMMI';
```

🚧 授予用户 TOMMI 对 Users 表 Uno 和 Uname 的查询权限：

```
GRANT SELECT(Uno,Uname)
ON Users
TO TOMMI;
```

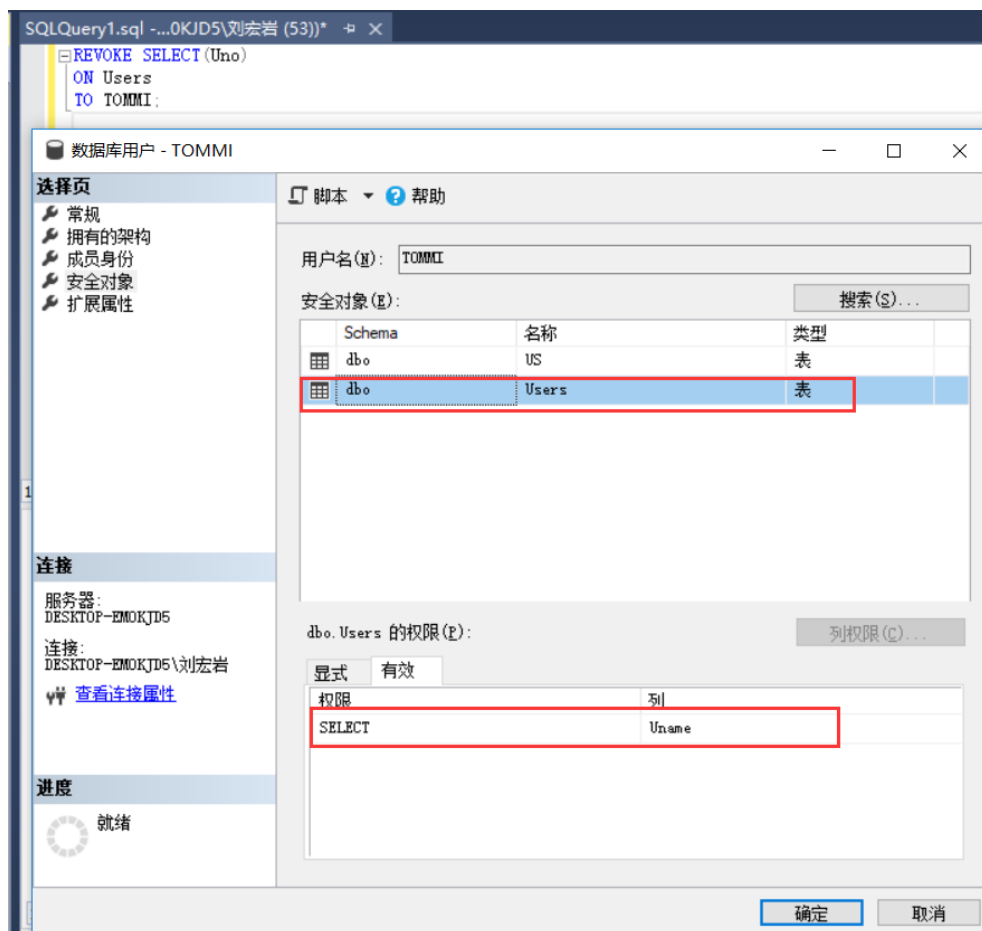


注意：这个 SQL 语句与书上不同。

授予用户 TOMMI 对 US 表的删除权限：


```
GRANT DELETE  
ON US  
TO TOMMI;
```






## 2) 触发器，存储过程的使用：

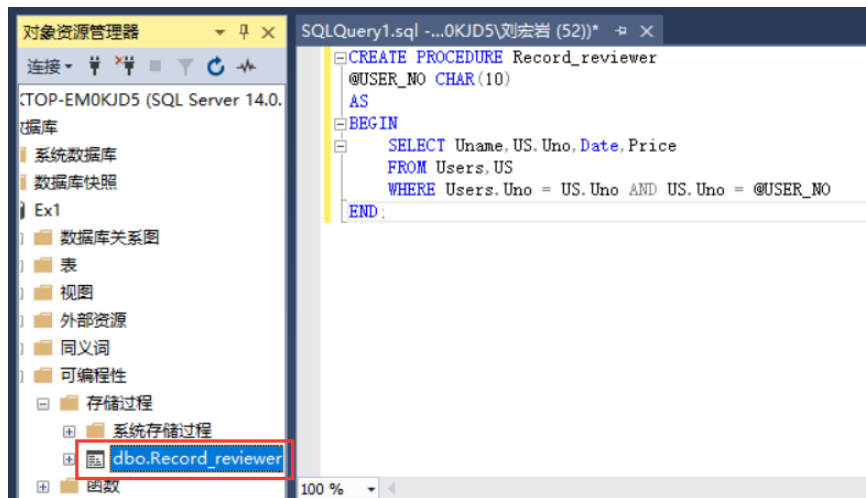
### 1. 存储过程的创建、执行与管理：

 存储过程 (Stored Procedure) 是一组为了完成特定功能的 SQL 语句集，经编译后存储在数据库中。用户通过指定存储过程的名字并给出参数 (如果该存储过程带有参数) 来执行它。

 创建存储过程，查询指定用户的消费记录：

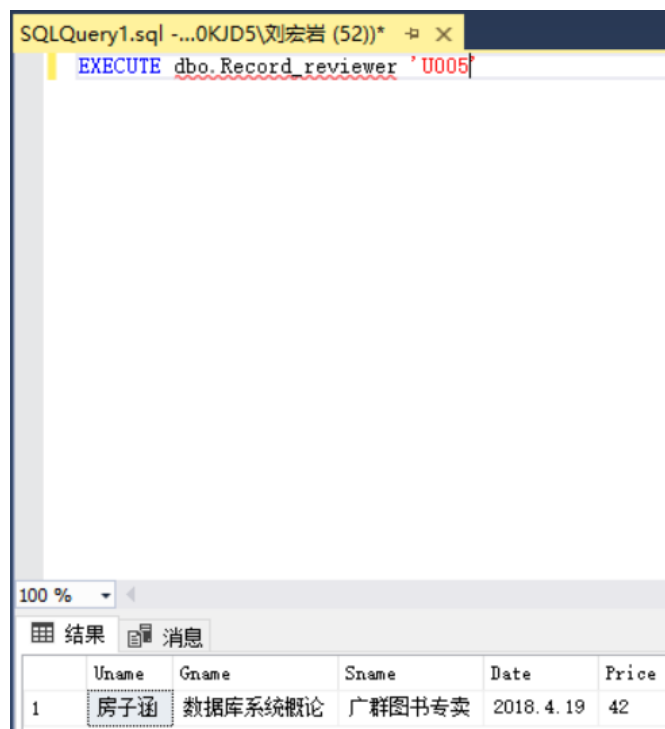
```
CREATE PROCEDURE Record_reviewer
@USER_NO CHAR(10)
AS
BEGIN
    SELECT Username,Gname,Sname,Date,Price
    FROM Users,US,Goods,Shop
    WHERE Users.Uno = US.Uno AND US.Gno = Goods.Gno AND US.Sno
    = Shop.Sno AND US.Uno = @USER_NO
END;
```





输入用户号即可得出购买记录：

EXECUTE dbo.Record\_reviewer 'U005'



存储过剩的修改，增加返回消费总额功能：

```

ALTER PROCEDURE Record_reviewer
@USER_NO CHAR(10)
AS
BEGIN
    SELECT Uname,Gname,Sname,Date,Price
    FROM Users,US,Goods,Shop
    WHERE Users.Uno = US.Uno AND US.Gno = Goods.Gno AND US.Sno
= Shop.Sno AND US.Uno = @USER_NO
END
BEGIN
    SELECT SUM(Price)
    FROM Users,US,Goods,Shop
    WHERE Users.Uno = US.Uno AND US.Gno = Goods.Gno AND US.Sno
= Shop.Sno AND US.Uno = @USER_NO
END;

```

🚦 执行:

```
EXECUTE dbo.Record_reviewer 'U005'
```

SQLQuery1.sql -...0KJD5\刘宏岩 (52))\* ✕

EXECUTE dbo.Record\_reviewer 'U001'

100 %

结果 消息

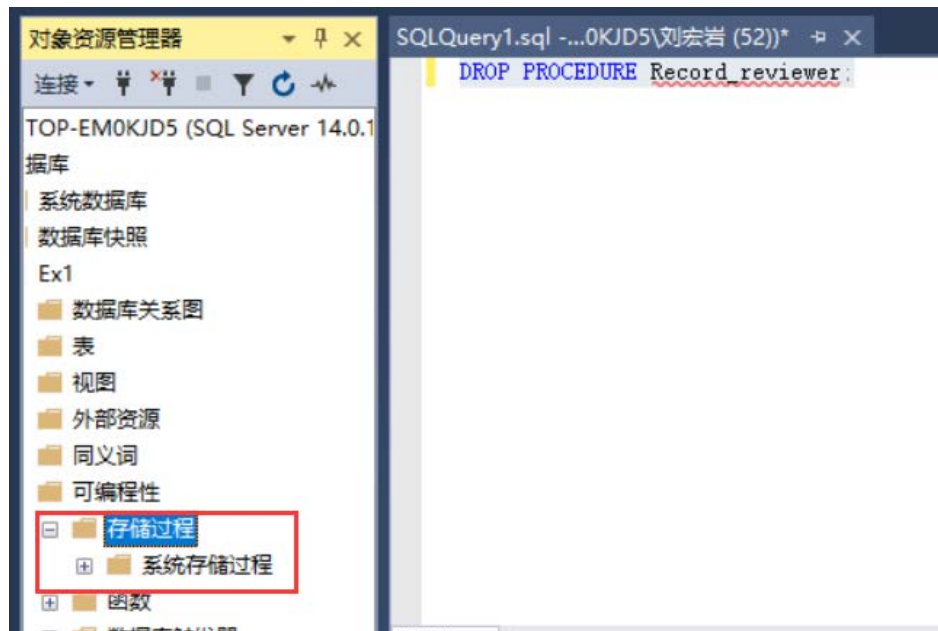
	Uname	Gname	Sname	Date	Price
1	余锦昌	数据库系统概论	广群图书专卖	2018.5.1	42
2	余锦昌	华为平板电脑	永安数码	2018.5.2	2000

	(无列名)
1	2042

🚦 存储过程删除：

```
DROP PROCEDURE Record_reviewer;
```

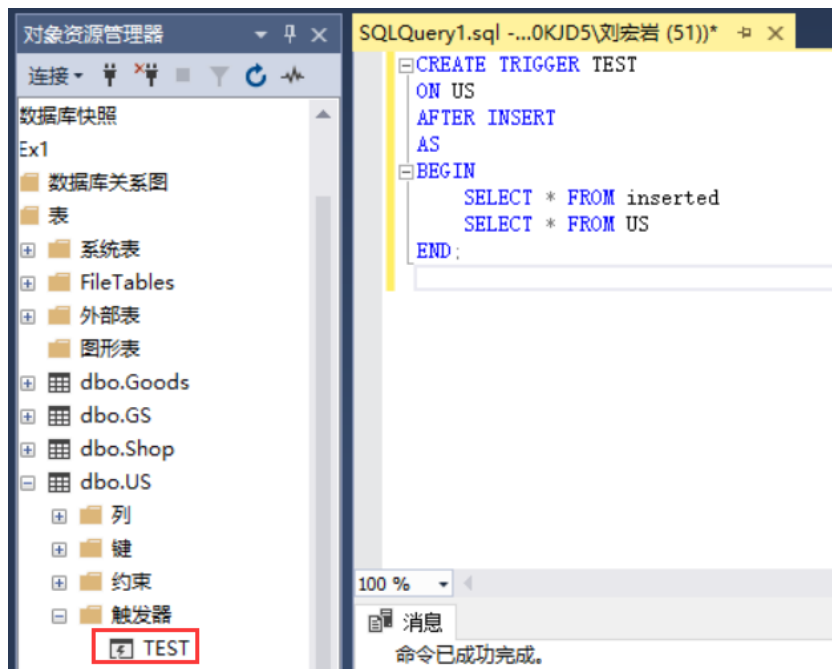


## 2. 触发器的创建与管理：

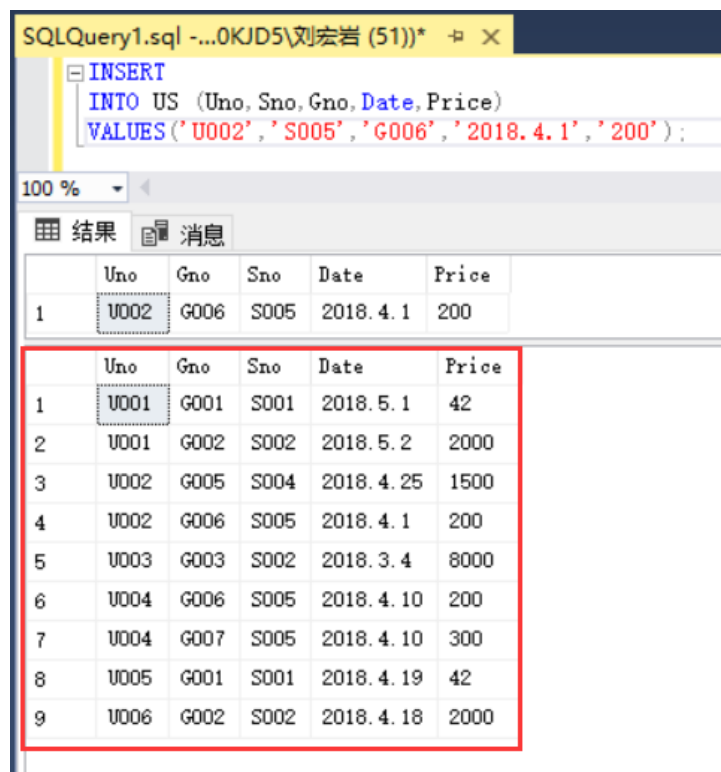
🚦 当我们向购买记录插入数据时，插入后打印出整表信息：

```
CREATE TRIGGER TEST  
ON US  
AFTER INSERT  
AS  
BEGIN  
    SELECT * FROM inserted  
    SELECT * FROM US  
END;
```

🚦 注意：SQL sever 的触发器语法和书上不同。



✚ 插入数据测试：

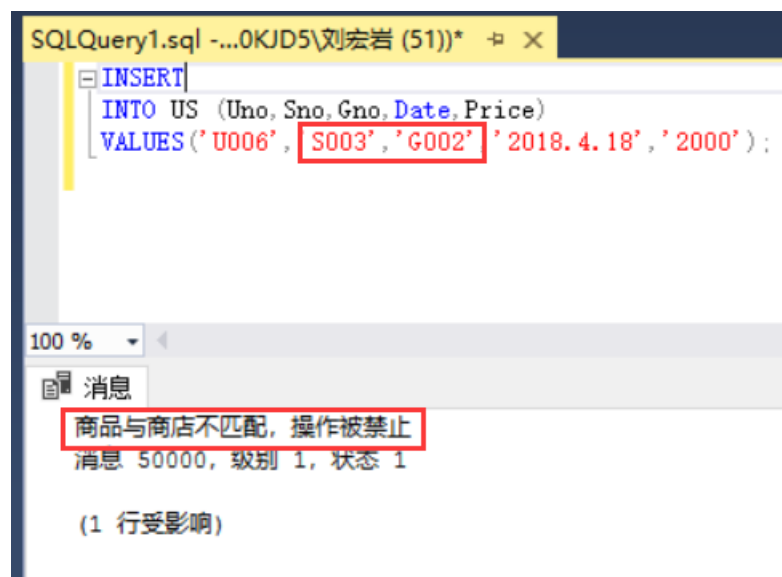


✚ 我们知道，当我们插入数据时，如果货物与商店不匹配是不行的，我们要拒绝这种非法插入，于是修改触发器为：

```
ALTER TRIGGER TEST
ON US
AFTER INSERT
AS
/*定义变量，用于临时存储插入的会员号、电脑编号和卡的编号*/
DECLARE @SNO CHAR(10)
DECLARE @GNO CHAR(4)
SELECT @SNO = Sno,@GNO = Gno FROM inserted
BEGIN
    IF(@SNO NOT IN (SELECT Sno FROM GS WHERE Gno = @GNO))
        RAISERROR('商品与商店不匹配，操作被禁止',1,1)
        DELETE FROM US WHERE Sno = @SNO AND Gno = @GNO
END;
```

✚ SQL sever 中插入的临时变量存储在表 inserted 中，我们通过要定义变量来使用它们。

✚ 我们尝试插入一个违法数据：



✚ 由于 3 号商店不出售 2 号商品，所以被拒绝插入。

✚ 删除触发器，（这里不做演示，因为这是一个有意义的触发器）：

```
DROP TRIGGER TEST
```

### 3) 数据库的备份与恢复:

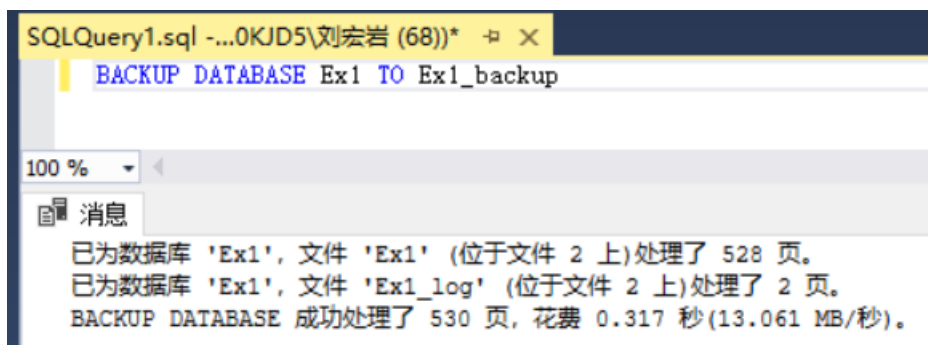
#### 1. 对所创建的数据库进行完整、差异、日志备份:

对所创建的数据库进行完整、差异、日志备份在备份之前，先需要建立一个磁盘备份设备。

```
EXEC sp_addumpdevice  
'disk','Ex1_backup','D:\SQL_backup\Ex1_backup.bak';
```

把数据库完整备份到这个备份设备中:

```
BACKUP DATABASE Ex1 TO Ex1_backup
```



查看备份信息:



数据库差异备份到这个备份设备中:

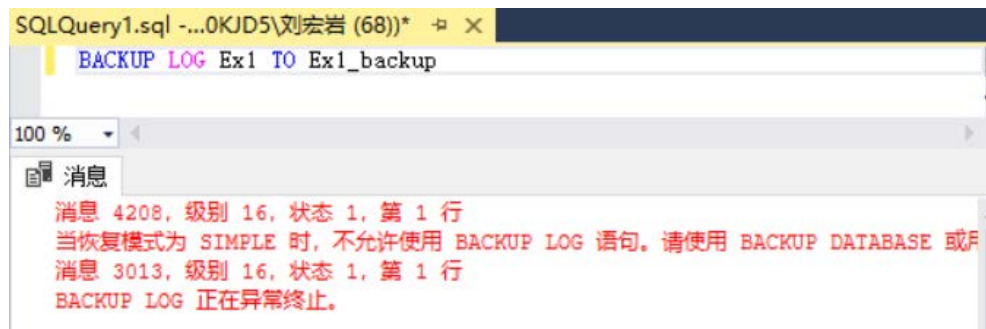
```
BACKUP DATABASE Ex1 TO Ex1_backup  
WITH DIFFERENTIAL
```

查看备份信息：

数据库	完整	DESK...	Ex1	3	2018...	2018...	3700...	3700...	3700...	4407296	DESK...
数据库	差异	DESK...	Ex1	4	2018...	2018...	3700...	3700...	3700...	409600	DESK...

数据库日志备份到这个备份设备中：

```
BACKUP LOG Ex1 TO Ex1_backup
```



这时会报错，由于数据库为简单恢复模式无法进行，需要先修改数据库为完整恢复模式。

```
ALTER DATABASE Ex1 SET RECOVERY FULL
```

然后再执行，这时还会报错：

无法执行 BACKUP LOG，

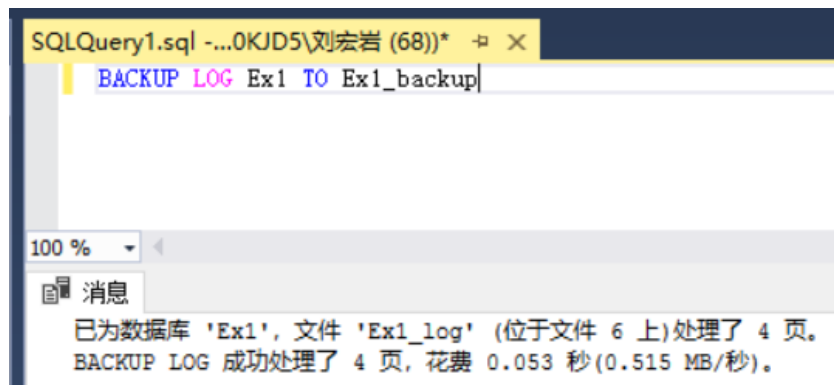
因为当前没有数据库备份。

BACKUP LOG 正在异常终止。

这是因为：

使用简单恢复模式，系统发生检查点之后，数据库中的事务历史记录会被清除，也就是备份记录链会被切断，因此没办法支持任何事务历史记录档备份，只能使用完整或差异备份来备份数据库。要解决这个问题，只要在从简易恢复模式切换至完整或大容量日志恢复模式之后，立刻进行完整或差异备份，记录链结就会再次被启动，您就可以继续进行事务历史记录备份的作业。

执行一次完整备份后再执行日志备份：



查看备份信息：

数据库	差异	DESK...	Ex1	4	2018/5/19 2:04:45	2018/5/19 2:04:45	37000000140000034	37000000143...	3700...	409600	DESK...
数据库	完整	DESK...	Ex1	5	2018/5/19 2:20:40	2018/5/19 2:20:40	37000000147200037	37000000150...	3700...	4407296	DESK...
	事务日志	DESK...	Ex1	6	2018/5/19 2:20:49	2018/5/19 2:20:49	37000000147200037	37000000152...	3700...	77824	DESK...

## 2. 数据库恢复：

我们可以在进行完整恢复和日志恢复之前，可以先插入故障数据：

```
INSERT  
INTO Users(Uno, Uname, Ukey, Usex, Ustar)  
VALUES('U007', '林银彬', 'lyb1', '男', 5);
```

进行完整恢复和日志恢复处理。

在恢复其他数据库下因为不能还原当前正在使用的数据库，我们使用

master：

```
USE master  
RESTORE DATABASE Ex1  
FROM Ex1_backup  
WITH REPLACE, NORECOVERY;  
  
RESTORE LOG Ex1  
FROM Ex1_backup;
```



```
SQLQuery1.sql -...0KJD5\刘宏岩 (68))*
USE master
RESTORE DATABASE Ex1
FROM Ex1_backup
WITH REPLACE, NORECOVERY;

RESTORE LOG Ex1
FROM Ex1_backup;
```

100 %

消息

已为数据库 'Ex1', 文件 'Ex1' (位于文件 1 上)处理了 528 页。  
 已为数据库 'Ex1', 文件 'Ex1\_log' (位于文件 1 上)处理了 3 页。  
 RESTORE DATABASE 成功处理了 531 页, 花费 0.217 秒(19.099 MB/秒)。  
 已为数据库 'Ex1', 文件 'Ex1' (位于文件 1 上)处理了 0 页。  
 已为数据库 'Ex1', 文件 'Ex1\_log' (位于文件 1 上)处理了 3 页。  
 RESTORE LOG 成功处理了 3 页, 花费 0.089 秒(0.219 MB/秒)。

这时我们查看 Users 表, 发现刚才插入的异常数据已经不见了:

	Uno	Uname	Usex	Ukey	Ustar
1	U001	余锦昌	男	yjc1	1
2	U002	郑云涛	男	dccl	2
3	U003	郑得华	男	zdh1	3
4	U004	许云涛	女	xyt1	4
5	U005	房子涵	男	fzh1	5
6	U006	陈星屹	男	cxy1	5

差异恢复。

使用差异恢复之前, 我们可以先删除 GS 表的全部数据。

```
DELETE
FROM GS
```

```
SQLQuery1.sql -...0KJD5\刘宏岩 (68))*
DELETE
FROM GS
```

100 %

消息

(7 行受影响)

结果	消息
Gno	Sno

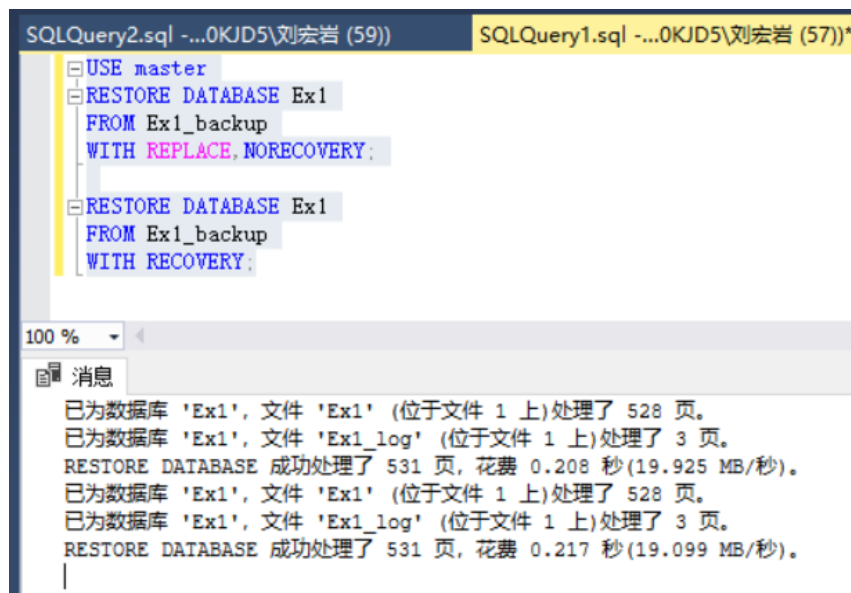
进行差异备份回复：

在恢复其他数据库下因为不能还原当前正在使用的数据库，需使用

master:

```
USE master
RESTORE DATABASE Ex1
FROM Ex1_backup
WITH REPLACE,NORECOVERY;
```

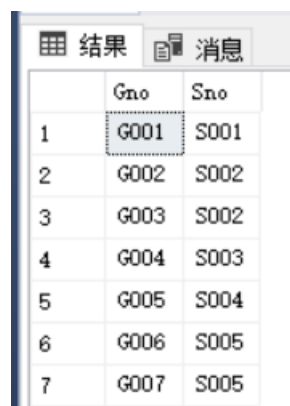
```
RESTORE DATABASE Ex1
FROM Ex1_backup
WITH RECOVERY;
```



The screenshot shows the SQL Server Enterprise Manager interface. The top pane displays two SQL queries: 'SQLQuery2.sql' and 'SQLQuery1.sql'. The bottom pane shows the execution results in the 'Messages' tab. The messages indicate that the database 'Ex1' was successfully restored with the 'REPLACE' option, and the log files were also restored. The messages are as follows:

```
已为数据库 'Ex1', 文件 'Ex1' (位于文件 1 上)处理了 528 页。
已为数据库 'Ex1', 文件 'Ex1_log' (位于文件 1 上)处理了 3 页。
RESTORE DATABASE 成功处理了 531 页, 花费 0.208 秒(19.925 MB/秒)。
已为数据库 'Ex1', 文件 'Ex1' (位于文件 1 上)处理了 528 页。
已为数据库 'Ex1', 文件 'Ex1_log' (位于文件 1 上)处理了 3 页。
RESTORE DATABASE 成功处理了 531 页, 花费 0.217 秒(19.099 MB/秒)。
```

查看被恢复的 GS 表：



	Gno	Sno
1	G001	S001
2	G002	S002
3	G003	S002
4	G004	S003
5	G005	S004
6	G006	S005
7	G007	S005

删除元组恢复完毕。

- 实验心得体会：

通过这次实验，我学会了数据库用户和权限管理机制；理解存储过程概念，掌握存储过程与触发器的使用；掌握数据库备份与恢复方法。

由于书上使用的 DBMS 和 SQL sever 有些许的不同，导致很多 SQL 语句在 SQL sever 上无法使用，通过查阅博客，我解决了这些问题。问题出现时的苦恼，等价于解决问题时的喜悦。在这个过程中，自己学到了很多课外知识，自己的能力也在不断提高。

这次实验提高了我解决问题的能力，也让我知道：一个合格的数据库工作者不仅需要扎实的理论基础，更需要熟悉适应各种 DBMS 和它们的不同版本。