第9章　存储过程和触发器

9.1存储过程

9.1.1　存储过程的基本知识

1. 概念

存储过程（Stored Procedure）是一组**编译好**存储在**服务器**上的完成特定功能T-SQL代码，是某数据库的对象。**客户端**应用程序可以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）来执行存储过程。

2. 优点

使用存储过程而不使用存储在客户端计算机本地的 T-SQL 程序的优点包括：

(1) 允许标准组件式编程，增强重用性和共享性

(2) 能够实现较快的执行速度

(3) 能够减少网络流量

(4) 可被作为一种安全机制来充分利用

3. 分类

在SQL Server 2005中存储过程分为三类：系统提供的存储过程、用户自定义存储过程和扩展存储过程。

系统：系统提供的存储过程，**sp_*，例如：sp_rename**

扩展：SQL Server环境之外的动态链接库DLL，xp_

远程：远程服务器上的存储过程

用户：创建在用户数据库中的存储过程

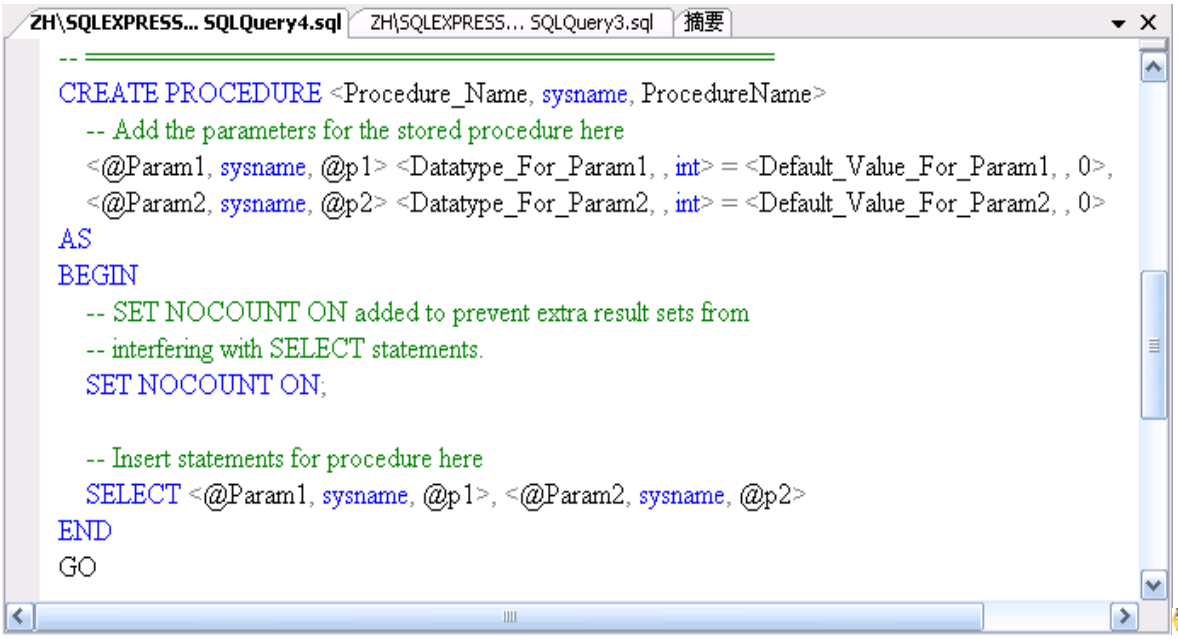
临时：属于用户存储过程，#开头（局部：一个用户会话）,##（全局：所有用户会话）

9.1.2　创建用户存储过程

1. 使用存储过程模板创建存储过程

在【对象资源管理器】窗口中，展开“数据库”节点，再展开所选择的具体数据库节点，再展开选择“可编程性”节点，右击“存储过程”，选择“新建存储过程”命令，如图所示：

在右侧查询编辑器中出现存储过程的模板，用户可以在此基础上编辑存储过程，单击“执行”按钮，即可创建该存储过程。



例9-1：创建一个简单的存储过程。

```
USE Library
GO
CREATE PROCEDURE borrowed_num
AS
SELECT Rname,Lendnum
FROM Reader
WHERE Rname='赵良宇'
```

存储过程建好了，什么时候，怎么用呢？

执行存储过程：borrowed_num 或EXEC borrowed_num

执行结果：

结果 消息

	Rname	Lendnum
1	赵良宇	1

2. 使用T-SQL语句创建表

格式：

CREATE PROC 过程名

@形参名 类型

@变参名 类型 OUTPUT

AS SQL语句

例9-2：创建一个多表查询的存储过程。

```
USE Library
GO
CREATE PROCEDURE borrowed_book1
AS
SELECT r.RID,r.Rname,b.BID,k.Bname,b.LendDate
FROM reader r INNER JOIN borrow b
ON r.RID=b.RID INNER JOIN book k
ON b.BID=k. BID
WHERE Rname='程鹏'
```

执行存储过程：borrowed_book1 或EXEC borrowed_book1

执行结果：

结果 消息

	RID	Rname	BID	Bname	LendDate
1	2005216119	程鹏	F1193-04	ERP系统的集成应用	2007-03-18 .
2	2005216119	程鹏	TP8283-02	SQL Server 2005基础教程	2007-03-30 .

9.1.3 存储过程的参数

1. 输入参数（值参）

例9-3：输入参数为某人的名字。

```
USE Library
GO
CREATE PROCEDURE borrowed_book2
@name varchar(10)    --形式参数
As
SELECT r.RID,r.Rname,b.BID,k.Bname,b.LendDate
FROM reader r INNER JOIN borrow b
ON r.RID=b.RID INNER JOIN book k
ON b.BID=k. BID
WHERE Rname=@name
GO
```

执行存储过程：

直接传值：`EXEC borrowed_book2 '程鹏'  --实参表`

变量传值：

```
DECLARE @temp1 char(20)
SET @temp1='杨树华'
EXEC borrowed_book2 @temp1  --实参表
```

执行结果：

结果 消息

	RID	Rname	BID	Bname	LendDate
1	2005216118	杨树华	TP97-05	单片计算机原理与应用	2007-03-15 .

例9-4：使用默认参数

```
USE Library
GO
CREATE PROCEDURE borrowed_book3
@name varchar(10)=NULL  --默认参数
AS
IF @name IS NULL
   SELECT r.RID,r.Rname,b.BID,k.Bname,b.LendDate
   FROM reader r INNER JOIN borrow b
   ON r.RID=b.RID INNER JOIN book k
   ON b.BID=k.BID
ELSE
   SELECT r.RID,r.Rname,b.BID,k.Bname,b.LendDate
   FROM reader r INNER JOIN borrow b
   ON r.RID=b.RID INNER JOIN book k
   ON b.BID=k.BID
   WHERE Rname=@name
GO
```

执行存储过程：`EXEC borrowed_book3`

结果 消息

	RID	Rna...	BID	Bname	LendDate
1	2000186010	张子建	TP5790-03	SQL Server 2005基础教程	2006-08-30.
2	2000186011	赵良宇	TP311-06	数据库系统概论	2006-12-06.
3	2004216010	任灿灿	F270-11	ERP从内部集成开始	2006-03-15.
4	2005216117	孟霞	TP118-12	算法与数据结构	2006-10-08.
5	2005216118	杨树华	TP97-05	单片计算机原理与应用	2007-03-15.
6	2005216119	程鹏	F1193-04	ERP系统的集成应用	2007-03-18.
7	2005216119	程鹏	TP8283-02	SQL Server 2005基础教程	2007-03-30.

2. 输出参数（变参）

例9-5：利用输出参数计算阶乘。

```
USE Library
IF EXISTS(SELECT name FROM sysobjects
          WHERE  name='factorial' AND type='P')
    DROP PROCEDURE factorial
```

```
GO
CREATE  PROCEDURE factorial
  @in float,            --输入形式参数
  @out float OUTPUT  --输出形式参数
AS
DECLARE @i int
DECLARE @s float
SET @i=1
SET @s=1
WHILE @i<=@in
      BEGIN
           SET @s=@s*@i
           SET @i=@i+1
      END
SET @out=@s             --给输出参数赋值
```

调用存储过程：

```
DECLARE @ou float
EXEC factorial 5,@ou OUT  --实参表
PRINT @ou
```

执行结果： 

9.2触发器

9.2.1　触发器的基本知识

1.基本概念

触发器是特殊的存储过程，基于一个表创建，主要作用就是实现由主键和外键所不能保证的复杂的参照完整性和数据一致性。

当触发器所保护的数据发生变化（update,insert,delete）后，自动运行以保证数据的完整性和正确性。通俗的说：通过一个动作（update,insert,delete）调用一个存储过程（触发器）。

2.类型

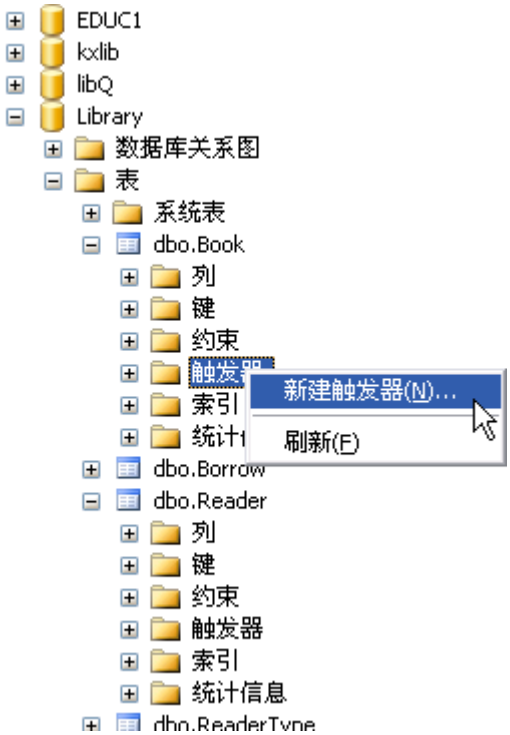
(1) DML触发器

在数据库中发生数据操作语言（DML）事件时将启用。DML 事件包括在指定表或视图中修改数据的 INSERT 语句、UPDATE 语句或 DELETE 语句。DML 触发器可以查询其他表，还可以包含复杂的 T-SQL 语句。系统将触发器和触发它的语句作为可在触发器内回滚的单个事务对待，如果检测到错误（例如，磁盘空间不足），则整个事务即自动回滚。

(2) DDL 触发器

SQL Server 2005 的新增功能。当服务器或数据库中发生数据定义语言（DDL）事件时将调用这些触发器。但与DML触发器不同的是，它们不会为响应针对表或视图的UPDATE、INSERT或DELETE语句而激发，相反，它们会为响应多种数据定义语言（DDL）语句而激发。这些语句主要是以CREATE、ALTER和DROP开头的语句。DDL触发器可用于管理任务，例如审核和控制数据库操作。

9.2.2

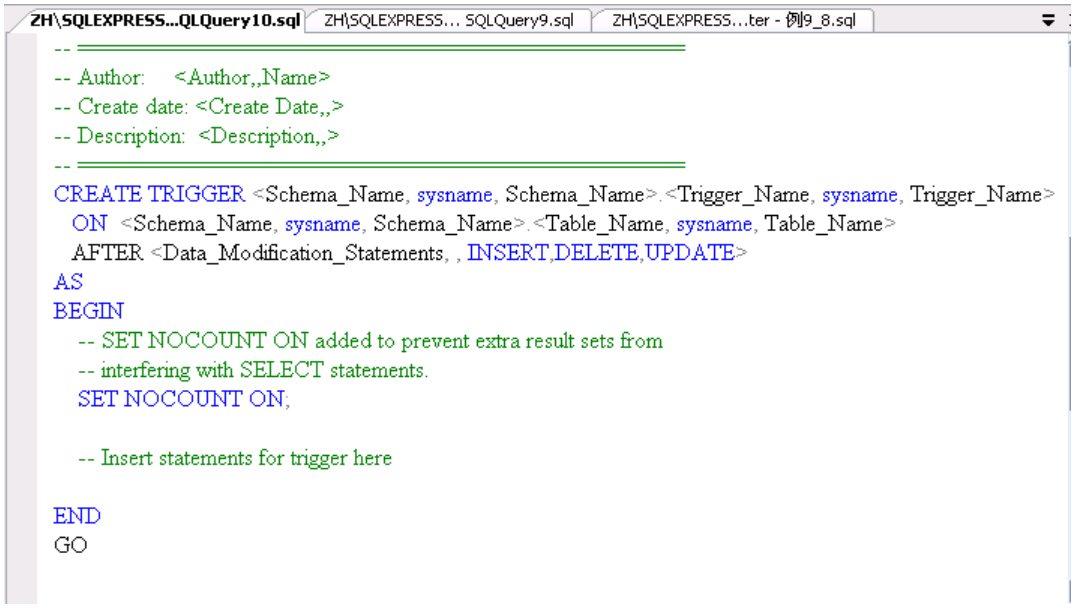


创建DML触发器

1. 使用存储过程模板创建存储过程

在【对象资源管理器】窗口中，展开“数据库”节点，再展开所选择的具体数据库节点，再展开“表”节点，右击要创建触发器的“表”，选择“新建触发器”命令，如图所示：

在右侧查询编辑器中出现触发器设计模板，用户可以在此基础上编辑触发器，单击“执行”按钮，即可创建该触发器。



2. 使用T-SQL语句创建表

```
CREATE TRIGGER 触发器
ON 表名
FOR[update,insert,delete ]
AS SQL语句
```

例9-6：创建基于表reader ，DELETE操作的触发器。

```
USE Library
GO
IF EXISTS(SELECT name FROM sysobjects
          WHERE name='reader_d' AND type='TR')
DROP TRIGGER reader_d      --如果已经存在触发器reader_d则删除
GO
```

```
CREATE TRIGGER reader_d  --创建触发器
ON reader                  --基于表
FOR DELETE               --删除事件
AS
PRINT '数据被删除!'         --执行显示输出
GO
```

应用：

```
USE Library
GO
DELETE reader
where Rname='aaa'
```

执行结果：数据被删除!（所影响的行数为 1 行）

例9-7：在表borrow中添加借阅信息记录时，得到该书的应还日期。

说明：在表borrow中增加一个应还日期SReturnDate。

```
USE Library
IF EXISTS (SELECT name FROM sysobjects
WHERE name ='T_return_date' AND type='TR')
DROP TRIGGER T_return_date
GO
CREATE TRIGGER T_return_date  --创建触发器
ON Borrow                     --基于表borrow
After INSERT                   --插入操作
AS
--查询插入记录INSERTED中读者的类型
DECLARE @type int,@dzbh char(10),@tsbh char(15)
SET @dzbh=(SELECT RID FROM inserted)
SET @tsbh=(SELECT BID FROM inserted)
SELECT @type= TypeID
FROM reader
WHERE RID=(SELECT RID FROM inserted)--副本
/*把Borrow表中的应还日期改为
当前日期加上各类读者的借阅期限*/
UPDATE Borrow SET SReturnDate=getdate()+
    CASE
      WHEN @type=1 THEN 90
      WHEN @type=2 THEN 60
      WHEN @type=3 THEN 30
    END
WHERE RID=@dzbh and BID=@tsbh
```

应用：

```
USE Library
INSERT INTO borrow(RID,BID) values('2000186010','TP85-08')
```

查看记录：

RID	BID	LendDate	ReturnDate	SReturnD
2000186010	TP5790-03	2006-8-30 13:...	2006-10-9 0:0...	*NULL*
2000186010	TP85-08	2007-4-9 21:5...	*NULL*	2007-7-8

例9-8：在数据库Library中，当读者还书时，实际上要修改表brorrowinf中相应记录还期列的值，请计算出是否过期。

```
USE Library
IF EXISTS(SELECT name FROM sysobjects
WHERE name='T_fine_js' AND type='TR')
DROP TRIGGER T_fine_js
GO
CREATE TRIGGER T_fine_js
ON borrow
After UPDATE
AS
```

```
DECLARE @days int,@dzbh char(10),@tsbh char(15)
SET @dzbh=(select RID from inserted)
SET @tsbh=(select BID from inserted)
SELECT @days=DATEDIFF(day, ReturnDate, SReturnDate)
--DATEDIFF函数返回两个日期之差，单位为DAY
FROM borrow
WHERE RID=@dzbh and BID=@tsbh
IF @days>0
   PRINT '没有过期！'
ELSE
   PRINT '过期'+convert(char(6),@days)+'天'
GO
```

应用：

```
USE Library
UPDATE borrow SET ReturnDate='2007-12-12'
WHERE RID='2000186010' and BID='TP85-08'
GO
```

执行结果：

过期-157 天

(1 行受影响)

例9-9：对Library库中Reader表的 DELETE操作定义触发器。

```
USE Library
GO
IF EXISTS(SELECT name FROM sysobjects
   WHERE name='reader_d' AND type='TR')
DROP TRIGGER reader_d
GO
CREATE TRIGGER reader_d
ON Reader
FOR DELETE
AS
DECLARE @data_yj int
SELECT  @data_yj=Lendnum
FROM deleted
IF @data_yj>0
   BEGIN
     PRINT '该读者不能删除！还有'+convert(char(2),@data_yj)+'本书没还。
     ROLLBACK
   END
ELSE
   PRINT '该读者已被删除！'
GO
```

应用：

```
USE Library
GO
DELETE Reader WHERE RID='2005216119'
```

执行结果：

该读者不能删除！还有4 本书没还。

9.2.3　创建DDL触发器

DDL 触发器会为响应多种数据定义语言 (DDL) 语句而激发。这些语句主要是以 CREATE、ALTER 和 DROP 开头的语句。DDL 触发器可用于管理任务，例如审核和控制数据库操作。

语法形式：

```
CREATE TRIGGER trigger_name
ON {ALL SERVER|DATABASE}[WITH <ddl_trigger_option> [ ,...n ]]
  {FOR|AFTER} {event_type|event_group}[,...n]
AS {sql_statement[;] [...n]|EXTERNAL NAME <method specifier>[;]}
```

其中：

```
<ddl_trigger_option>::=[ENCRYPTION] EXECUTE AS Clause]
<method_specifier> ::= assembly_name.class_name.method_name
```

例9-10：使用DDL触发器来防止数据库中的任一表被修改或删除。

```
CREATE TRIGGER safety
ON DATABASE
FOR DROP_TABLE, ALTER_TABLE
AS
PRINT 'You must disable Trigger "safety" to drop or alter tables!'
ROLLBACK
```

例9-11：使用 DDL 触发器来防止在数据库中创建表。

```
CREATE TRIGGER safety
ON DATABASE
FOR CREATE_TABLE
AS
PRINT 'CREATE TABLE Issued.'
SELECT
EVENTDATA().value('(/EVENT_INSTANCE/TSQLCommand/CommandText)[1]','nvarchar(max)')
 RAISERROR ('New tables cannot be created in this database.', 16, 1)
 ROLLBACK
```

9.2.4 修改触发器

ALTER TRIGGER 触发器

9.2.5 删除触发器

DROP TRIGGER 触发器

9.2.6 查看触发器

```
sp_helptext trigger_name
sp_helptrigger table_name
```