第5章 数据库编程技术基础

学习目标

正确理解和掌握 SQL Server 变量；掌握编写顺序结构、选择结构和循环结构程序的方法；掌握 SQL Server 函数的使用；掌握 SQL Server 2017 游标的使用方法；学会使用 Transact-SQL 对数据库进行应用编程，以掌握开发数据库应用系统的基本能力。

5.1 SQL 基础

在 SQL Server 2017 中，与 SQL Server 实例通信的所有应用程序都通过将 Transact-SQL 语句发送到服务器来实现数据的检索、操纵和控制等功能，因此 Transact-SQL 是 SQL Server 与应用程序之间的语言，是 SQL Server 的应用程序开发接口。

用 Transact-SQL 编写程序一般包括以下成分：常量、变量、表达式、函数、流程控制语句、事务和游标等。下面分别介绍这些组成部分。

5.1.1 Transact-SQL 的分类

Transact-SQL 分为 5 类，具体说明如下。

（1）数据定义语言（DDL）：用于创建数据库和数据库对象的命令，绝大部分以 CREATE 开头，如 CREATE TABLE 等。

（2）数据操作语言（DML）：用于操作数据库中的各种对象，对数据进行修改和检索。DML 语句主要有 4 种：SELECT（查询）、INSERT（插入）、UPDATE（更新）和 DELETE（删除）。

（3）数据控制语言（DCL）：用于控制数据库组件的存取许可、权限等的命令。

（4）事务管理语言（TML）：用于管理数据库中的事务的命令。

（5）其他语言元素：如标识符、数据类型、流程控制和函数等。

5.1.2 Transact-SQL 语法约定

1. Transact–SQL 语法格式约定

Transact-SQL 语法格式约定如表 5-1 所示。

2. 标识符

标识符就是用来定义服务器、数据库、数据库对象和变量等的名称。标识符可分为常规标识符和分隔标识符。SQL Server 为标识符制定了如下一系列命名规则。

表 5-1　Transact-SQL 语法格式约定

语法约定	说　明
大写	Transact-SQL 关键字
斜体	用户提供的 Transact-SQL 语法的参数
粗体	数据库名、表名、列名、索引名、存储过程、实用工具、数据类型名以及必须按所显示的原样输入的文本
下画线	当语句中省略了包含带下画线的值的子句时应用的默认值
|（竖线）	分隔括号或大括号中的语法项，只能选择其中一项
[ ]（方括号）	可选语法项，不要输入方括号
{ }（大括号）	必选语法项，不要输入大括号
[,... n]	指示前面的项可以重复 n 次，每一项由逗号分隔
[... n]	指示前面的项可以重复 n 次，每一项由空格分隔
[;]	可选的 Transact-SQL 语句终止符，不要输入方括号
<标签> ::=	语法块的名称，此约定用于对可在语句中的多个位置使用的过长语法段或语法单元进行分组和标记

（1）第一个字符必须是字母、下画线（_）、at 符号（@）和数字标记（#）。

（2）第一个字符后可以是字母、来自基本拉丁字母或其他国家/地区的十进制数字、美元符号（$）、下画线、at 符号和数字标记。

（3）标识符不能是 Transact-SQL 的保留字。

（4）不允许嵌入空格或其他特殊字符。

（5）包含的字符数必须为 1~128。

3. 续行

在很多情况下，Transact-SQL 语句都写得很长，可以将一条语句放在多行中编写，Transact-SQL 会忽略空格和行尾的换行符号，这样数据库开发人员不需要使用特殊的符号就可以编写长达数行的 Transact-SQL 语句，显著地提高了 Transact-SQL 语句的可读性。

例如：

```
SELECT Sname 姓名,Sdept 所在系
FROM student
WHERE Sno IN (SELECT Sno
              FROM sc
              WHERE Cno =(SELECT Cno
                          FROM course
                          WHERE Cname= '高等数学'
                         )
              )
GO
```

以上 SELECT 语句可以使用一行来表达，也可以使用多行。

4. 注释

在 Transact-SQL 中，注释语句有“--”（双减号）和“/*…*/”两种表示方法。

1）嵌入行内的注释语句

--（双减号）：创建单行文本注释语句。

【例 5-1】 创建单行文本注释语句。

```
--查询学生信息
SELECT * FROM student
```

2）块注释语句

在注释文本的起始处输入“/*”，在注释语句的结束处输入“*/”，就可以使两个符号间的所有字符成为注释，从而可以创建包含多行的块注释语句。

“/*”和“*/”一定要配套使用，否则将会出现错误，并且“/”必须和“*”连在一起，中间不能有空格。

【例 5-2】 在程序中创建块注释语句。

```
SELECT sname 姓名, Sdept 所在系
FROM student
/* WHERE Sno IN(SELECT Sno
                      FROM sc)
 */
```

其中，WHERE 子句将被作为注释处理，不再起作用。

5. 批处理

批处理是由一条或多条 Transact-SQL 语句构成的。SQL Server 2017 从批处理中读取所有语句，并将它们编译成可执行的单元（执行计划），然后 SQL Server 就一次执行计划中的所有语句，这里使用 GO 关键字结束批处理。

【例 5-3】 打印自定义变量。

```
DECLARE @aa int
SELECT @aa=10
PRINT @aa
GO
```

各语句的含义将在例 5-5 中再做介绍。

5.1.3 Transact-SQL 数据库对象命名方法

所有数据库对象名都是由 4 部分名称组成，格式如下。

```
[server_name.[database_name].[schema_name].
    | database_name.[schema_name].
| schema_name.
]
object_name
```

各部分说明如下。

server_name：连接的服务器名称或远程服务器名称。

database_name：SQL Server 数据库的名称。

schema_name：指定包含对象的架构名称。

object_name：对象的名称。

对象名的有效格式如表 5-2 所示。

表 5-2　对象名的有效格式

对象引用格式	说　　明
Server.database.schema.object	4 个部分的名称
Server.database..object	省略架构名称
Server..schema.object	省略数据库名称
Server…object	省略数据库和架构名称
Database.schema.object	省略服务器名
Database..object	省略服务器和架构名称
Schema.object	省略服务器和数据库名称
object	省略服务器、数据库和架构名称

5.1.4　常量

在程序运行过程中，其值不变的符号称为常量。常量格式取决于它所表示值的数据类型。根据常量值的不同类型，常量分为字符串常量、二进制常量、整型常量、实数常量、日期和时间常量、货币常量和唯一标识常量。

5.1.5　变量

变量在编程中占有重要的地位。利用变量可以存储临时性数据。SQL Server 2017 提供两种变量：局部变量和全局变量。

1. 局部变量

用户自定义的变量称为局部变量。局部变量用于保存特定类型的单个数据值的对象。

1）局部变量的定义

语法格式如下。

```
DECLARE 局部变量名 数据类型 [ ,... n]
```

其中，局部变量名必须以@开头，以与全局变量区别开。局部变量名必须符合有关标识符的命名规则。一个 DECLARE 语句可以同时声明多个变量，变量之间用逗号分隔。

【例 5-4】　定义一个整型变量。

```
--定义一个整型变量@Number
DECLARE @Number int
```

【例 5-5】　定义 3 个 varchar 类型的变量和一个整型变量。

```
/* 定义可变长度字符型变量@name，长度为 8；
可变长度字符型变量@sex，长度为 2；
小整型变量@age；
可变长度的字符型变量@address，长度为 50 */
DECLARE @name varchar(8),@sex varchar(2),@age smallint
DECLARE @address varchar(50)
```

2）局部变量的赋值

用 SET 或 SELECT 语句为局部变量赋值，它的语法格式如下。

```
SET @局部变量名 = 表达式[,...n]
SELECT @局部变量名 = 表达式[,...n] [FROM 子句] [WHERE 子句]
```

其中，使用 SELECT 语句为变量赋值时，如果省略了 FROM 子句和 WHERE 子句，就等同于 SET 语句赋值。如果有 FROM 子句和 WHERE 子句，若 SELECT 语句返回多个值，则将返回的最后一个值赋给局部变量。

【例 5-6】 打印信息系系主任的姓名。

```
DECLARE @name varchar(10)        --定义可变长度字符型的变量
SELECT @name='胡大智'             --给@name 赋值
PRINT '信息系系主任:'+@name       --显示@name 的内容
GO                               --批处理结束
```

执行结果如图 5-1 所示。

图 5-1 打印信息系系主任的姓名

【例 5-7】 以消息的形式返回学生选课数据库中的学生人数。

分析：以消息的形式就是使用 PRINT 语句。利用查询语句 SELECT 查询出学生人数，然后赋值给一个变量，最后用 PRINT 语句把变量打印出来。

在查询编辑器中执行如下语句。

```
USE 学生选课
GO
DECLARE @Number int
SELECT @Number=Count(*)

FROM student
PRINT '学生总人数为:'
PRINT @Number
GO
```

执行结果如图 5-2 所示。

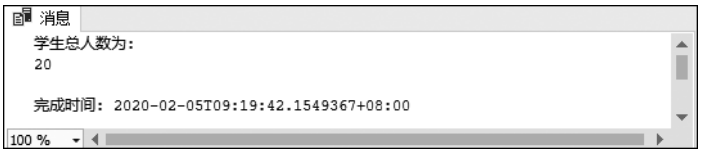

图 5-2　查询学生总人数

2. 全局变量

全局变量是由系统定义和维护的变量，用于记录服务器活动状态的一组数据。全局变量名由@@符号开始。用户不能创建全局变量，也不能使用 SET 语句修改全局变量的值。在 SQL Server 2017 中，全局变量以系统函数的形式使用。

例如，@@version 全局变量将返回当前 SQL Server 服务器的版本和处理器类型。@@language 全局变量将返回当前 SQL Server 服务器的语言。

5.1.6　表达式和运算符

表达式是标识符、值和运算符的组合。

1. 算术运算符

算术运算符用于对两个表达式进行数学运算，如表 5-3 所示。

表 5-3　算术运算符

运算符	含　义
+（加）	加法运算
−（减）	减法运算
*（乘）	乘法运算
/（除）	除法运算
%（取模）	返回一个除法运算的整数余数。例如，12 % 5 = 2，这是因为 12 除以 5，余数为 2

2. 赋值运算符

等号（=）是 Transact-SQL 中唯一的赋值运算符。

3. 位运算符

位运算符用于在两个表达式之间进行位操作，这两个表达式可以是整型数据中的任意数据类型。位运算符如表 5-4 所示。

表 5-4　位运算符

运算符	含　义
&（位与）	逻辑与运算（两个操作数）
|（位或）	位或（两个操作数）
^（位异或）	位异或（两个操作数）

4. 比较运算符

比较运算符用于测试两个表达式是否相同。除了 text、ntext 和 image 数据类型的表达式外，比较运算符可以用于所有的表达式。比较运算的结果有 3 个值：TRUE（真）、FALSE

（假）和 UNKNOWN（未知）。比较运算符如表 5-5 所示。

表 5-5　比较运算符

运算符	含　义	运算符	含　义
=	等于	<>	不等于
>	大于	!=	不等于（非 SQL-92 标准）
<	小于	!<	不小于（非 SQL-92 标准）
>=	大于或等于	!>	不大于（非 SQL-92 标准）
<=	小于或等于		

5. 逻辑运算符

逻辑运算符用于对某些条件进行测试，以获得其真实情况。逻辑运算返回 TRUE 或 FALSE 值。逻辑运算符如表 5-6 所示。

表 5-6　逻辑运算符

运算符	含　义
ALL	如果一组的比较都为 TRUE，那么结果就为 TRUE
AND	如果两个布尔表达式都为 TRUE，那么结果就为 TRUE
ANY	如果一组的比较中任何一个为 TRUE，那么结果就为 TRUE
BETWEEN	如果操作数在某个范围之内，那么结果就为 TRUE
EXISTS	如果子查询包含一些行，那么结果就为 TRUE
IN	如果操作数等于表达式列表中的一个，那么结果就为 TRUE
LIKE	如果操作数与一种模式相匹配，那么结果就为 TRUE
NOT	对任何其他布尔运算符的值取反
OR	如果两个布尔表达式中的一个为 TRUE，那么结果就为 TRUE
SOME	如果在一组比较中有些为 TRUE，那么结果就为 TRUE

6. 字符串串联运算符

加号（+）是字符串串联运算符，可将字符串串联起来。例如，'采购部主管: ' + '张立' 的结果就是'采购部主管:张立'。

7. 一元运算符

一元运算符只对一个表达式进行操作，该表达式可以是数值型数据中的任意数据类型。一元运算符如表 5-7 所示。

表 5-7　一元运算符

运算符	含　义
+	数值为正
−	数值为负
~	返回数字的非

8. 运算符优先级

当一个复杂的表达式中有多个运算符时，由运算符优先级决定运算的先后顺序。在计

算较低级别的运算符前，先对较高级别的运算符求值。运算符的优先级如表 5-8 所示。

表 5-8　运算符的优先级

级别	运　算　符
1	~（位非）
2	*（乘）、/（除）、%（取模）
3	+（正）、−（负）、+（加）、(+ 连接)、−（减）、&（位与）
4	=、>、<、>=、<=、<>、!=、!>、!<（比较运算符）
5	^（位异或）、|（位或）
6	NOT
7	AND
8	ALL、ANY、BETWEEN、IN、LIKE、OR、SOME
9	=（赋值）

5.2　流程控制语句

流程控制语句主要用于控制程序的顺序。下面逐个介绍 SQL Server 2017 提供的流程控制语句。

1. BEGIN…END 语句

BEGIN…END 语句用于将多个 Transact-SQL 语句组合为一个逻辑块，相当于一个语句，达到一起执行的目的。其语法格式如下。

```
BEGIN
   {
      语句 1
      语句 2
      …
   }
END
```

SQL Server 2017 允许嵌套使用 BEGIN…END 语句。

2. IF…ELSE 语句

IF…ELSE 语句用于实现程序选择结构。其语法格式如下。

```
IF   逻辑表达式
    { 语句块 1 }
[ ELSE
    { 语句块 2 }
]
```

其中，语句块可以是单个语句或一组语句。

IF…ELSE 语句的执行过程为：如果逻辑表达式的值为 TRUE，则执行语句块 1；如果有 ELSE 语句，且逻辑表达式的值为 FALSE，则执行语句块 2。SQL Server 2017 允许嵌套使用 IF…ELSE 语句。

【例 5-8】 在学生选课数据库中，查询 1 号课程的平均成绩是否超过 80 分，并显示相关信息。

分析：首先定义一个局部变量@avg_grade 来保存 1 号课程的平均成绩，对应的查询语句为“SELECT @Avg_Grade=AVG（Grade） FROM sc WHERE Cno=1”，然后将查询结果@Avg_Grade 与 80 进行比较，再显示比较结果。

在查询编辑器窗口中执行如下 Transact-SQL 语句。

```
USE 学生选课
GO
DECLARE @Avg_Grade NUMERIC(3,1)
SELECT @Avg_Grade =AVG(Grade) FROM sc WHERE Cno=1
IF @Avg_Grade>80
PRINT '选课平均成绩超过 80 分'
ELSE
PRINT '选课平均成绩不超过 80 分'
GO
```

执行结果如图 5-3 所示。

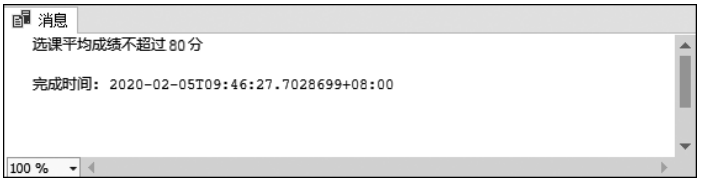

图 5-3 查询选修 1 号课程的平均成绩是否超过 80 分

3. WHILE、CONTINUE 和 BREAK 语句

WHILE 语句用于实现循环结构。如果指定的条件为真，就重复执行语句块，直到逻辑表达式为假。其语法格式如下。

```
WHILE   逻辑表达式
   BEGIN
        语句块 1
        [CONTINUE ]
        [BREAK ]
        语句块 2
END
```

参数说明如下。

BREAK：无条件退出 WHILE 循环。

CONTINUE：结束本次循环，进入下次循环，忽略 CONTINUE 后面的任何语句。

【例 5-9】 计算并输出 1+2+3+…+100 的值。

在查询编辑器窗口中执行如下 Transact-SQL 语句。

```
DECLARE @I int,@Sum int
SELECT @Sum=0
SELECT @I=1
```

```
WHILE @I<=100
  BEGIN
     SET @Sum= @Sum+@I
     SET @I=@I+1
  END
PRINT @Sum
```

执行结果为 5050。

【例 5-10】 求 1～100 的奇数的和。

```
DECLARE @I int, @Sum int
SELECT @Sum=0
SELECT @I=0
WHILE @I>=0
    BEGIN
        SET @I=@I+1
        IF @I>=100
            BEGIN
               SELECT'1~100 的奇数和'=@Sum
               BREAK
            END
           IF(@I%2)=0
           CONTINUE
        ELSE
           SELECT @Sum=@Sum+@I
    END
```

执行结果为 2500。

4. GOTO 语句

GOTO 语句用于让执行流程跳转到程序中的指定标签处，即跳过 GOTO 之后的语句，在标签处继续执行。其语法格式如下。

```
GOTO 标签名
     语句组 1
标签名:
     语句组 2
```

当程序执行到 GOTO 语句时，直接跳转到定义的标签名处，执行语句组 2，而忽略语句组 1。

【例 5-11】 利用 GOTO 语句，求 5 的阶乘。

```
DECLARE @i int,@jc int
SELECT @jc=1
SELECT @i=1
Lable1:
    SET @jc=@jc*@i
    SET @i=@i+1
    IF @i<=5
        GOTO Lable1
SELECT '5 的阶乘'=@jc
```