



实验报告

实验（六）

姓 名	熊恪峥
学 号	22920202204622
日 期	2023年6月13日
学 院	信息学院
课程名称	数据库

实验（六）

目录

1	实验目的	1
2	实验内容	1
3	实验步骤	1
4	实验总结	10

1 实验目的

通过本次实验，要求掌握变量定义，流程控制，存储过程，存储函数，游标等内容。

2 实验内容

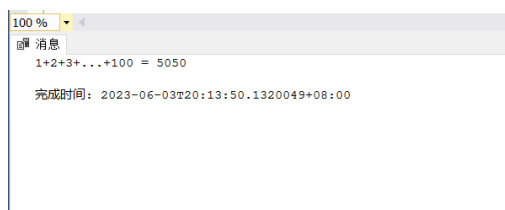
1. 变量的声明和使用，掌握@@ERROR、@@ROWCOUNT、@@IDENTITY等全局变量的使用。2. 使用BEGIN...END、IF...ELSE...、WHILE...CONTINUE...BREAK...、CASE等流程控制语句。
2. 使用存储过程。
3. 使用系统函数和用户自定义函数。
4. 使用游标处理数据。

3 实验步骤

1. 用T-SQL语言完成 $1 + 2 + 3 + \dots + 100$ ，并使用@@ERROR判断是否执行成功，如果成功则输出值，否则打印执行失败。

```
DECLARE @sum INT
SET @sum = 0
DECLARE @i INT
SET @i = 1
WHILE @i <= 100
BEGIN
    SET @sum = @sum + @i
    SET @i = @i + 1
END
IF @@ERROR <> 0
BEGIN
    PRINT '执行失败'
END
ELSE
BEGIN
    PRINT '1+2+3+...+100 = ' + CAST(@sum AS VARCHAR(10))
END
```

图 1: 运行结果



2. 更新STUDENTS表中sid为80000759500的学生的email为ddff@sina.com，并通过@@ROWCOUNT判断是否有数据被更新，如果没有则打印警告。

```
use School
```

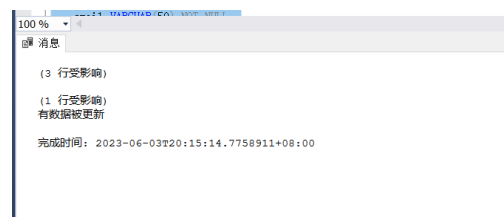
```
CREATE TABLE STUDENTS (  
    sid VARCHAR(20) PRIMARY KEY,  
    name VARCHAR(50) NOT NULL,  
    gender VARCHAR(2) NOT NULL,  
    birthday DATE NOT NULL,  
    email VARCHAR(50) NOT NULL,  
    phone VARCHAR(20) NOT NULL,  
    address VARCHAR(100) NOT NULL  
);
```

```
INSERT INTO STUDENTS (sid, name, gender, birthday, email, phone, address) VALUES  
( '80000759500', '张三', '男', '1999-01-01', 'zhangsan@qq.com', '13888888888', '福建省厦门市思明区厦  
( '80000759501', '李四', '女', '1999-02-02', 'lisi@qq.com', '13999999999', '福建省厦门市思明区岛内'),  
( '80000759502', '王五', '男', '2000-03-03', 'wangwu@qq.com', '15888888888', '福建省厦门市集美区杏林
```

```
UPDATE STUDENTS SET email = 'ddff@sina.com' WHERE sid = '80000759500'
```

```
IF @@ROWCOUNT = 0  
BEGIN  
    PRINT '警告！没有数据被更新'  
END  
ELSE  
BEGIN  
    PRINT '有数据被更新'  
END
```

图 2: 运行结果



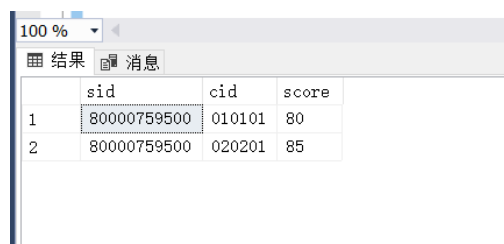
3. 使用IF...ELSE...语句，查询STUDENTS表中学号为800007595的学生，如果学生存在，则输出学生的各科成绩，否则打印查无此人。

```
CREATE TABLE COURSES (  
    cid VARCHAR(20) PRIMARY KEY,  
    cname VARCHAR(50) NOT NULL,  
    credit INT NOT NULL  
);
```

```
INSERT INTO COURSES (cid, cname, credit) VALUES  
( '010101', '高等数学', 4),  
( '020201', '大学英语', 3),  
( '030301', '物理实验', 2);
```

```
CREATE TABLE SC (  
    sid VARCHAR(20) NOT NULL,  
    cid VARCHAR(20) NOT NULL,  
    score INT NOT NULL,  
    PRIMARY KEY(sid, cid),  
    FOREIGN KEY(sid) REFERENCES STUDENTS(sid),  
    FOREIGN KEY(cid) REFERENCES COURSES(cid)  
);  
  
INSERT INTO SC (sid, cid, score) VALUES  
( '80000759500', '010101', 80),  
( '80000759500', '020201', 85),  
( '80000759502', '010101', 75),  
( '80000759502', '030301', 90),  
( '80000759501', '010101', 95),  
( '80000759501', '020201', 80);  
  
IF (EXISTS(SELECT * FROM STUDENTS WHERE sid='80000759500'))  
BEGIN  
    SELECT * FROM SC WHERE sid='80000759500';  
END  
ELSE  
BEGIN  
    PRINT '查无此人';  
END
```

图 3: 运行结果



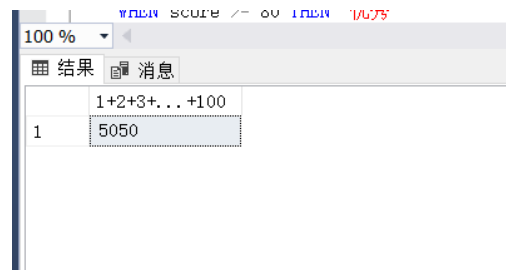
	sid	cid	score
1	80000759500	010101	80
2	80000759500	020201	85

4. 使用WHILE语句计算 $1 + 2 + 3 + \dots + 100$

```
DECLARE @sum INT  
SET @sum = 0  
DECLARE @i INT  
SET @i = 1  
WHILE @i <= 100  
BEGIN  
    SET @sum = @sum + @i  
    SET @i = @i + 1  
END  
SELECT @sum AS '1+2+3+...+100'
```

5. 使用CASE语句，查询学号为800007595所选择的课程号为10042的成绩，如果为80分或以上，打印优秀，如果在60—80分之间则打印及格，否则打印不及格。

图 4: 运行结果



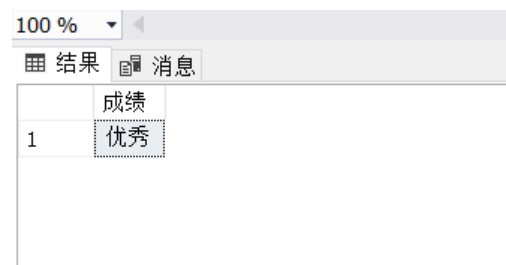
	1+2+3+...+100
1	5050

```

SELECT CASE
    WHEN score >= 80 THEN '优秀'
    WHEN score >= 60 AND score < 80 THEN '及格'
    ELSE '不及格'
END AS '成绩'
FROM SC
WHERE sid = '80000759500' AND cid = '020201';

```

图 5: 运行结果

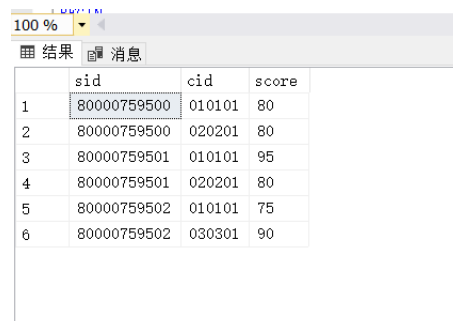


	成绩
1	优秀

6. 使用T-SQL命令CREATE PROC语句可创建存储过程，创建一个带输入和输出参数的存储过程，查询学生选修课程成绩，将分数低于60分的成绩改为60分，高于80分的成绩改为80分。输入参数为学生的学号，输出参数为提示信息，如果不学生不存在则参数值为查无此人，更改失败则为更改失败，更改成功则为更改成功。

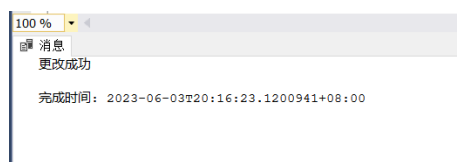
图 6: 运行结果

(b) 修改结果



	sid	cid	score
1	80000759500	010101	80
2	80000759500	020201	80
3	80000759501	010101	95
4	80000759501	020201	80
5	80000759502	010101	75
6	80000759502	030301	90

(a) 输出消息



消息
更改成功

完成时间: 2023-06-03T20:16:23.1200941+08:00

```

CREATE PROCEDURE SP_UPDATE_SCORE
    @sid VARCHAR(20),
    @msg VARCHAR(50) OUTPUT
AS
BEGIN

```

```
SET NOCOUNT ON;

DECLARE @result INT;
DECLARE @transactionName VARCHAR(20);
SET @transactionName = 'UpdateScoreTransaction';

BEGIN TRY
    BEGIN TRANSACTION @transactionName;

    -- 判断学生是否存在
    IF EXISTS(SELECT 1 FROM STUDENTS WHERE sid=@sid)
    BEGIN
        -- 更新成绩
        UPDATE SC SET score=CASE
            WHEN score < 60 THEN 60
            WHEN score > 80 THEN 80
            ELSE score
        END WHERE sid=@sid;

        -- 提交事务
        SET @result = 1;
        SET @msg = '更改成功';
        COMMIT TRANSACTION @transactionName;
    END
    ELSE
    BEGIN
        -- 回滚事务
        SET @result = 0;
        SET @msg = '查无此人';
        ROLLBACK TRANSACTION @transactionName;
    END
END TRY
BEGIN CATCH
    -- 回滚事务
    SET @result = -1;
    SET @msg = '更改失败';
    ROLLBACK TRANSACTION @transactionName;
END CATCH

RETURN @result;
END

DECLARE @sid VARCHAR(20);
DECLARE @msg VARCHAR(50);

SET @sid = '80000759500';

EXEC SP_UPDATE_SCORE @sid=@sid, @msg=@msg OUTPUT;

PRINT @msg;
```

```
select * from sc
```

7. 查询学号为800007595的学生的email转换成大写输出，并查询其选修课程名的前三个字符。提示：使用UPPER（）函数和SUBSTRING（）函数。

```
SELECT UPPER(email) AS 'Email', SUBSTRING(cname, 1, 3) AS 'CourseName'
FROM STUDENTS
JOIN SC ON STUDENTS.sid = SC.sid
JOIN COURSES ON COURSES.cid = SC.cid
WHERE STUDENTS.sid = '80000759500';
```

图 7: 运行结果



	Email	CourseName
1	DDFF@SINA.COM	高等数
2	DDFF@SINA.COM	大学英

8. 用户自定义函数分为：标量值函数、内联表值函数、多语句表值函数。实验要求：创建标量值函数，要求根据输入的学生学号参数，返回学生的选课的平均成绩。

```
CREATE FUNCTION FN_AVG_SCORE(@sid VARCHAR(20))
RETURNS FLOAT
AS
BEGIN
    DECLARE @avgScore FLOAT;
    SELECT @avgScore = AVG(score) FROM SC WHERE sid = @sid;
    RETURN @avgScore;
END;

CREATE FUNCTION FN_SHOW_COURSE_SCORE(@name VARCHAR(50))
RETURNS TABLE
AS
RETURN
(
    SELECT COURSES.cname AS 'CourseName', SC.score AS 'Score'
    FROM SC
    JOIN STUDENTS ON SC.sid = STUDENTS.sid
    JOIN COURSES ON SC.cid = COURSES.cid
    WHERE STUDENTS.name = @name
);

CREATE FUNCTION FN_SHOW_STUDENT_SCORE(@cname VARCHAR(50))
RETURNS @tb_scores TABLE (name VARCHAR(50), score INT)
AS
BEGIN
    DECLARE @cid VARCHAR(20)
    SELECT @cid = cid FROM COURSES WHERE cname = @cname;
```



```

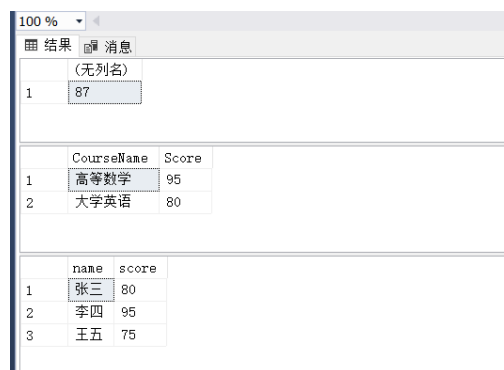
INSERT INTO @tb_scores(name, score)
SELECT STUDENTS.name, SC.score
FROM SC
JOIN STUDENTS ON STUDENTS.sid = SC.sid
WHERE SC.cid = @cid
RETURN
END;

DECLARE @sid VARCHAR(20)
SET @sid = '80000759501'
SELECT dbo.FN_AVG_SCORE(@sid)

SELECT * FROM dbo.FN_SHOW_COURSE_SCORE('李四');
SELECT * FROM dbo.FN_SHOW_STUDENT_SCORE('高等数学');

```

图 8: 运行结果



The screenshot shows a SQL query result window with a zoom level of 100%. It displays three tables of data:

(无列名)	
1	87

	CourseName	Score
1	高等数学	95
2	大学英语	80

	name	score
1	张三	80
2	李四	95
3	王五	75

9. 游标不同于查询语句，查询语句只能参整个结果集进行同一种操作，而游标允许定位在结果集的特定行，从结果集的当前位置检索一行或多行，支持对结果集中当前位置的行进行数据修改，为其他用户对显示在结果集中的数据库数据所做的更改提供不同级别的可见性支持，提供脚本、存储过程和触发器中用于访问结果集中数据的T-SQL语句。

实验要求：定义一个游标，将学号为800007595的学生的选修课程名和成绩逐行打印出来。定义一个游标，将学号为800007595的学生的第二门选修课程成绩（成绩降序排列）改为75分。创建一个没有唯一索引的表，定义一个游标，删除其中一条记录，查看是否允许删除。

```

DECLARE @sid VARCHAR(20)
DECLARE @cname VARCHAR(50)
DECLARE @score INT

DECLARE cur_student CURSOR FOR
SELECT COURSES.cname, SC.score
FROM SC
JOIN COURSES ON SC.cid = COURSES.cid
WHERE SC.sid = '80000759501'

OPEN cur_student

FETCH NEXT FROM cur_student INTO @cname, @score

```

表格 1: 存储过程和存储函数的异同点和相同点

类型	存储过程	存储函数
用途	用于执行一系列的T-SQL语句，可以进行数据处理，如插入、删除和更新数据等	用于计算一个单一的数值或结果集并返回该值或结果集
返回值	存储过程可以返回多个结果集	存储函数只能返回单一的标量值或结果集
输入参数	存储过程支持默认值，可以使用‘OUTPUT’修饰符来暴露参数	存储函数必须拥有至少一个输入参数
输出参数	可以定义输出参数，并将它们用作返回值和在存储过程执行结束时传递到调用程序	不支持输出参数
异常处理	存储过程可以使用‘TRY...CATCH’结构来处理异常	存储函数不能包含异常处理
执行计划	存储过程在第一次执行时会编译和缓存，可以提高代码的执行性能	存储函数也会编译和缓存，但如果函数中使用了可变数据，该函数不会缓存执行计划
调用方式	存储过程可以通过‘EXEC’命令或存储过程名称调用	存储函数可以直接在查询中使用，也可以通过函数名称调用
事务处理	存储过程可以在事务中执行，并可以使用事务控制语句	存储函数不能用于处理事务
可重用性	存储过程可以设计为可重用的代码模块	存储函数也可以设计为可重用的代码模块
安全性	存储过程可以设置安全性，包括权限和角色	存储函数也可以设置安全性，包括权限和角色

```
WHILE @@FETCH_STATUS = 0
BEGIN
    PRINT 'CourseName: ' + @cname + ', Score: ' + CAST(@score AS VARCHAR(10))
    FETCH NEXT FROM cur_student INTO @cname, @score
END

CLOSE cur_student
DEALLOCATE cur_student
```

图 9: 运行结果



```
DECLARE @sid VARCHAR(20)
DECLARE @cid VARCHAR(20)
DECLARE @score INT
```

```

DECLARE cur_score CURSOR FOR
SELECT SC.cid, SC.score
FROM SC
WHERE SC.sid = '80000759501'
ORDER BY SC.score DESC OFFSET 1 ROWS FETCH NEXT 1 ROW ONLY

OPEN cur_score

FETCH NEXT FROM cur_score INTO @cid, @score

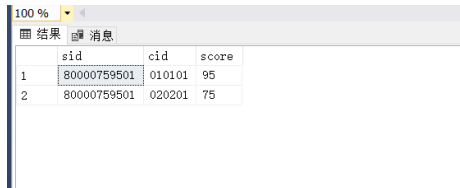
WHILE @@FETCH_STATUS = 0
BEGIN
    UPDATE SC SET score = 75 WHERE cid = @cid AND sid = '800007595'
    FETCH NEXT FROM cur_score INTO @cid, @score
END

CLOSE cur_score
DEALLOCATE cur_score

```

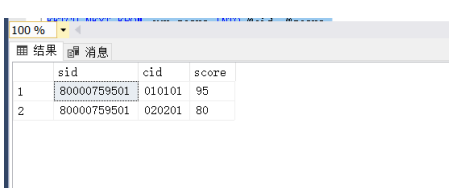
图 10: 运行结果

(a) 修改前



	sid	cid	score
1	80000759501	010101	95
2	80000759501	020201	75

(b) 修改后



	sid	cid	score
1	80000759501	010101	95
2	80000759501	020201	80

```

CREATE TABLE Customers (
    customer_id INT,
    customer_name VARCHAR(50),
    city VARCHAR(50)
)

INSERT INTO Customers VALUES (1, 'John', 'New York')
INSERT INTO Customers VALUES (2, 'Jane', 'Washington')
INSERT INTO Customers VALUES (3, 'David', 'Tokyo')

DECLARE @customer_id INT
DECLARE cur_customer CURSOR FOR
SELECT customer_id FROM Customers

OPEN cur_customer

FETCH NEXT FROM cur_customer INTO @customer_id

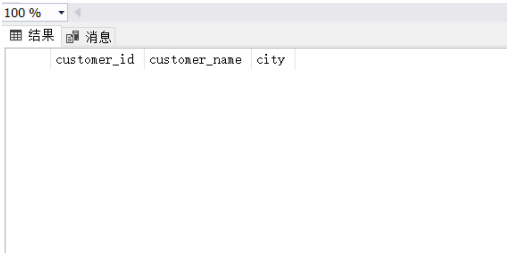
WHILE @@FETCH_STATUS = 0
BEGIN
    DELETE FROM Customers WHERE customer_id = @customer_id
    FETCH NEXT FROM cur_customer INTO @customer_id

```

```
END

CLOSE cur_customer
DEALLOCATE cur_customer
```

图 11: 运行结果



customer_id	customer_name	city
-------------	---------------	------

可见是可以修改的。

4 实验总结

通过本次实验，我学到了数据库管理系统中过程语言的基本概念与语法以及如何使用过程语言进行高效的数据库操作。

在实验过程中，我掌握了变量定义与声明的方法，灵活运用@@ERROR、@@ROWCOUNT、@@IDENTITY等全局变量处理异常。同时，我还学习了多种流程控制语句的使用方法，例如BEGIN...END、IF...ELSE...、WHILE...CONTINUE...BREAK...、CASE等等，运用这些语句可以轻松实现复杂的数据处理功能。

此外，我还了解了存储过程的概念，学习了如何创建存储过程以及如何优化存储过程的性能。同时，我也学习了如何使用系统函数和用户自定义函数，这些函数可以极大地简化代码，提高操作效率。

最后，掌握游标的使用也是本次实验的一个重要内容。我学习了如何在过程语言中使用游标，并灵活应用游标处理数据。

综上所述，此次实验让我对数据库系统的管理和操作有了更深入的了解，掌握了使用过程语言进行高效的数据库操作的方法和技巧，也加深了对于数据库系统的学习和理解。