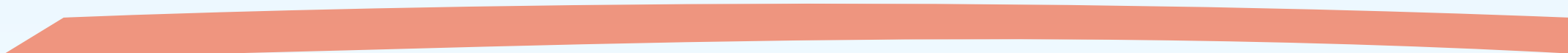


数据库系统

第8章 数据库编程



胡 敏

合肥工业大学

jsjxhumin@hfut.edu.cn

uhnim@163.com

第 8 章 数据库编程

8.1 嵌入式SQL

8.2 过程化SQL

8.3 存储过程和函数

8.4 ODBC编程

8.5 OLE DB

8.6 JDBC编程

8.7 小结



8.2 过程化SQL

8.2.1 PL/SQL与Transact-SQL

8.2.2 变量常量的定义

8.2.3 流程控制



8.2.1 PL/SQL与Transact-SQL

■ PL/SQL :

- Oracle 对SQL的扩展
- 增加了过程化语句功能
- 基本结构是块
 - 块之间可以互相嵌套
 - 每个块完成一个逻辑操作



8.2.1 PL/SQL与Transact-SQL

■ PL/SQL的作用

- ✓ 能够使一组SQL语句的功能更具模块化程序特点;
- ✓ 采用了过程性语言控制程序的结构;
- ✓ 可以对程序中的错误进行自动处理, 使程序能够在遇到错误的时候不会被中断;
- ✓ 具有较好的可移植性, 可以移植到另一个Oracle数据库中;
- ✓ 集成在数据库中, 调用更快;
- ✓ 减少了网络的交互, 有助于提高程序性能



PL/SQL的块结构（续）

■ PL/SQL块的基本结构:

1.定义部分

DECLARE

-----变量、常量、游标、异常等

- 定义的变量、常量等只能在该基本块中使用

- 当基本块执行结束时，定义就不再存在

■ PL/SQL块的基本结构(续):

2.执行部分

BEGIN—必要部分

SQL语句和PL/SQL语句构成的执行程序

.....
EXCEPTION—可选部分

程序出现异常时，捕捉异常并处理异常

.....
END; —必须部分



8.2 过程化SQL

8.2.1 PL/SQL的块结构

8.2.2 变量常量的定义

8.2.3 流程控制



8.2.2 变量常量的定义

1. PL/SQL中定义变量的语法形式是:

变量名 [constant] 变量类型 [not null] [default 值 | :=值]

变量名 数据类型 [[NOT NULL] :=初值表达式] 或

变量名 数据类型 [[NOT NULL] 初值表达式]

2. 常量的定义类似于变量的定义:

常量名 数据类型 **CONSTANT :=常量表达式**

常量必须要给一个值，并且该值在存在期间或常量的作用域内不能改变。如果试图修改它，PL/SQL将返回一个异常。

3. 赋值语句

变量名称:=表达式

v_name varchar2(10);

v_age constant number:=20;

v_sex char(2) default '男';



8.2 过程化SQL

8.2.1 PL/SQL的块结构

8.2.2 变量常量的定义

8.2.3 流程控制



8.2.3 流程控制

■ PL/SQL 功能:

- 一、条件控制语句
- 二、循环控制语句
- 三、错误处理



控制结构（续）

■ 一、 条件控制语句

IF-THEN, IF-THEN-ELSE和嵌套的IF语句

1. IF *condition* THEN

Sequence_of_statements;

END IF

2. IF *condition* THEN

Sequence_of_statements1;

ELSE

Sequence_of_statements2;

END IF;

**IF (new.Job='讲师') AND (new.Sal < 3000) THEN
new.Sal :=3000;
END IF;**

3. 在THEN和ELSE子句中还可以再包括IF语句，即IF语句可以嵌套



二、循环控制语句

LOOP， WHILE-LOOP和FOR-LOOP

1.最简单的循环语句**LOOP**

LOOP

语句1;

语句2;

.....

exit [when 条件];

END LOOP;

多数数据库服务器的PL/SQL都提供EXIT、BREAK或LEAVE等循环结束语句，保证LOOP语句块能够结束。



控制结构（续）

二、循环控制语句（续）

2. WHILE-LOOP

WHILE condition LOOP

Sequence_of_statements;

END LOOP;

- 每次执行循环体语句之前，首先对条件进行求值
- 如果条件为真，则执行循环体内的语句序列。
- 如果条件为假，则跳过循环并把控制传递给下一个语句

3. FOR-LOOP

FOR count IN [REVERSE] *bound1 ... bound2*

Sequence_of_statements;

END LOOP;



■ 三、错误处理：

- 如果PL/SQL在执行时出现异常，则应该让程序在产生异常的语句处停下来，根据异常的类型去执行异常处理语句
- SQL标准对数据库服务器提供什么样的异常处理做出了建议，要求PL/SQL管理器提供完善的异常处理机制

EXCEPTION

WHEN exception_name THEN

Code for handing exception_name;

[WHEN another_exception THEN

Code for handing another_exception];

[WHEN others THEN

code for handing any other exception.];

Oracle预定义异常：no_data_found, too_many_rows; storage_error, zero_divide
等



■ Transact-SQL

T-SQL是Microsoft公司在SQL Server中的SQL-3标准的实现，是对SQL的扩展，具有SQL的主要特点，同时增加了变量、运算符、函数、流程控制和注释等语言元素，使得其功能更加强大。

T-SQL语句分为四大类，分别为：**数据定义语句，数据操作语句，数据控制语句和一些附加的语言元素。**



■ T-SQL的变量

- 局部变量

declare @id char(10)

- 全局变量

必须以@@开头



■ T-SQL的语句

● 赋值语句

```
set @id = '10010001'  
select @id = '10010001'
```

● 条件语句

```
if @x > @y  
    print 'x > y'  
else if @y > @z print 'y > z'  
    else print 'z > y'
```



● 分支语句

Case

when 条件表达式1 then

.....

when 条件表达式n then

.....

else

.....

End



● 循环语句

while <表达式>

begin

 <命令行或程序块>

 [**break**]

 [**continue**]

 <命令行或程序块>

end



■ 异常处理

Transact-SQL 代码中的错误可使用 **TRY...CATCH** 构造处理。**TRY...CATCH** 构造包括两部分：一个 **TRY** 块和一个 **CATCH** 块

```
BEGIN TRY
    SELECT 1/0;
END TRY
BEGIN CATCH
    SELECT
        ERROR_NUMBER() AS ErrorNumber,
        ERROR_PROCEDURE() AS ErrorProcedure,
        ERROR_MESSAGE() AS ErrorMessage;
END CATCH;
```



第 8 章 数据库编程

8.1 嵌入式SQL

8.2 过程化SQL

8.3 存储过程和函数

8.4 ODBC编程

8.5 OLE DB

8.6 JDBC编程

8.7 小结



8.3 存储过程和函数

8.3.1 存储过程

8.3.2 函数

*8.3.3 过程化SQL中的游标



8.3.1 存储过程

■ 过程化SQL块类型

➤ 命名块

- ★ 编译后保存在数据库中，可以被反复调用，运行速度较快，过程和函数是命名块

➤ 匿名块

- ★ 每次执行时都要进行编译，它不能被存储到数据库中，也不能在其他过程化SQL块中调用



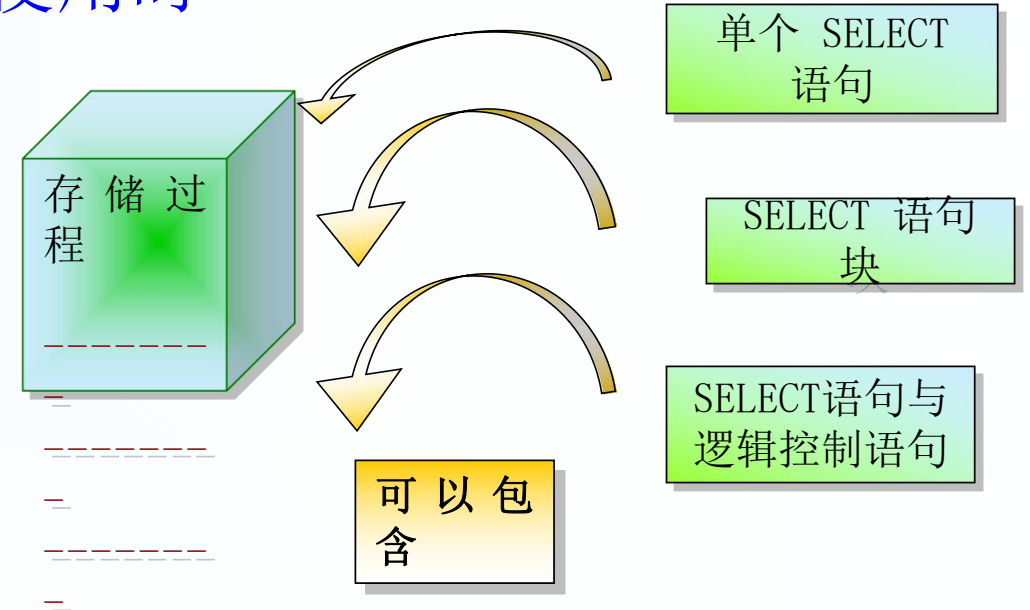
存储过程（续）

1. 存储过程的优点
2. 存储过程的用户接口



存储过程（续）

- 存储过程：由过程化SQL语句书写的过程，经编译和优化后存储在数据库服务器中，使用时只要调用即可。
- 存储过程的优点
 - （1）运行效率高
 - （2）降低了客户机和服务器之间的通信量
 - （3）方便实施企业规则



- 存储过程可以包含数据操纵语句、变量、逻辑 控制语句等



存储过程（续）

■ 存储过程的用户接口

- （1）创建存储过程
- （2）执行存储过程
- （3）修改存储过程
- （4）删除存储过程



2. 存储过程的用户接口

(1) 创建存储过程

CREATE OR REPLACE PROCEDURE 过程名([参数1,参数2,...]) AS <过程化SQL块>;

- 过程名：数据库服务器合法的对象标识
- 参数列表：用名字来标识调用时给出的参数值，必须指定值的数据类型。参数也可以定义输入参数、输出参数或输入/输出参数，默认为输入参数
- 过程体：是一个<过程化SQL块>，包括声明部分和可执行语句部分



存储过程的用户接口（续）

- [例8.8] 利用存储过程来实现下面的应用：从账户1转指定数额的款项到账户2中。

```
CREATE OR REPLACE PROCEDURE TRANSFER(inAccount INT,outAccount  
INT,amount FLOAT)
```

```
/*定义存储过程TRANSFER，其参数为转入账户、转出账户、转账额度*/
```

```
AS DECLARE          /*定义变量*/
```

```
    totalDepositOut Float;
```

```
    totalDepositIn Float;
```

```
    inAccountnum INT;
```



存储过程的用户接口（续）

```
BEGIN                                /*检查转出账户的余额*/
    SELECT Total INTO totalDepositOut FROM Accout
    WHERE accountnum=outAccount;
    IF totalDepositOut IS NULL THEN
                                    /*如果转出账户不存在或账户中没有存款*/
        ROLLBACK;                  /*回滚事务*/
        RETURN;
    END IF;
```



存储过程的用户接口（续）

```
IF totalDeposit Out< amount THEN          /*如果账户存款不足*/
    ROLLBACK;                               /*回滚事务*/
    RETURN;
END IF;
SELECT Accountnum INTO inAccountnum FROM Account
WHERE accountnum=inAccount;
IF inAccount IS NULL THEN                  /*如果转入账户不存在*/
    ROLLBACK;                               /*回滚事务*/
    RETURN;
ENDIF;
```



存储过程的用户接口（续）

```
UPDATE Account SET total=total-amount  
WHERE accountnum=outAccount;
```

/* 修改转出账户余额，减去转出额 */

```
UPDATE Account SET total=total + amount  
WHERE accountnum=inAccount;
```

/* 修改转入账户余额，增加转入额 */

```
COMMIT;  
END;
```

/* 提交转账事务 */



存储过程的用户接口（续）

（2）执行存储过程

CALL/PERFORM PROCEDURE 过程名([参数1,参数2,...]);

- 使用CALL或者PERFORM等方式激活存储过程的执行
- 在过程化SQL中，数据库服务器支持在过程体中调用其他存储过程

- [例8.9] 从账户01003815868转10000元到01003813828账户中。
CALL PROCEDURE TRANSFER(01003813828,01003815868,10000);



存储过程的用户接口（续）

（3）修改存储过程

ALTER PROCEDURE 过程名1 **RENAME TO** 过程名2;

（4）删除存储过程

DROP PROCEDURE 过程名();



8.3 存储过程和函数

8.3.1 存储过程

8.3.2 函数

*8.3.3 过程化SQL中的游标



8.3.2 函数

■ 函数和存储过程的异同

- 同：都是持久性存储模块
- 异：函数必须指定返回的类型



函数（续）

1. 函数的定义语句格式

CREATE OR REPLACE FUNCTION 函数名 ([参数1,参数2,...])
RETURNS <类型> AS <过程化SQL块>;

2. 函数的执行语句格式

CALL/SELECT 函数名 ([参数1,参数2,...]);

3. 修改函数

➤ 重命名

ALTER FUNCTION 过程名1 RENAME TO 过程名2;

➤ 重新编译

ALTER FUNCTION 过程名 COMPILE;



创建PL/SQL函数

```
create or replace function get_grade(s_no sc.sno%type,c_no sc.cno%type)
return number
is
    r_grade number;
begin
    select grade into r_grade from sc where sno=s_no and cno=c_no;
    if r_grade is null then
        r_grade:=0;
    end if;
    return r_grade;
exception
    when no_data_found then
        begin
            dbms_output.put_line('指定的数据不存在! ');
            return -1;
        end;
    when others then
        begin
            dbms_output.put_line(sqlcode || '-----' || sqlerrm);
            return -10;
        end;
end get_grade;
```



❖ PL/SQL的函数调用

```
declare
```

```
    vv number;
```

```
begin
```

```
    vv := shen.get_grade('20090001', '2010');
```

```
    dbms_output.put_line('成绩=' || vv);
```

```
end;
```

❖ T-SQL函数的使用



下课了。。



休息一会儿。。。。

