第8章
数据库编程

第8章 数据库编程

8.1 Transact-SQL基本语法

8.2 函数

8.3 存储过程

8.4 触发器

8.5 JDBC编程

8.1 Transact-SQL基本语法

- 标识符和命名规范
- 局部变量和全局变量
- 运算符和表达式
- 控制语句
- 游标

8.1 Transact-SQL基本语法

- 标识符和命名规范
  - 首字符为字母，之后可以是“_”、“@”、“#”、“$”及数字
  - 不允许空格和特殊字符
  - 不允许用保留字
  - 不区分大小写

8.1 Transact-SQL基本语法

- 2.局部变量和全局变量
  - 局部变量的声明：DECLARE

```
DECLARE @n1 int, @n2 int --声明两个整型变量
```

  - 局部变量的赋值：SET和SELECT

```
SET @n1=2, @n2 =3 --使用SET一次性给多个变量赋值时会报错
```

```
DECLARE @n1 int, @n2 int
SET @n1=2
SET @n2 =3
SELECT @n1 AS '变量1',@n2 AS '变量2'
```

  - 常见的全局变量的使用

```
SELECT @@version,@@servername,@@rowcount
```

8.1 Transact-SQL基本语法

- 3.运算符和表达式

```
DECLARE @n1 int, @n2 int,@result int --声明三个变量
SELECT @n1 = 10,@n2 = 5 --使用SELECT给变量赋值
SET @result = @n1 * @n2 --使用SET给结果赋值
PRINT @result --打印结果

DECLARE @c char(20) --声明字符变量长度为20
SET @c='Hello'
PRINT @c+',Transact SQL!!!'
```

8.1 Transact-SQL基本语法

- 4.控制语句
  - BEGIN … END：通常包含在其他控制语句中
  - IF … ELSE
  - CASE
  - WHILE

```
DECLARE @grade INT
SET @grade = 100
IF @grade >= 60
    PRINT '及格'
ELSE
    PRINT '不及格'
```

```
DECLARE @grade INT
SET @grade = 100

SELECT CASE
            WHEN @grade >= 90 THEN '优秀'
            WHEN @grade >= 80 THEN '良好'
            WHEN @grade >= 70 THEN '中等'
            WHEN @grade >= 60 THEN '及格'
            ELSE '不及格'
       END
       AS '成绩'
```

8.1 Transact-SQL基本语法

- 4.控制语句

```
DECLARE @count INT
SELECT @count = 0
WHILE @count < 10
BEGIN
    PRINT @count
    SELECT @count = @count + 1
END

PRINT 'loop finished, count = ' + CONVERT(VARCHAR(10), @count)
```

8.1 Transact-SQL基本语法

- 5.游标

游标是SQL Server的一种数据访问机制，允许用户访问单独的数据行

声明游标→打开游标→读取数据→关闭游标→删除游标

- 声明游标

```
DECLARE cursor_name CURSOR [ LOCAL | GLOBAL ]
    [ FORWARD_ONLY | SCROLL ]
    [ STATIC | KEYSET | DYNAMIC | FAST_FORWARD ]
    [ READ_ONLY | SCROLL_LOCKS | OPTIMISTIC ]
    [ TYPE_WARNING ]
    FOR select_statement
    [ FOR UPDATE [ OF column_name [ ,...n ] ] ]
[;]
```

- 打开游标

```
OPEN { { [ GLOBAL ] cursor_name } | cursor_variable_name }
```

- 读取数据

```
FETCH
[ [ NEXT | PRIOR | FIRST | LAST | ABSOLUTE { n | @nvar }
| RELATIVE { n | @nvar } ]
FROM ] { { [ GLOBAL ] cursor_name } | @cursor_variable_name }
[ INTO @variable_name [ ,...n ] ]
```

--into说明将读取的游标数据存放到指定的变量中

- 关闭游标

  CLOSE { { [GLOBAL] cursor_name } | cursor_variable_name }

- 删除游标

  DEALLOCATE { { [GLOBAL] cursor_name } | @cursor_variable_name }

```
DECLARE cursor_S CURSOR
FOR
SELECT * FROM Student;
OPEN cursor_S
FETCH NEXT FROM cursor_S
WHILE @@FETCH_STATUS=0
BEGIN
FETCH NEXT FROM cursor_S
END
CLOSE cursor_S
DEALLOCATE cursor_S
```

```
DECLARE @sno CHAR(10),@sname VARCHAR(20)
DECLARE cursor_S CURSOR
FOR
SELECT Sno,Sname FROM Student;
OPEN cursor_S
FETCH NEXT FROM cursor_S INTO @sno,@sname
WHILE @@FETCH_STATUS=0
BEGIN
PRINT @sno+@sname
FETCH NEXT FROM cursor_S INTO @sno,@sname
END
CLOSE cursor_S
DEALLOCATE cursor_S
```