实验 6　Transact-SQL 语言

一、目的和要求

本实验要求了解 T-SQL 的基本知识，掌握表达式中典型的函数应用，掌握 T-SQL 常用的语句以及简单应用。具体体现在：

（1）掌握 T-SQL 程序设计中 GO 语句的使用、变量的定义以及注释的使用；

（2）掌握 T-SQL 程序设计中流程控制语句的使用（IF…ELSE…，BEGIN…END，WHILE…BREAK…CONTINUE…，CASE…WHEN）；

（3）掌握 T-SQL 中常用系统函数和自定义函数的使用；

（4）掌握 PRINT 的用法；

（5）掌握类型转换函数的用法；

（6）掌握 T-SQL 程序设计中游标的使用。

二、背景知识

Transact-SQL(简称 T-SQL)语言是 Microsoft 公司在关系型数据库管理系统 SQL Server 中实现一种计算机高级语言，是微软对 SQL-92 标准的扩展。Transact-SQL 语言具有 SQL 的主要特点,同时扩充了数据类型，增加了变量与常量、运算符、函数、流程控制和注释等语言元素，并对标准 SQL 的存储过程、触发器和游标等机制进行了一定的修改，使得其功能更加强大。

Transact-SQL 中提供了变量定义，赋值操作，流程控制，函数等语句，供用户使用。

（1）SQL Server 2005 支持的系统数据类型

SQL Server 2005 支持的系统数据类型如表 6.1 所示。

表 6.1 SQL Server 2005 提供的系统数据类型

分类	数据类型定义符
字符类型	Char、Varchar、Text
整数类型	Bigint、Int、Smallint、Tinyint
Unicode 字符类型	Nchar、Nvarchar、Ntext
精确小数类型	Decimal、Numeric
浮点数值类型	Float、Real
二进制数据类型	Binary、Varbinary、Image
货币数据类型	Money、Smallmoney
位数据类型	Bit
日期时间类型	Datetime、Smalldatetime
其他数据类型	Cursor、Xml、Sql_variant、Table、Timestamp、Uniqueidentifier

（2）用户自定义数据类型

用户自定义数据类型是建立在 SQL Server 系统数据类型基础上的。一般情况下，当多个表的列中要存储同样类型的数据，且想确保这些列具有完全相同的数据类型、长度、取值范围和是否等特点时，可使用用户定义数据类型。

SQL Server 为用户提供了两种方法来创建自定义数据类型即使用 SQL Server Management Studio 创建用户自定义数据类型和调用系统存储过程 sp_addtype 创建用户自定义数据类型。

（3）批处理 GO

GO 用于向 SQL Server 实用工具发出一批 Transact-SQL 语句结束的信号,即批处理。

批处理是一组 SQL 语句的集合，以 GO 结束。批处理中的所有语句被一次提交 SQL Server 2005，SQL Server 2005 将这些语句编译为一个执行单元，称为 SQL Server 2005 执行计划。注意 GO 命令和 Transact-SQL 语句不能在同一行中。

在一个批处理中，如果出现编译错误(如某条语句存在语法错误)，SQL Server 2005 将取消整个批处理内所有语句的执行。

（4）注释

注释可用于对代码进行说明或暂时禁用正在进行诊断的部分 Transact-SQL 语句和批（前面的例子中已使用了注释）。注释可用于说明复杂的计算或解释编程方法，使用注释可以使程序清晰可读，有助于以后的管理维护。SQL Server 支持行注释和块注释两种方式：

行注释符：

--注释内容

说明：

1) 以双减号开始直到本行结束的全部内容都被认为是注释内容。

2) 行注释可以单独一行，也可以跟在 SQL 语句之后，注释内容中还可以有双减号(允许嵌套)，双减号之后也可以没有内容。

块注释符：

/*注释内容*/

说明：

1) 以“/*”开始不论多少行，直到“*/”之间的所有文字都被作为注释内容。

2) 块注释可以从一行开头开始，也可以跟在 SQL 语句之后开始，注释内容中还可以有“/*”字符组合，也可以有单个“*”“/”字符，但不能有“*/”组合(不允许嵌套)，中间可以没有注释内容。

（5）变量及其定义

在 SQL Server 中，变量分为局部变量和全局变量。全局变量名称为@@字符，由系统定义和维护。局部变量前面有一个@字符，由用户定义和使用。

局部变量使用 DECLARE 语句定义，并且指定变量的数据类型，然后可以使用 SET 或 SELECT 语句为变量初始化；局部变量必须以“@”开头，而且必须先声明后使用。其声明格式如下：

DECLARE { @local_variable data_type }[,…n]

其中：

@local_variable：局部变量名，局部变量名必须以 at 符（**@**）开头；

data_type：局部变量类型，可以是 SQL Server 支持的系统数据类型，也可以是用户自定义的数据类型。但变量不能是 text、ntext 或 image 数据类型。

初始化格式如下：

SET { @local_variable = expression }　[,...n]

或者 **SELECT { @local_variable = expression } [,...n]**

其中，参数@local_variable 是给其赋值并声明的局部变量，参数 expression 是任何有效的 SQL Server 表达式。

（6）流程控制语句

Transact-SQL 提供了用于更改语句执行顺序的流程控制语句，利用它们能够更好地组织语句，方便地实现程序的功能。流程控制语句主要用来控制 SQL 语句、语句块或者存储过程的执行流程，实现选择、循环等结构，以实现结构化编程，并完成比较复杂的操作。

Transact-SQL 提供的流程控制语句如表 6.2 所示。

表 6.2 流程控制语句及功能

流程控制语句	功能说明
BEGIN…END	定义语句块
IF…ELSE	条件选择语句（二分支语句）
CASE	多分支语句
WHILE	循环语句
BREAK	跳出循环语句
CONTINUE	重新开始循环语句
GOTO	无条件跳转语句
RETURN	无条件退出语句
WAITFOR	等待延迟语句

（7）执行动态生成的、含有变量的 SQL 字符串的方法：

使用字符串串联运算符+，将若干字符串和变量串联，动态生成可执行的 SQL 字符串。每个字符串表达式长度不得超过 8000 个字节。执行方式为：

EXEC('string')

在 EXECUTE 语句执行前，SQL SERVER 不会编译 EXECUTE 语句内的语句。

（8）函数

函数是由一条或多条 Transact-SQL 语句组成的集合，用于完成某个特定的功能。在 Transact-SQL 语言中也提供了大量的函数，用来执行一些特殊的运算和操作。SQL Server 2005 支持两种函数类型：系统函数和用户定义函数。

系统函数是由系统预定义的函数，是 Transact-SQL 语言的一部分。SQL Server 2005 提供了丰富的具有执行某些功能的系统函数。SQL Server 2005 提供的系统函数分类见表 6.3。

表 6.3 SQL Server 2005 系统函数分类

系统函数类别	功能说明
聚合函数	执行的操作是将多个值合并为一个值，进行统计如求和、最大值、最小值、平均值、计数等
配置函数	返回当前的配置信息
游标函数	返回有关游标的信息
日期和时间函数	对日期和时间输入值进行处理
数学函数	对作为函数参数提供的输入值执行计算
元数据函数	返回有关数据库和数据库对象的信息
其他函数	常用的有类型转换函数
行集函数	返回行集，这些行集可用在Transact-SQL语句中表引用所在的位置
安全函数	返回有关用户和角色的信息
字符串函数	对字符串（char 或 varchar）输入值执行相应操作
系统统计函数	返回系统的统计信息
文本和图像函数	对文本或图像输入值或列执行操作，返回有关这些值的信息

具体函数及其功能请参加联丛书。

用户自自定义函数是用户根据需要定义的可重复调用的函数。SQL Server 2005 支持 3 种用户定义函数即：标量函数、表值函数和多语句表值函数。

用户可以通过 SQL Server Management Studio 创建和管理用户自定义函数，也可以通过 CREATE FUNCTION 创建用户自定义函数，它是由一个或多个 Transact-SQL 语句组成的子程序，该子程序接受参数、执行操作并将操作结果返回。返回值可以是单个标量值或结果集。其中用 CREATE FUNCTION 创建标量函数的简化格式如下：

```
CREATE FUNCTION function_name
( [ { @parameter_name [ AS ]   parameter_data_type [ = default ] }
   [ ,...n ]
 ]
)
RETURNS return_data_type
   [ WITH <function_option> [ ,...n ] ]
   [ AS ]
   BEGIN
         function_body
         RETURN scalar_expression
   END
[ ; ]
```

其中：function_name 表示用户自定义函数的名称；@parameter_name 和 parameter_data_type 为函数所带的参数及其数据类型；return_data_type 表示返回值的类型；

function_body 表示函数体；scalar_expression 表示函数要返回的标量值，必须与 return_data_type 的类型一致。

（9）游标的定义和使用

在大型数据库中，游标提供了一种能从包括多条数据记录的结果集中每次提取一条记录的机制。游标可以被看做一个指针，它与一个查询结果集（一个缓冲区）相关联。游标可以指向结果集的任何位置，以便对指定位置的记录进行处理。

在 SQL Server 2005 中，根据运行的位置和用途分类、用途等的不同，可以将游标分为：T-SQL 游标、应用程序编程接口服务器游标和客户端游标。其中应用程序编程接口服务器游标支持 OLEDB、ODBC、ADO 和 DB-library 中使用的游标函数；客户端游标即在客户端实现的游标；而 T-SQL 游标就是在 T-SQL 脚本程序、存储过程、触发器中对 SELECT 语句返回的结果集进行逐行逐字段处理，把一个完整的数据表按行分开，一行一行的逐一提取记录，并从这一记录行中逐一提取各项数据。以上三种游标中，T-SQL 游标、应用程序编程接口服务器游标都在服务器上实现，所以将它们统称为服务器游标。以下重点介绍 T-SQL 游标。

在 SQL Server 2005 中，T-SQL 游标的使用过程如下：声明游标→打开游标→检索游标→关闭游标→释放游标，如图 6.1 所示。

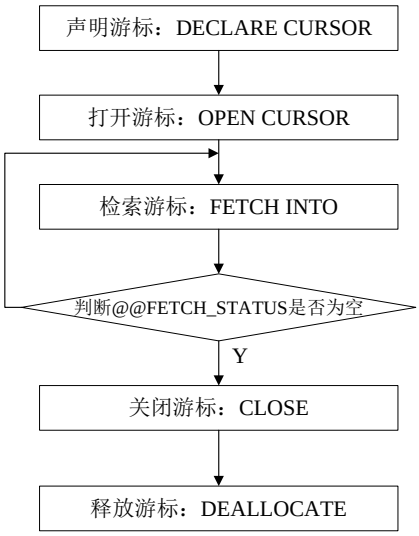

图 6.1 游标的使用过程

1）说明游标

DECLARE <游标名> CURSOR　FOR <SELECT 语句>

2）打开游标

OPEN <游标名>

功能说明：打开游标实际上是执行相应的 SELECT 语句，把所有满足查询条件的记录从指定表取到缓冲区中，这时游标处于活动状态，指针指向查询结果集中第一条记录。

3）推进游标指针并提取当前记录

FETCH [[NEXT|PRIOR|FIRST|LAST] FROM]<游标名>

INTO <变量 1>,<变量 2>...;

功能说明：

（1）按指定方向推动游标指针，然后将缓冲区中的当前记录取出来送至变量供系统进一步处理。

（2）NEXT|PRIOR|FIRST|LAST：指定推动游标指针的方式

NEXT：向前推进一条记录；PRIOR：向回退一条记录；FIRST：推向第一条记录；LAST：推向最后一条记录。缺省值为 NEXT

4） 关闭游标

CLOSE <游标名>

功能说明：

（1） 关闭游标，释放结果集占用的缓冲区及其他资源。

（2） 游标被关闭后，就不再和原来的查询结果集相联系

（3） 被关闭的游标可以再次被打开，与新的查询结果相联系

5）游标函数

（1） 在使用游标时，经常使用两个游标函数 @@CURSOR_ROWS 和 @@FETCH_STATUS 获取游标的相关状态。

（2） @@CURSOR_ROWS 表示当前打开的游标中存在的记录数量。

（3） @@FETCH_STATUS 返回被 FETCH 语句执行的最后游标的状态。

三、实验内容

通过查询编辑器，设置 XSGL 为当前数据库，调试实验步骤中提供的相关语句，熟悉 SQL SERVER 提供的流程控制语句。

四、实验步骤

1．定义一个实型变量，并将其值输出

```
DECLARE  @f   float
SET @f=456.256
PRINT  CAST( @f  AS  varchar(12))
```

2．定义一个字符变量，并将其处理后输出

```
DECLARE @mynumber char(20)
SET @mynumber='test'
SELECT 'COMPUTER'+@mynumber AS '计算结果'
```

3．根据授课班号自定义变量，查询符合要求的学生成绩

```
DECLARE @cn char(6),@f float
SET @cn='218801'
SET @f=85
```

```
SELECT  * FROM  sc  WHERE CNO=@cn  AND  GRADE>=@f
```

4．BEGIN……END……的使用

```
BEGIN
    DECLARE @myval  float
    SET  @myval=456.234
    BEGIN
        PRINT '变量@myval 的值为'
        PRINT  CAST(@myval  AS  char(12))
    END
END
```

5．利用 CASE 查看学生的成绩等级

```
SELECT SNO, CNO,
CASE
    WHEN GRADE>=80 THEN 'A'
    WHEN GRADE>=70 THEN 'B'
    WHEN GRADE>=60 THEN 'C'
    ELSE  'D'
    END
AS 等级
FROM sc
```

说明：SELECT 后的列表值，也可以来源于一个判断结构

6. 创建一个视图，统计每个学生的学习情况。若其平均成绩超过 90，则其学习情况为优秀；若其平均成绩在 80 和 90 之间，则其学习情况为优秀良好；依次类推。

```
CREATE VIEW student_grade
AS
SELECT SNO,SNAME,
    CASE
        WHEN (SELECT AVG(GRADE) FROM sc WHERE SNO=STUDENT.SNO)>90  THEN '优秀'
        WHEN  (SELECT AVG(GRADE)  FROM sc WHERE SNO=STUDENT.SNO)>80 THEN '良好'
        WHEN  (SELECT AVG(GRADE)  FROM sc WHERE SNO=STUDENT.SNO)>70 THEN '中等'
        WHEN  (SELECT AVG(GRADE)  FROM sc WHERE SNO=STUDENT.SNO)>60 THEN '合格'
```

```
        WHEN    (SELECT AVG(GRADE)  FROM sc WHERE SNO=STUDENT.SNO)<60
THEN '不合格'
      END
   AS '成绩等级'
   FROM student
```

7．利用 WHILE 结构，求:1+2+3+..+100 的值

```
   DECLARE @s int,@i int
   SET @i=0
   SET @s=0
   WHILE  @i<=100
     BEGIN
       SET  @s=@s+@i
       SET  @i=@i+1
     END
   PRINT '1+2+3+…+100='+cASt(@s AS char)
```

8．利用 IF…ELSE，查询课程号为'234901'的学生的总平均成绩，如果大于 90，输出优秀，在 80-90 之间，输出优良，其它输出一般

```
   DECLARE  @chegji  int
   SELECT  @chegji=AVG(GRADE)  FROM  sc  WHERE CNO='234901'
   IF   @chegji>90
          PRINT '优秀'
   ELSE
     BEGIN
       IF     @chegji>80
          PRINT '优良'
       ELSE
          PRINT '一般'
     END
```

9．定义一函数，实现如下功能，对于一给定的学号 studentSNO，查询该值在 student 中是否存在，存在返回 1，不存在返回 0。

```
   CREATE  FUNCTION  check_id
   (@studentSNO  char(8))
   RETURNS  integer
   AS
     BEGIN
```

```
DECLARE   @num   int
IF(EXISTS(SELECT * FROM student WHERE SNO=@studentSNO))
    SET @num=1
ELSE
    SET   @num=0
return   @num
END
```

10．使用下面的 SQL 语句调用第 9 题的函数。要求：当向表 student 中插入一条记录时，首先调用函数 check_id，检查该记录的学号在表 student 中是否存在，不存在，才可以插入。

```
DECLARE @num int
SET   @num=dbo.check_id('20081200')
IF @num=0
  INSERT   INTO   student(SNO,SNAME)   values('20081200','张文')
```

11．求学生选课的门数，列出其姓名及选课门数

```
SELECT SNAME ,( SELECT COUNT(*) FROM sc WHERE SNO=student.SNO ) AS 选课门数
FROM student
```

说明：SELECT 后的列表值，可以来源于内嵌 sql 语句,但必须保证内嵌 sql 语句只能返回一个值

12．根据课程号动态查询学生的选课人数

```
DECLARE @cno char(8)
DECLARE    @sql    varchar(8000)
SET @cno='218801'
--动态生成 SQL 语句
SET @sql='SELECT COUNT(*) FROM sc   WHERE CNO='''+@cno+''''
EXEC(@sql)
SET @cno='203402'
SET @sql=' SELECT COUNT(*) FROM sc   WHERE CNO='''+@cno+''''
EXEC (@sql)
```

说明：可以利用变量以及字符串连接字符‘＋’，动态生成 SQL 语句，达到依据条件，进行数据库动态查询目的。其中，连接字符串中若包含单引号’字符，则必须用两个’’,表示一个单引号’字符。

13．用游标结合循环，输出全校各种姓氏及其人数

```
--记录数
DECLARE @count int
--循环变量
DECLARE @i int
--各种姓氏和人数
DECLARE @xing char(6),@rs int
--定义游标
DECLARE @name_cursOR CURSOR
--给游标赋值
SET @name_cursOR =CURSOR LOCAL SCROLL FOR
          SELECT DISTINCT SUBSTRING(SNAME,1,1) AS xing, count(*) AS rs
          FROM student
          GROUP BY SUBSTRING(SNAME ,1,1)
--循环变量初始化
SET @i=0
--打开游标
OPEN @name_cursOR
SET   @count=@@CURSOR_ROWS
--@@CURSOR_ROWS 表示游标对应的记录集中的记录个数
PRINT @count
--判断游标中是否有记录
IF @count<= 0   --没有纪录
   BEGIN
      PRINT '没有记录'
   END
--存在记录
ELSE   BEGIN
       --循环处理每一条记录
            WHILE(@i<@count)
            BEGIN
               FETCH NEXT FROM @name_cursOR INTO   @xing,@rs
               PRINT CONVERT(CHAR(20),@i)+@xing+CAST(@rs AS CHAR(20))
               SET @i=@i+1
            END
        CLOSE @name_cursOR     --关闭游标
      END
```

说明：本例中用游标存放全校各种姓氏的人数统计结果，然后，通过循环的方式，进行输出。游标更具体的应用请看下一题的三维透视表的查询。

14．利用游标结合循环，统计学院各种姓氏的人数

```
DECLARE   @count int
DECLARE   @i int
DECLARE   @xing char(6)
DECLARE   @rs int
DECLARE   @sql varchar(8000)
SET @sql=''
SET @i=0
DECLARE @name_cursOR CURSOR
SET @name_cursOR =CURSOR LOCAL SCROLL FOR
          SELECT DISTINCT SUBSTRING(SNAME,1,1) AS xing,count(*) AS rs
          FROM student
          GROUP BY SUBSTRING(SNAME,1,1)
OPEN @name_cursOR
SET @count=@@CURSOR_ROWS
IF @count<= 0
   BEGIN
      PRINT '没有记录'
   END
ELSE
   BEGIN
      WHILE(@i<@count-300)
      BEGIN
          FETCH @name_cursOR INTO   @xing,@rs
--动态生成前 300 个姓氏 SQL 语句
          SET @sql=@sql+'(SELECT COUNT(*) FROM student WHERE DNO=a.DNO
AND SUBSTRING(SNAME,1,1)='''+@xing+''') AS '+@xing+','
          SET @i=@i+1
      END
      FETCH @name_cursOR INTO   @xing,@rs
--动态生成第 301 个姓氏 SQL 语句
      SET @sql=@sql+'(SELECT count(*) FROM student WHERE DNO=a.DNO
AND SUBSTRING(SNAME,1,1)='''+@xing+''') AS '+@xing
     CLOSE @name_cursOR
     PRINT   @sql
     SET    @sql='SELECT   DNAME   ,'+@sql+'FROM   (SELECT   DISTINCT
STUDENT.DNO,DNAME FROM student,DEPT WHERE   student.DNO=DEPT.DNO) AS a'
```

```
    EXEC(@sql)
END
```

说明：由于字符变量最多只能存储 8000 个字符，所以本例只统计了前 301 个姓氏。

思考模仿题：

利用 CASE 实现学生表中学院编号到学院名称的映射

```
SELECT SNO,SNAME,
   CASE
     WHEN    SUBSTRING(SNO,5,1)='1'    THEN '机电学院'
     WHEN    SUBSTRING(SNO,5,1)='2'    THEN '信息学院'
     WHEN    SUBSTRING(SNO,5,1)='3'    THEN '工商学院'
   END
FROM    student
```

2. 定义一函数，实现如下功能，对于一给定的学号 studentSNO 和课程号 studentCNO 查询该值在 student 和 course 中是否存在，存在返回 1，不存在返回 0。

```
CREATE  FUNCTION  check_sc(@studentSNO  char(8),@studentCNO  CHAR(8))
RETURNS  integer
AS
      BEGIN
      DECLARE  @num  int
      IF( EXISTS(SELECT * FROM student WHERE SNO=@studentSNO) AND
          EXISTS(SELECT * FROM course  WHERE CNO=@studentCNO) )
          SET @num=1
      ELSE
          SET  @num=0
      RETURN  @num
      END
```

3．根据教师名自定义变量，查询符合要求的教师授课情况

```
DECLARE @tname char(6)
SET @tname='张聪'
SELECT  * FROM  course  WHERE TNAME=@tname
```

4．求授课班号及选修该授课班号的学生人数

```
SELECT    CNO  AS  授课班号 ,
 ( SELECT COUNT(*)  FROM  sc    WHERE CNO=course.CNO)    AS 选课人数
```

```
FROM    course
```

5．定义一函数，根据学号返回学生的选课门数（参考 INSERT 触发器）

```
CREATE    FUNCTION    COUNT_course(@studentSNO    char(8)) RETURNS    integer
AS
        BEGIN
        DECLARE    @num int
        SELECT @num=COUNT(*) FROM sc WHERE SNO=@studentSNO
        RETURN    @num
        END
```

请同学编写 SQL 语句调用上述函数，查询'20002059'号学生的选课门数。

6．修改学生的成绩，若大于 80 分，增加 5 分，否则，增加 8 分

```
UPDATE sc
SET GRADE=
(
        CASE
        WHEN GRADE>=80     THEN    GRADE+5
        ELSE GRADE+8
        END
 )
```

思考题

（1） 按课程名称，统计其平均分，列出其课程名称和平均分

（2） 求每个学生选课的门数及其平均分，列出其姓名、课程门数及平均分

（3） 定义一函数，依据学生的姓名，查询其所选课程的门数

（4） 根据学院名称，统计学生人数，列出学院名称和学生人数

（5） 若存在学号为‘20081200’的学生，则显示其姓名，否则，显示相应提示信息

（6） 查找每个学生超过他选修课程平均成绩的课程相关信息，列出学号，课程号，成绩，选课平均成绩

（7） 创建一视图，统计每门课程的学习情况。若课程平均成绩超过 90，则其学习情况为优秀；若课程平均成绩在 80 和 90 之间，则其学习情况为优秀良好；依次类推。

（8） 利用游标结合循环，统计各门课程的各种分数的人数。