第**8**章存储过程

**8.1** 存储过程和函数

**8.1** 存储过程和函数

8.1.1 认识存储过程和函数

存储过程和函数也是一种PL/SQL 块， 是存入数据库的

PL/SQL块。但存储过程和函数不同于已经介绍过的PL/SQL程

序，我们通常把PL/SQL程序称为无名块，而存储过程和函数是以

命名的方式存储于数据库中的。和PL/SQL程序相比，存储过程有

很多优点，具体归纳如下：

\* 存储过程和函数以命名的数据库对象形式存储于数据库当

中。存储在数据库中的优点是很明显的，因为代码不保存在本

地，用户可以在任何客户机上登录到数据库，并调用或修改代码。

\* 存储过程和函数可由数据库提供安全保证，要想使用存储

过程和函数，需要有存储过程和函数的所有者的授权，只有被授

权的用户或创建者本身才能执行存储过程或调用函数。

\* 存储过程和函数的信息是写入数据字典的，所以存储过程

可以看作是一个公用模块，用户编写的PL/SQL程序或其他存储过

程都可以调用它(但存储过程和函数不能调用PL/SQL程序)。一个

重复使用的功能，可以设计成为存储过程，比如：显示一张工资

统计表，可以设计成为存储过程；一个经常调用的计算，可以设

计成为存储函数；根据雇员编号返回雇员的姓名，可以设计成存

储函数。

\* 像其他高级语言的过程和函数一样，可以传递参数给存储

过程或函数，参数的传递也有多种方式。存储过程可以有返回

值，也可以没有返回值，存储过程的返回值必须通过参数带回；

函数有一定的数据类型，像其他的标准函数一样，我们可以通过

对函数名的调用返回函数值。

存储过程和函数需要进行编译，以排除语法错误，只有编

译通过才能调用。

8.1.2 创建和删除存储过程

创建存储过程，需要有CREATE PROCEDURE或CREATE

ANY PROCEDURE的系统权限。该权限可由系统管理员授予。创

建一个存储过程的基本语句如下：

CREATE [OR REPLACE] PROCEDURE 存储过程名[(参数

[IN|OUT|IN OUT] 数据类型...)]

{AS|IS}

[说明部分]

BEGIN

可执行部分

[EXCEPTION

错误处理部分]

END [过程名];

其中：

可选关键字OR REPLACE 表示如果存储过程已经存在，则用新的存

储过程覆盖，通常用于存储过程的重建。

参数部分用于定义多个参数(如果没有参数，就可以省略)。参数有三

种形式：IN、OUT和IN OUT。如果没有指明参数的形式，则默认为IN。

关键字AS也可以写成IS，后跟过程的说明部分，可以在此定义过程

的局部变量。

编写存储过程可以使用任何文本编辑器或直接在SQL\*Plus环境下进

行，编写好的存储过程必须要在SQL\*Plus环境下进行编译，生成编译代

码，原代码和编译代码在编译过程中都会被存入数据库。编译成功的存储

过程就可以在Oracle环境下进行调用了。

一个存储过程在不需要时可以删除。删除存储过程的人是过

程的创建者或者拥有DROP ANY PROCEDURE系统权限的人。删

除存储过程的语法如下：

DROP PROCEDURE 存储过程名；

如果要重新编译一个存储过程，则只能是过程的创建者或者

拥有ALTER ANY PROCEDURE系统权限的人。语法如下：

ALTER PROCEDURE 存储过程名COMPILE；

执行( 或调用) 存储过程的人是过程的创建者或是拥有

EXECUTE ANY PROCEDURE系统权限的人或是被拥有者授予

EXECUTE权限的人。执行的方法如下：

方法1：

EXECUTE 模式名.存储过程名[(参数...)];

方法2：

BEGIN

模式名.存储过程名[(参数...)];

END;

传递的参数必须与定义的参数类型、个数和顺序一致(如果

参数定义了默认值，则调用时可以省略参数)。参数可以是变量、

常量或表达式，用法参见下一节。

如果是调用本账户下的存储过程，则模式名可以省略。要调

用其他账户编写的存储过程，则模式名必须要添加。

以下是一个生成和调用简单存储过程的训练。注意要事先授

予创建存储过程的权限。

【训练1】创建一个显示雇员总人数的存储过程。

步骤1：登录SCOTT账户(或学生个人账户)。

步骤2：在SQL\*Plus输入区中，输入以下存储过程：

CREATE OR REPLACE PROCEDURE EMP\_COUNT

AS

V\_TOTAL NUMBER(10);

BEGIN

SELECT COUNT(\*) INTO V\_TOTAL FROM EMP;

DBMS\_OUTPUT.PUT\_LINE('雇员总人数为：'||V\_TOTAL);

END;

步骤3：按“执行”按钮进行编译。

如果存在错误，就会显示:

警告: 创建的过程带有编译错误。

如果存在错误，对脚本进行修改，直到没有错误产生。

如果编译结果正确，将显示：

过程已创建。

步骤4：调用存储过程，在输入区中输入以下语句并执行：

EXECUTE EMP\_COUNT;

显示结果为：

雇员总人数为：14

PL/SQL 过程已成功完成。

说明：在该训练中，V\_TOTAL变量是存储过程定义的局部变量，

用于接收查询到的雇员总人数。

注意：在SQL\*Plus中输入存储过程，按“执行”按钮是进行编译，

不是执行存储过程。

如果在存储过程中引用了其他用户的对象，比如表，则必须有其

他用户授予的对象访问权限。一个存储过程一旦编译成功，就可以由其

他用户或程序来引用。但存储过程或函数的所有者必须授予其他用户执

行该过程的权限。

存储过程没有参数，在调用时，直接写过程名即可

表8-1 参数的类型

参数类型 说 明

IN 定义一个输入参数变量，用于传递参数给存储过程

OUT 定义一个输出参数变量，用于从存储过程获取数据

IN OUT 定义一个输入、输出参数变量，兼有以上两者的功能

参数的定义形式和作用如下：

参数名**IN** 数据类型**DEFAULT** 值；

定义一个输入参数变量，用于传递参数给存储过程。在调用存储过程

时，主程序的实际参数可以是常量、有值变量或表达式等。DEFAULT 关

键字为可选项，用来设定参数的默认值。如果在调用存储过程时不指明参

数，则参数变量取默认值。在存储过程中，输入变量接收主程序传递的

值，但不能对其进行赋值。

参数名**OUT** 数据类型；

定义一个输出参数变量，用于从存储过程获取数据，即变量从存储过

程中返回值给主程序。

在调用存储过程时，主程序的实际参数只能是一个变量，而不能

是常量或表达式。在存储过程中，参数变量只能被赋值而不能将其用

于赋值，在存储过程中必须给输出变量至少赋值一次。

参数名IN OUT 数据类型DEFAULT 值；

定义一个输入、输出参数变量，兼有以上两者的功能。在调用存

储过程时，主程序的实际参数只能是一个变量，而不能是常量或表达

式。DEFAULT 关键字为可选项，用来设定参数的默认值。在存储过

程中，变量接收主程序传递的值，同时可以参加赋值运算，也可以对

其进行赋值。在存储过程中必须给变量至少赋值一次。

如果省略IN、OUT或IN OUT，则默认模式是IN。

【训练1】编写给雇员增加工资的存储过程CHANGE\_SALARY，

通过IN类型的参数传递要增加工资的雇员编号和增加的工资额。

步骤1：登录SCOTT账户。

步骤2：在SQL\*Plus输入区中输入以下存储过程并执行：

CREATE OR REPLACE PROCEDURE

CHANGE\_SALARY(P\_EMPNO IN NUMBER DEFAULT 7788,P\_RAISE

NUMBER DEFAULT 10)

AS

V\_ENAME VARCHAR2(10);

V\_SAL NUMBER(5);

BEGIN

SELECT ENAME,SAL INTO V\_ENAME,V\_SAL FROM EMP

WHERE EMPNO=P\_EMPNO;

UPDATE EMP SET SAL=SAL+P\_RAISE WHERE

EMPNO=P\_EMPNO;

DBMS\_OUTPUT.PUT\_LINE('雇员'||V\_ENAME||'的工资被改

为'||TO\_CHAR(V\_SAL+P\_RAISE));

COMMIT;

EXCEPTION

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('发生错误，修改失败！');

ROLLBACK;

END;

执行结果为：

过程已创建。

步骤3：调用存储过程，在输入区中输入以下语句并执行：

EXECUTE CHANGE\_SALARY(7788,80)

显示结果为：

雇员SCOTT的工资被改为3080

说明：从执行结果可以看到，雇员SCOTT的工资已由原来的

3000改为3080。

参数的值由调用者传递，传递的参数的个数、类型和顺序应该和

定义的一致。如果顺序不一致，可以采用以下调用方法。如上例，执行

语句可以改为：

EXECUTE CHANGE\_SALARY(P\_RAISE=>80,P\_EMPNO=>7788);

可以看出传递参数的顺序发生了变化，并且明确指出了参数名和

要传递的值，=>运算符左侧是参数名，右侧是参数表达式，这种赋值方

法的意义较清楚。

BEGIN

可执行部分

RETURN (表达式)

[EXCEPTION

错误处理部分]

END [函数名];

其中，参数是可选的，但只能是IN类型(IN关键字可以省略)。

在定义部分的RETURN 数据类型，用来表示函数的数据类

型，也就是返回值的类型，此部分不可省略。

在可执行部分的RETURN(表达式)，用来生成函数的返回值，其

表达式的类型应该和定义部分说明的函数返回值的数据类型一致。在

函数的执行部分可以有多个RETURN语句，但只有一个RETURN语句

会被执行，一旦执行了RETURN语句，则函数结束并返回调用环境。

一个存储函数在不需要时可以删除，但删除的人应是函数的创

建者或者是拥有DROP ANY PROCEDURE系统权限的人。其语法如

下：

DROP FUNCTION 函数名

重新编译一个存储函数时，编译的人应是函数的创建者或者拥

有ALTER ANY PROCEDURE系统权限的人。重新编译一个存储函

数的语法如下：

ALTER PROCEDURE 函数名COMPILE；

函数的调用者应是函数的创建者或拥有EXECUTE ANY

PROCEDURE系统权限的人，或是被函数的拥有者授予了函数执行

权限的账户。函数的引用和存储过程不同，函数要出现在程序体

中，可以参加表达式的运算或单独出现在表达式中，其形式如下：

变量名:=函数名(...)

【训练1】创建一个通过雇员编号返回雇员名称的函数

GET\_EMP\_NAME。

步骤1：登录SCOTT账户。

步骤2：在SQL\*Plus输入区中输入以下存储函数并编译：

CREATE OR REPLACE FUNCTION GET\_EMP\_NAME(P\_EMPNO

NUMBER DEFAULT 7788)

RETURN VARCHAR2

AS

V\_ENAME VARCHAR2(10);

BEGIN

SELECT ENAME INTO V\_ENAME FROM EMP WHERE

EMPNO=P\_EMPNO;

RETURN(V\_ENAME);

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('没有该编号雇员！');

RETURN (NULL);

WHEN TOO\_MANY\_ROWS THEN

DBMS\_OUTPUT.PUT\_LINE('有重复雇员编号！');

RETURN (NULL);

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('发生其他错误！');

RETURN (NULL);

END;

步骤3：调用该存储函数，输入并执行以下程序：

BEGIN

DBMS\_OUTPUT.PUT\_LINE(' 雇员7369 的名称是： '||

GET\_EMP\_NAME(7369));

DBMS\_OUTPUT.PUT\_LINE(' 雇员7839 的名称是： '||

GET\_EMP\_NAME(7839));

END;

显示结果为：

雇员7369的名称是：SMITH

雇员7839的名称是：KING

PL/SQL 过程已成功完成。

说明： 函数的调用直接出现在程序的

DBMS\_OUTPUT.PUT\_LINE语句中，作为字符串表达式的一部分。

如果输入了错误的雇员编号，就会在函数的错误处理部分输出错

误信息。试修改雇员编号，重新运行调用部分。