# 第5章 存储过程

程序设计语言访问数据库有两种方式：嵌入式 SQL 编程和调用数据库端编程。嵌入式 SQL 编程就是将 SQL 语句嵌入在程序设计语言代码中，被嵌入的程序设计语言，如 C++、Java 等称为宿主语言，简称主语言；数据库端编程主要包括三种：存储过程、函数和触发器。

存储过程是一组用于完成特定功能的 T-SQL 语句集，经编译后存储在数据库服务器中，用户通过存储过程名和参数来调用它们。存储过程中也可以再调用存储过程。

存储过程是数据库端编程的一种方式，它具有执行速度快、能提高数据库安全性、可降低网络流量等优点，可以独立于前端程序进行编程和维护。

存储过程分为系统存储过程、用户自定义存储过程、扩展存储过程三种。系统存储过程是系统创建的，是以 sp_前缀命名的存储过程，常用的系统存储过程见附录 E。本章将介绍如何创建和使用用户自定义存储过程。

视频讲解

## 5.1 存储过程的语法

### 1. 创建存储过程

```
CREATE PROC[EDURE] 存储过程名
        [ { @参数名称 参数数据类型} [ = 参数的默认值] [ OUTPUT ] ] [ ,…n ]
        [ WITH ENCRYPTION ] [ WITH RECOMPILE ]
AS
        sql_statement
```

参数说明如下。

PROCEDURE：存储过程的关键字，既可以写全称，也可以缩写为四个字母 PROC。

存储过程名：自命名，必须符合数据库命名规则，而且在当前数据库唯一，最好有个统一的前缀，表明是存储过程，后面的命名也最好和内容相关，如 pro_login 表示登录验证存储过程。SQL Server 中的系统存储过程统一使用 sp_前缀命名。

参数：存储过程可以没有参数，也可以定义一个或多个参数。存储过程的参数分为输入参数和输出参数，输入参数用来向存储过程中传入值，输出参数则用于从存储过程中返回值。参数名必须以@开头，之后是类型和长度的说明，类似变量定义格式，多个参数之间以逗号分隔。存储过程参数定义时可以直接给出默认值。

OUTPUT：参数后面加 OUTPUT 表示该参数是输出参数，在存储过程中必须为输出参数赋值。

WITH ENCRYPTION：对存储过程语句加密，加密后任何用户无法查看存储过程的定义。

WITH RECOMPILE：每次执行存储过程时都要重新编译。但重新编译会影响执行速度，一般不用此选项。

AS：定义存储过程必需的关键字之一，AS后面是存储过程的语句体。

sql_statement：存储过程语句体，如果是单条语句直接写即可，多条语句需要用BEGIN…END语句组合成语句块。

**2. 执行存储过程**

按参数位置传递参数值：

```
EXEC[UTE]存储过程名 [ 参数值1,参数值2,…,参数值n ]
```

使用参数名传递参数值：

```
EXEC[UTE]存储过程名 [ @参数名 = 参数值 ] [ Default ][ ,…n ]
```

**说明**：执行存储过程传递参数的方法有两种：一种是按参数位置传递参数；一种是使用参数名传递参数。如果存储过程有多个参数，按参数位置传递参数时，参数传递的个数和顺序必须与参数定义的一致；使用参数名传递参数时，参数的前后顺序可以改变。

参数传递时要注意参数的数据类型，对于字符型和日期型参数要用单引号引起来。传递输出参数，后面一定要加OUTPUT关键字。

如果执行存储过程的语句是批处理的第一条语句，可以省略EXEC或EXECUTE命令，直接写存储过程的名字进行执行。

**3. 修改存储过程**

```
ALTER PROC[EDURE] 存储过程名
        [ { @参数名称 参数数据类型} [ = 参数的默认值] [ OUTPUT ] ] [ ,…n ]
        [ WITH ENCRYPTION ] [ WITH RECOMPILE ]
AS
        sql_statement
```

**说明**：修改存储过程也需要给出存储过程的完整定义，此操作可以用删除存储过程然后再重新创建的操作替代。数据库、表创建好后不能随意删除，因为删除会造成数据丢失，但存储过程不涉及数据丢失问题，可以随意删除重创建。

**4. 删除存储过程**

```
DROP  PROC[EDURE]存储过程名 [ ,…n ]
```

**5. 查看存储过程**

查看存储过程的参数及数据类型：

```
sp_help 存储过程名
```

查看存储过程的源代码：

```
sp_helptext 存储过程名
```

**说明**：使用系统存储过程可以查看存储过程的信息，但如果创建存储过程时使用了

[WITH ENCRYPTION]选项,则无法通过 sp_helptext 系统存储过程查看存储过程的源代码。常用系统存储过程见附录 E。

## 5.2 存储过程的例题

### 1. 无参数存储过程

**【例题 5-1】** 创建无参数存储过程。创建一个名为 pro_readersInfor 的存储过程,用于查询 bookDB 数据库中读者的信息。

代码:

```
USE bookDB
GO
CREATE PROCEDURE pro_readersInfor                    --定义过程名
AS
SELECT * FROM readers                                --过程体
```


图 5-1 查看存储过程

查看创建的存储过程如图 5-1 所示。

**说明**:前面进行 T-SQL 编程,程序没有存储在数据库服务器上,下次使用时还要重复写代码。而创建了存储过程,程序就会永久存储在数据库服务器上,在【对象资源管理器】窗口刷新可以看到创建好的存储过程。存储过程可多次重复执行,执行此存储过程的语句是:EXEC pro_readersInfor。

存储过程是创建在某个数据库中的,存储过程体的语句中不可以有打开数据库的语句。

**【例题 5-2】** 创建简单的不带参数的存储过程。创建一个名为 pro_readerInforBorr 的存储过程,用于查询 bookDB 数据库中读者周杰的借书信息,查询结果包括读者姓名、所借图书名称、借书日期、还书日期。

代码:

```
CREATE PROCEDURE pro_readerInforBorr
AS
  SELECT readers.ReaderName as 读者姓名 , books.BookName as 所借图书名称 ,
         Borrow.BorrowerDate as 借书日期 , borrow.ReturnDate as 还书日期
  FROM readers , books , borrow
  WHERE readers.ReaderID = borrow.ReaderID and books.BookID = borrow.BookID
        and readers.ReaderName = '周杰'
```

查看创建的存储过程如图 5-2 所示。

执行存储过程语句是:EXECUTE pro_readerInforBorr。

执行结果如图 5-3 所示。


图 5-2 查看创建的存储过程

**说明**:本例题的存储过程体中是三表连接查询的 SELECT 语句,只有一条语句可以省略 BEGIN…END。

### 2. 只有输入参数的存储过程

**【例题 5-3】** 创建带输入参数的存储过程。修改例题 5-2 中创建的名为 pro_readerInforBorr 的存储过程,使之可以查询


图 5-3　存储过程执行结果

任意读者的借书信息，读者姓名通过输入参数传递，查询结果包括读者姓名、所借图书名称、借书日期、还书日期。

代码：
```
ALTER PROCEDURE pro_readerInforBorr @NAME VARCHAR(8)          -- 定义输入参数
AS
SELECT readers.ReaderName as 读者姓名 , books.BookName as 所借图书名称 ,
       Borrow.BorrowerDate as 借书日期 , borrow.ReturnDate as 还书日期
FROM readers , books , borrow
WHERE readers.ReaderID = borrow.ReaderID and books.BookID = borrow.BookID
       and readers.ReaderName = @NAME                          -- 使用参数
```

执行存储过程查找不同人的借书信息：

```
EXECUTE pro_readerInforBorr '周杰'
EXECUTE pro_readerInforBorr '田湘'
EXECUTE pro_readerInforBorr '王海涛'
```

**说明**：修改后的存储过程不是把人名固定写在程序里，而是通过输入参数@NAME传递进去，每次执行存储过程时通过传递不同的参数可以查询不同人的借书信息，修改后的存储过程功能更加强大。存储过程参数定义时，如果参数用于和数据库表中的信息交互，参数的类型和长度应尽量与表中字段定义一致，参数的名称与表中字段名相同或者不相同都可以。

**【例题 5-4】** 创建名为 pro_booksInfor 的存储过程，其功能为在 bookDB 数据库的 books 表中查找指定图书类别，并且库存数量小于一定数量的图书信息，显示图书编号、图书名称、作者、图书类别、库存量，查询结果按库存量降序排列。

代码：
```
USE bookDB
GO
CREATE PROCEDURE pro_booksInfor
@booktype VARCHAR(50),
@kucunliang int              -- 定义两个输入参数，传入指定的图书类型和库存量
AS
  SELECT bookid as 图书编号 , bookname as 图书名称 , author as 作者 ,
         booktype as 图书类别 , kucunliang as 库存量
  FROM books
  WHERE booktype = @booktype and kucunliang <= @kucunliang
  order by kucunliang DESC
```

(1) 执行创建的 pro_booksInfor 存储过程，查询图书类别为“计算机类”、库存量小于 10 本的图书信息。

方法一：按参数位置传递参数。

```
EXEC pro_booksInfor '计算机类', 10
```

方法二：按参数名传递参数。

```
EXEC pro_booksInfor @booktype = '计算机类', @kucunliang = 10
```

方法三：用变量传递参数。

```
DECLARE @lx VARCHAR(50)
SET @lx = '计算机类'
EXEC pro_booksInfor @lx , 10
```

(2) 执行创建的 pro_booksInfor 存储过程，查询图书类别为“综合类”、库存量小于 5 本的图书信息。

方法一：按参数位置传递参数。

```
EXEC pro_booksInfor '综合类', 5
```

方法二：按参数名传递参数。

```
EXEC pro_booksInfor @kucunliang = 5 , @booktype = '综合类'
```

方法三：按用变量传递参数。

```
DECLARE @lx VARCHAR(50) , @sl int
SET @lx = '综合类'
SET @sl = 5
EXEC pro_booksInfor @lx , @sl
```

**说明**：执行存储过程传递输入参数可以直接传递常量，也可以定义变量传递，存储过程嵌入在前端程序中使用时，一般都是通过变量传递参数的，变量的值由人机交互输入。

**3. 有输入和输出参数的存储过程**

**【例题 5-5】** 创建存储过程 pro_borCount，输入任意读者的姓名，统计该读者借书的数量，借书数量通过存储过程的输出参数传递出来。

代码：

```
USE bookDB
GO
CREATE PROCEDURE pro_borCount
@NAME VARCHAR(8) ,                                          --定义输入参数
@sl int = 0  OUTPUT                                         --定义输出参数
AS
BEGIN
  DECLARE @reaID  int                                       --定义中间变量
  SELECT @reaID = readerID  FROM  readers  WHERE  ReaderName = @NAME
  SELECT @sl = COUNT( * )  FROM  borrow  WHERE  readerID = @reaID
END
```

执行存储过程：

```
DECLARE @sl int
EXEC pro_borCount '周杰', @sl OUTPUT
PRINT '周杰的借书数量是:' + ltrim(str(@sl ))
```

执行结果如图 5-4 所示。

**说明**：定义存储过程输出参数要加 OUTPUT 关键字。执行存储过程时，输入参数可

以用变量传递，也可以用常量传递，而输出参数必须用变量进行传递，也必须加 OUTPUT 关键字。输出参数传递出来后，可以在数据库平台上显示值，也可以应用到前台程序中。

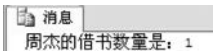

图 5-4　例题 5-5 的执行结果

本例题的存储过程中没有使用连接查询语句，而是用两条单表简单查询语句，先通过传入的读者姓名在读者表中查找该读者的读者编号，将读者编号保存在一个中间变量中，然后通过读者编号在 borrow 表中统计该读者的借书数量，统计结果保存在输出参数@sl 中。带输出参数的存储过程体中必须有为输出参数赋值的语句。

**【例题 5-6】** 编写存储过程，调整 stuDB 数据库 SC 表中的成绩，计算“C 语言程序设计”课程的不及格人数，如果不及格人数多于 5 人，给每位不及格学生加 3 分，再次计算不及格人数，如果不及格人数还是超出标准，继续增加，直到达到要求。

方法一代码：创建无参数存储过程。

```
USE stuDB
GO
CREATE PROCEDURE pro_grade1
AS
BEGIN
  /* 定义变量@cc 用于存储不及格人数, 变量@cno 用于存储课程号 */
  DECLARE @cc int , @cno int
  /* 先根据课程名查出课程号, 使用简单查询, 避免连接或嵌套查询语句较长 */
  SELECT @cno = Cno FROM Course WHERE Cname = 'C 语言程序设计'
  /* 统计该课程号的不及格人数, 存入变量@cc */
  SELECT @cc = COUNT( * ) FROM SC WHERE GRADE < 60 AND Cno = @cno
  WHILE @cc > 5                                    -- 循环,如果不及格人数大于 5,重复执行
    BEGIN
      UPDATE SC SET Grade = Grade + 3 WHERE Grade < 60 and Cno = @cno
      /* 更新成绩之后一定不要忘记重新统计平均成绩 */
      SELECT @cc = COUNT( * ) FROM SC WHERE GRADE < 60 AND Cno = @cno
    END
END
```

执行存储过程：EXEC pro_grade1。

**说明**：方法一代码编写的存储过程只能为“C 语言程序设计”课程调整成绩，条件也固定为不及格人数不能超过 5 人，程序用途单一。

方法二代码：创建带输入参数的存储过程，将课程名和不及格人数的限制作为输入参数传入。

```
CREATE PROCEDURE pro_grade2
@cname varchar(50) ,                               -- 输入参数:课程名
@rs int                                            -- 输入参数:不及格人数限制
AS
BEGIN
  DECLARE @cc int , @cno int
  SELECT @cno = Cno FROM Course WHERE Cname = @cname        -- 条件改为输入的变量
  SELECT @cc = COUNT( * ) FROM SC WHERE GRADE < 60 AND Cno = @cno
  WHILE @cc > @rs                                  -- 循环条件改为输入的变量
    BEGIN
```

```
        UPDATE SC SET Grade = Grade + 3 WHERE Grade < 60 and Cno = @cno
        SELECT @cc = COUNT( * ) FROM SC WHERE GRADE < 60 AND Cno = @cno
    END
END
```

执行存储过程：

限制“C 语言程序设计”课程只能少于 10 人不及格。

```
EXECUTE pro_grade2 'C 语言程序设计', 10
```

限制“数据库原理”课程不及格人数不可超过 5 人。

```
EXECUTE pro_grade2 '数据库原理', 5
```

**说明**：方法二代码的存储过程增加了两个输入参数，将课程名称和不及格人数作为输入参数传入，方法二代码比方法一代码的存储过程的功能强大，但遗憾的是没有看到执行结果的反馈。

方法三代码：创建既有输入参数，也有输出参数的存储过程。不仅灵活地将课程名称和不及格人数作为输入参数传入，也将更新次数、更新之后的不及格人数反馈出来。

```
CREATE PROCEDURE pro_grade3
@cname varchar(50) ,                          -- 输入参数:课程名
@rs int       ,                               -- 输入参数:不及格人数限制
@gxcs int OUTPUT,                             -- 输出参数:更新次数
@cc int OUTPUT                                -- 输出参数:不及格人数
AS
BEGIN
  DECLARE @cno int                            -- 去掉@cc 的变量定义
  SELECT @gxcs = 0 , @cc = 0
  SELECT @cno = Cno FROM Course WHERE Cname = @cname     -- 条件改为输入参数
  SELECT @cc = COUNT( * ) FROM SC WHERE GRADE < 60 AND Cno = @cno
  WHILE @cc > @rs                             -- 循环条件改为输入参数
    BEGIN
      UPDATE SC SET Grade = Grade + 3 WHERE Grade < 60 and Cno = @cno
      SELECT @cc = COUNT( * ) FROM SC WHERE GRADE < 60 AND Cno = @cno
      SET @gxcs = @gxcs + 1                   -- 累计更新次数
    END
END
```

执行存储过程：

限制“C 语言程序设计”课程不及格人数不可超过 5 人。

```
DECLARE @cname varchar(50) , @gxcs int , @cc int
SET @cname = 'C 语言程序设计'
EXECUTE pro_grade3 @cname , 3 , @gxcs OUTPUT , @cc OUTPUT
SELECT @cname + '不及格人数:' + STR(@cc,1) + ',更新次数:' + LTRIM(STR(@gxcs))
```

执行结果：

```
C 语言程序设计不及格人数: 3,更新次数: 6
```

**说明**：方法三代码的存储过程既有输入参数，也有输出参数，不仅灵活地将课程名称和

不及格人数作为输入参数传入，也能返回执行结果。本例题用三种方法演示了存储过程的创建和执行过程，也演示了参数的效果，实际开发时根据具体问题分析是否需要输入输出参数。

# 实验 10　存储过程练习

**一、实验目的**

了解存储过程的用途，掌握存储过程的创建和使用的基本语法。

**二、实验内容**

（1）按照图 4-9 所示的储蓄存款流程图将实验 9 中的 T-SQL 语句块改写为存储过程，将账号和存款金额作为输入参数传入，输出提示信息。运行检测存储过程功能。

（2）创建一个随机生成账号的存储过程 p_zh。以中国银行的借记卡为例，账号共有 19 位数字，前 12 位固定，后 7 位随机产生，并且唯一，因此需要产生 7 位随机数和 12 位固定数字连接在一起。随机数函数的用法见附表 A-3 中 Rand()函数的应用示例。

（3）按照图 5-5 所示的储蓄系统开户流程图编写存储过程，并调用随机生成账号的存储过程 p_zh，实现开户功能。

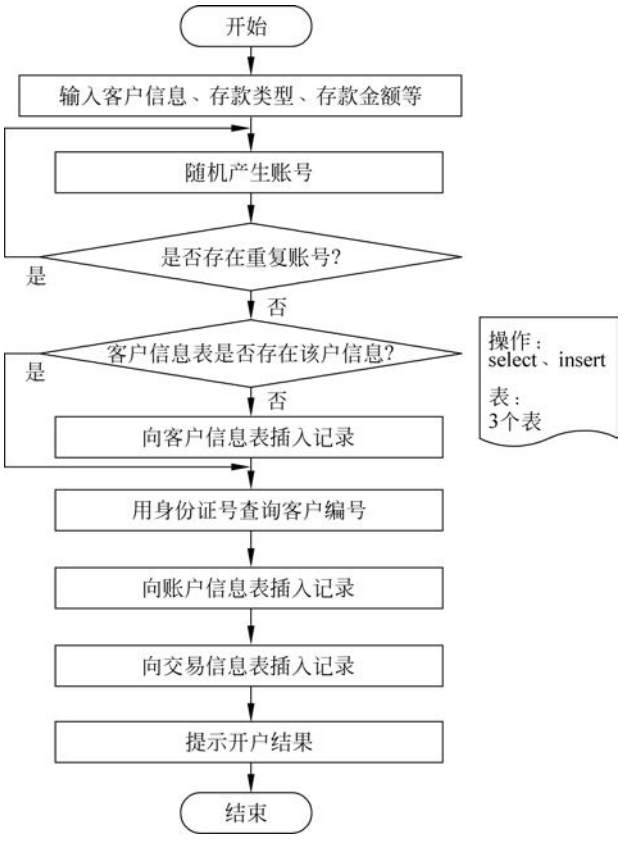


图 5-5　储蓄系统开户流程图

# 习题

习题解析

## 一、选择题

1. 创建存储过程的命令是(　　)。(多选)

A. create proc　　B. create function

C. create procedure　　D. create view

2. 删除存储过程的命令是(　　)。

A. drop view　　B. drop function　　C. drop database　　D. drop proc

3. 为使程序员编程时既可使用数据库语言又可使用常规的程序设计语言,数据库系统需要把数据库语言嵌入(　　)中。

A. 编译程序　　B. 操作系统　　C. 中间语言　　D. 宿主语言

4. SQL 语言具有两种使用方式,分别称为交互式 SQL 和(　　)。

A. 提示式 SQL　　B. 多用户 SQL　　C. 嵌入式 SQL　　D. 解释式 SQL

5. 数据库应用系统通常会提供开发接口,对于需要更新的数据,则以(　　)的方式供外部调用,并由提供者完成对系统中表的更新。

A. 基本表　　B. 存储过程　　C. 视图　　D. 触发器

6. 在 SQL Server 中用来显示数据库信息的系统存储过程是(　　)。

A. sp_dbhelp　　B. sp_db　　C. sp_help　　D. sp_helpdb

## 二、判断题

1. 参数化存储过程有助于保护程序不受 SQL 注入式攻击。(　　)

2. 在存储过程中不可以调用存储过程。(　　)

3. 用户自定义存储过程是指由用户创建的,能完成某一特定功能的可重用代码的模块或例程。(　　)

4. 存储过程独立于应用程序,可以单独修改,可以提高应用程序的可维护性。(　　)

5. 存储过程必须有参数。(　　)

6. 存储过程的输出参数有且只能有一个。(　　)

习题解析

## 三、填空题

1. 存储过程是一组完成特定功能的 T-SQL 语句集,经编译后存储在数据库中,用户通过_____和给出_____来调用它们。

2. SQL Server 存储过程分为三类:系统存储过程、_____和扩展存储过程。

3. SQL Server 中的许多管理活动都是通过系统存储过程实现的,系统存储过程以_____为前缀命名。

4. 创建存储过程 p_proc1 的语句是_____。

5. 定义存储过程的输出参数需要在参数变量定义之后加_____关键字。

6. SQL Server 的基本数据类型 char、varchar 和 text 中,不能用作存储过程参数的数据类型是_____。

7. 删除存储过程的命令是_____。

8. create、alter、drop 命令的作用分别为_____、_____、_____数据库对象。

9. database、table、procedure、function、view 关键字在数据库系统中的含义分别为_____、_____、_____、_____、_____。

10. 执行存储过程的语句是_____,可以简写为_____。

**四、简答题**

1. 请说明以下程序的功能,并写出使用它的语句。

```
CREATE PROCEDURE xsbm
@name varchar(8),
@bm varchar(10) OUTPUT
AS
BEGIN
  DECLARE @bmh char(4)
  SELECT @bmh = Department_ID FROM employee WHERE employee_name = @name
  SELECT @bm = Department_name FROM department WHERE Department_ID = @bmh
END
```

程序的功能:

使用它的语句:

2. 请把以下程序补充完整,并说明其功能。

```
CREATE PROCEDURE PRO_SUM
@N1 INT,@N2 INT,
@RESULT INT OUTPUT
AS
    SET (       ) = @N1 + @N2
```

程序的功能:

3. 请把以下程序补充完整,并说明其功能。

```
CREATE (         ) listEmployee
@sex varchar(2),
@salary money
AS
  SELECT * FROM  employee  WHERE sex = @sex and salary>@salary
```

程序的功能: