**程序包的创建与应用**

7.1  程序包简介

    程序包(PACKAGE，简称包)是一组相关过程、函数、变量、常量和游标等PL/SQL程序设计元素的组合，作为一个完整的单元存储在数据库中，用名称来标识包。它具有面向对象程序设计语言的特点，是对这些PL/SQL 程序设计元素的封装。包类似于c#和JAVA语言中的类，其中变量相当于类中的成员变量，过程和函数相当于类方法。把相关的模块归类成为包，可使开发人员利用面向对象的方法进行存储过程的开发，从而提高系统性能。

       与高级语言中的类相同，包中的程序元素也分为公用元素和私用元素两种，这两种元素的区别是他们允许访问的程序范围不同，即它们的作用域不同。公用元素不仅可以被包中的函数、过程所调用，也可以被包外的PL/SQL程序访问，而私有元素只能被包内的函数和过程序所访问。

当然，对于不包含在程序包中的过程、函数是独立存在的。一般是先编写独立的过程与函数，待其较为完善或经过充分验证无误后，再按逻辑相关性组织为程序包。

**程序包的优点**

       **简化应用程序设计：**程序包的说明部分和包体部分可以分别创建各编译。主要体现     在以下三个方面：

1)        可以在设计一个应用程序时，只创建各编译程序包的说明部分，然后再编写引用该                     程序包的PL/SQL块。

2)        当完成整个应用程序的整体框架后，再回头来定义包体部分。只要不改变包的说明部分，就可以单独调试、增加或替换包体的内容，这不会影响其他的应用程序。

3)        更新包的说明后必须重新编译引用包的应用程序，但更新包体，则不需重新编译引用包的应用程序，以快速进行进行应用程序的原形开发。

       **模块化：**可将逻辑相关的PL/SQL块或元素等组织在一起，用名称来唯一标识程序 包。把一个大的功能模块划分人适当个数小的功能模块，分别完成各自的功能。这样组织的程序包都易于编写，易于理解更易于管理。

       **信息隐藏：**因为包中的元素可以分为公有元素和私有元素。公有元素可被程序包内的过程、函数等的访问，还可以被包外的PL/SQL访问。但对于私有元素只能被包内的过程、函数等访问。对于用户，只需知道包的说明，不用了解包休的具体细节。

       **效率高：**程序包在应用程序第一次调用程序包中的某个元素时，ORACLE将把整个程序包加载到内存中，当第二次访问程序包中的元素时，ORACLE将直接从内在中读取，而不需要进行磁盘I/O操作而影响速度，同时位于内在中的程序包可被同一会话期间的其它应用程序共享。因此，程序包增加了重用性并改善了多用户、多应用程序环境的效率。

对程序包的优点可总结如下：在PL/SQL程序设计中，使用包不仅可以使程序设计模块化，对外隐藏包内所使用的信息（通过使用私用变量），而写可以提高程序的执行效率。因为，当程序首次调用包内函数或过程时，ORACLE将整个包调入内存，当再次访问包内元素时，ORACLE直接从内存中读取，而不需要进行磁盘I/O操作，从而使程序执行效率得到提高。

    一个包由两个分开的部分组成：

    包说明（PACKAGE）：包说明部分声明包内数据类型、变量、常量、游标、子程序和异常错误处理等元素，这些元素为包的公有元素。

    包主体（PACKAGE BODY）：包主体则是包定义部分的具体实现，它定义了包定义部分所声明的游标和子程序，在包主体中还可以声明包的私有元素。

    包说明和包主体分开编译，并作为两部分分开的对象存放在数据库字典中，可查看数据字典user\_source, all\_source, dba\_source，分别了解包说明与包主体的详细信息。

**7.2  程序包的定义**

程序包的定义分为程序包说明定义和程序包主体定义两部分组成。

程序包说明用于声明包的公用组件，如变量、常量、自定义数据类型、异常、过程、函数、游标等。包说明中定义的公有组件不仅可以在包内使用，还可以由包外其他过程、函数。但需要说明与注意的是，我们为了实现信息的隐藏，建议不要将所有组件都放在包说明处声明，只应把公共组件放在包声明部分。包的名称是唯一的，但对于两个包中的公有组件的名称可以相同，这种用“包名.公有组件名“加以区分。

包体是包的具体实现细节，其实现在包说明中声明的所有公有过程、函数、游标等。当然也可以在包体中声明仅属于自己的私有过程、函数、游标等。创建包体时，有以下几点需要注意：

       包体只能在包说明被创建或编译后才能进行创建或编译。

       在包体中实现的过程、函数、游标的名称必须与包说明中的过程、函数、游标一致，包括名称、参数的名称以及参数的模式（IN、OUT、IN OUT）。并建设按包说明中的次序定义包体中具体的实现。

       在包体中声明的数据类型、变量、常量都是私有的，只能在包体中使用而不能被印刷体外的应用程序访问与使用。

       在包体执行部分，可对包说明，包体中声明的公有或私有变量进行初始化或其它设置。

**创建程序包说明语法格式:**

CREATE [OR REPLACE] PACKAGE package\_name  
  [AUTHID {CURRENT\_USER | DEFINER}]  
  {IS | AS}  
  [公有数据类型定义[公有数据类型定义]…]  
  [公有游标声明[公有游标声明]…]  
  [公有变量、常量声明[公有变量、常量声明]…]  
  [公有函数声明[公有函数声明]…]  
  [公有过程声明[公有过程声明]…]  
END [package\_name];

其中：AUTHID CURRENT\_USER和AUTHID DEFINER选项说明应用程序在调用函数时所使用的权限模式，它们与CREATE FUNCTION语句中invoker\_right\_clause子句的作用相同。

**创建程序包主体语法格式:**

CREATE [OR REPLACE] PACKAGE BODY package\_name  
  {IS | AS}  
  [私有数据类型定义[私有数据类型定义]…]  
  [私有变量、常量声明[私有变量、常量声明]…]  
  [私有异常错误声明[私有异常错误声明]…]  
  [私有函数声明和定义[私有函数声明和定义]…]  
  [私有函过程声明和定义[私有函过程声明和定义]…]  
  [公有游标定义[公有游标定义]…]  
  [公有函数定义[公有函数定义]…]  
  [公有过程定义[公有过程定义]…]  
BEGIN  
  执行部分(初始化部分)  
END package\_name;

其中：在包主体定义公有程序时，它们必须与包定义中所声明子程序的格式完全一致。

**7.3  包的开发步骤**

   与开发存储过程类似，包的开发需要几个步骤：

1．   将每个存储过程调式正确；

2．   用文本编辑软件将各个存储过程和函数集成在一起；

3．   按照包的定义要求将集成的文本的前面加上包定义；

4．   按照包的定义要求将集成的文本的前面加上包主体；

5．   使用SQLPLUS或开发工具进行调式。

**7.4  包定义的说明**

**例1:**创建的包为DEMO\_PKG, 该包中包含一个记录变量DEPTREC、两个函数和一个过程。实现对dept表的增加、删除与查询。

CREATE OR REPLACE PACKAGE  DEMO\_PKG  
IS  
  DEPTREC DEPT%ROWTYPE;  
    
  --Add dept...  
  FUNCTION add\_dept(  
           dept\_no    NUMBER,   
           dept\_name VARCHAR2,   
           location  VARCHAR2)  
  RETURN NUMBER;  
    
  --delete dept...  
  FUNCTION delete\_dept(dept\_no NUMBER)  
  RETURN NUMBER;  
    
  --query dept...  
  PROCEDURE query\_dept(dept\_no IN NUMBER);  
END DEMO\_PKG;

   包主体的创建方法，它实现上面所声明的包定义，并在包主体中声明一个私有变量flag和一个私有函数check\_dept，由于在add\_dept和remove\_dept等函数中需要调用check\_dpet函数，所以，在定义check\_dept 函数之前首先对该函数进行声明，这种声明方法称作前向声明。

CREATE OR REPLACE PACKAGE BODY DEMO\_PKG  
IS   
FUNCTION add\_dept  
(  
   dept\_no NUMBER,   
   dept\_name VARCHAR2,   
   location VARCHAR2  
)  
RETURN NUMBER  
IS   
  empno\_remaining EXCEPTION; --自定义异常  
  PRAGMA EXCEPTION\_INIT(empno\_remaining, -**1**);  
   /\* -1 是违反唯一约束条件的错误代码 \*/  
BEGIN  
  INSERT INTO dept VALUES(dept\_no, dept\_name, location);  
  IF SQL%FOUND THEN  
     RETURN **1**;  
  END IF;  
EXCEPTION  
     WHEN empno\_remaining THEN   
        RETURN **0**;  
     WHEN OTHERS THEN  
        RETURN -**1**;  
END add\_dept;  
  
FUNCTION delete\_dept(dept\_no NUMBER)  
RETURN NUMBER  
IS   
BEGIN  
  DELETE FROM dept WHERE deptno = dept\_no;  
  IF SQL%FOUND THEN  
    RETURN **1**;  
  ELSE  
    RETURN **0**;  
   END IF;  
EXCEPTION  
  WHEN OTHERS THEN  
    RETURN -**1**;  
END delete\_dept;  
  
PROCEDURE query\_dept  
(dept\_no IN NUMBER)  
IS  
BEGIN  
      SELECT \* INTO DeptRec FROM dept WHERE deptno=dept\_no;  
EXCEPTION  
       WHEN NO\_DATA\_FOUND THEN    
          DBMS\_OUTPUT.PUT\_LINE('温馨提示:数据库中没有编码为'||dept\_no||'的部门');  
       WHEN TOO\_MANY\_ROWS THEN  
          DBMS\_OUTPUT.PUT\_LINE('程序运行错误,请使用游标进行操作!');  
       WHEN OTHERS THEN  
           DBMS\_OUTPUT.PUT\_LINE(SQLCODE||'----'||SQLERRM);  
END query\_dept;  
  
BEGIN   
    Null;  
END DEMO\_PKG;

   对包内共有元素的调用格式为：包名.元素名称

调用DEMO\_PKG包内函数对dept表进行插入、查询和删除操作，并通过DEMO\_PKG包中的记录变量DEPTREC显示所查询到的数据库信息：

DECLARE  
    Var NUMBER;  
BEGIN  
    Var := DEMO\_PKG.add\_dept(**90**,'HKLORB', 'HAIKOU');  
    IF var =-**1** THEN  
        DBMS\_OUTPUT.PUT\_LINE(SQLCODE||'----'||SQLERRM);  
    ELSIF var =**0** THEN  
        DBMS\_OUTPUT.PUT\_LINE('温馨提示:该部门记录已经存在！');  
    ELSE  
        DBMS\_OUTPUT.PUT\_LINE('温馨提示:添加记录成功！');  
        DEMO\_PKG.query\_dept(**90**);  
        DBMS\_OUTPUT.PUT\_LINE(DEMO\_PKG.DeptRec.deptno||'---'||  
         DEMO\_PKG.DeptRec.dname||'---'||DEMO\_PKG.DeptRec.loc);  
        var := DEMO\_PKG.delete\_dept(**90**);  
        IF var =-**1** THEN  
            DBMS\_OUTPUT.PUT\_LINE(SQLCODE||'----'||SQLERRM);  
        ELSIF var=**0** THEN  
            DBMS\_OUTPUT.PUT\_LINE('温馨提示:该部门记录不存在！');  
        ELSE  
            DBMS\_OUTPUT.PUT\_LINE('温馨提示:删除记录成功！');  
        END IF;  
    END IF;  
END;

**例2:**创建包EMP\_PKG，读取emp表中的数据

**--创建包说明  
CREATE OR REPLACE PACKAGE EMP\_PKG   
IS  
  TYPE emp\_table\_type IS TABLE OF emp%ROWTYPE   
  INDEX BY BINARY\_INTEGER;  
    
  PROCEDURE read\_emp\_table (p\_emp\_table OUT emp\_table\_type);  
END EMP\_PKG;  
  
--创建包体  
CREATE OR REPLACE PACKAGE BODY EMP\_PKG   
IS  
PROCEDURE read\_emp\_table (p\_emp\_table OUT emp\_table\_type)   
IS  
I BINARY\_INTEGER := 0;  
BEGIN  
   FOR emp\_record IN ( SELECT \* FROM emp ) LOOP  
      P\_emp\_table(i) := emp\_record;  
      I := I + 1;  
    END LOOP;  
  END read\_emp\_table;  
END EMP\_PKG;  
  
--执行  
DECLARE   
  E\_table EMP\_PKG.emp\_table\_type;  
BEGIN  
  EMP\_PKG.read\_emp\_table(e\_table);  
  FOR I IN e\_table.FIRST ..e\_table.LAST LOOP  
    DBMS\_OUTPUT.PUT\_LINE(e\_table(i).empno||'  '||e\_table(i).ename);  
  END LOOP;  
END;**

**例3:**创建包MANAGE\_EMP\_PKG，对员工进行管理（新增员工、新增部门、删除指定员工、删除指定部门、增加指定员工的工资与奖金）：

**--创建序列从100开始,依次增加1  
CREATE SEQUENCE empseq   
START WITH 100   
INCREMENT BY 1   
ORDER NOCYCLE;  
  
--创建序列从100开始,依次增加10  
CREATE SEQUENCE deptseq  
START WITH 100  
INCREMENT BY 10   
ORDER NOCYCLE;  
  
-- \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
  -- 创建包说明  
  -- 包   名：MANAGE\_EMP\_PKG   
  -- 功能描述：对员工进行管理(新增员工,新增部门  
  --            ,删除员工,删除部门,增加工资与奖金等)  
  -- 创建人员：胡勇  
  -- 创建日期：2010-05-19  
  -- Q     Q: 80368704  
  -- E-mail : 80368704@yahoo.com.cn  
  -- WebSite: http://www.cnblogs.com/huyong  
-- \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
CREATE OR REPLACE PACKAGE MANAGE\_EMP\_PKG   
AS  
  --增加一名员工       
  FUNCTION hire\_emp  
    (ename VARCHAR2, job VARCHAR2  
    , mgr NUMBER, sal NUMBER  
    , comm NUMBER, deptno NUMBER)  
  RETURN NUMBER;  
  
  --新增一个部门  
  FUNCTION add\_dept(dname VARCHAR2, loc VARCHAR2)  
  RETURN NUMBER;  
    
  --删除指定员工  
  PROCEDURE remove\_emp(empno NUMBER);  
  --删除指定部门  
  PROCEDURE remove\_dept(deptno NUMBER);  
  --增加指定员工的工资  
  PROCEDURE increase\_sal(empno NUMBER, sal\_incr NUMBER);  
  --增加指定员工的奖金  
  PROCEDURE increase\_comm(empno NUMBER, comm\_incr NUMBER);  
END MANAGE\_EMP\_PKG;--创建包说明结束  
  
-- \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
  -- 创建包体  
  -- 包   名：MANAGE\_EMP\_PKG   
  -- 功能描述：对员工进行管理(新增员工,新增部门  
  --            ,删除员工,删除部门,增加工资与奖金等)  
  -- 创建人员：胡勇  
  -- 创建日期：2010-05-19  
  -- Q     Q: 80368704  
  -- E-mail : 80368704@yahoo.com.cn  
  -- WebSite: http://www.cnblogs.com/huyong  
-- \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
CREATE OR REPLACE PACKAGE BODY MANAGE\_EMP\_PKG   
AS  
    total\_emps  NUMBER; --员工数  
    total\_depts NUMBER; --部门数  
    no\_sal    EXCEPTION;  
    no\_comm   EXCEPTION;  
  --增加一名员工   
  FUNCTION hire\_emp(ename VARCHAR2, job VARCHAR2, mgr NUMBER,  
                       sal NUMBER, comm NUMBER, deptno NUMBER)  
  RETURN NUMBER  --返回新增加的员工编号  
  IS  
    new\_empno NUMBER(4);  
  BEGIN  
SELECT empseq.NEXTVAL INTO new\_empno FROM dual;  
SELECT COUNT(\*) INTO total\_emps FROM emp;--当前记录总数  
  
    INSERT INTO emp   
    VALUES (new\_empno, ename, job, mgr, sysdate, sal, comm, deptno);  
    total\_emps:=total\_emps+1;  
  RETURN(new\_empno);  
  EXCEPTION  
     WHEN OTHERS THEN  
        DBMS\_OUTPUT.PUT\_LINE('温馨提示:发生系统错误!');  
  END hire\_emp;  
    
  --新增一个部门  
  FUNCTION add\_dept(dname VARCHAR2, loc VARCHAR2)  
  RETURN NUMBER   
  IS  
    new\_deptno NUMBER(4); --部门编号  
  BEGIN  
    --得到一个新的自增的员工编号  
    SELECT deptseq.NEXTVAL INTO new\_deptno FROM dual;  
    SELECT COUNT(\*) INTO total\_depts FROM dept;--当前部门总数  
    INSERT INTO dept VALUES (new\_deptno, dname, loc);  
    total\_depts:=total\_depts;  
  RETURN(new\_deptno);  
  EXCEPTION  
     WHEN OTHERS THEN  
        DBMS\_OUTPUT.PUT\_LINE('温馨提示:发生系统错误!');  
  END add\_dept;  
    
  --删除指定员工  
  PROCEDURE remove\_emp(empno NUMBER)   
  IS  
    no\_result EXCEPTION; --自定义异常  
  BEGIN   
    DELETE FROM emp WHERE emp.empno=remove\_emp.empno;  
    IF SQL%NOTFOUND THEN  
        RAISE no\_result;  
    END IF;  
    total\_emps:=total\_emps - 1; --总的员工数减1  
  EXCEPTION  
     WHEN no\_result THEN   
        DBMS\_OUTPUT.PUT\_LINE('温馨提示:你需要的数据不存在!');  
     WHEN OTHERS THEN  
        DBMS\_OUTPUT.PUT\_LINE('温馨提示:发生系统错误!');  
  END remove\_emp;  
    
  --删除指定部门  
  PROCEDURE remove\_dept(deptno NUMBER)   
  IS  
     no\_result EXCEPTION; --自定义异常  
     exception\_deptno\_remaining EXCEPTION; --自定义异常  
     /\*-2292 是违反一致性约束的错误代码\*/  
     PRAGMA EXCEPTION\_INIT(exception\_deptno\_remaining, -2292);  
  BEGIN  
    DELETE FROM dept WHERE dept.deptno=remove\_dept.deptno;  
      
    IF SQL%NOTFOUND THEN  
        RAISE no\_result;  
    END IF;  
    total\_depts:=total\_depts-1; --总的部门数减1  
  EXCEPTION  
     WHEN no\_result THEN   
        DBMS\_OUTPUT.PUT\_LINE('温馨提示:你需要的数据不存在!');  
     WHEN exception\_deptno\_remaining THEN   
        DBMS\_OUTPUT.PUT\_LINE('温馨提示:违反数据完整性约束!');  
     WHEN OTHERS THEN  
        DBMS\_OUTPUT.PUT\_LINE('温馨提示:发生系统错误!');  
  END remove\_dept;  
  
  --给指定员工增加指定数量的工资  
  PROCEDURE increase\_sal(empno NUMBER, sal\_incr NUMBER)  
  IS  
    curr\_sal NUMBER(7, 2); --当前工资  
  BEGIN  
    --得到当前工资  
    SELECT sal INTO curr\_sal FROM emp WHERE emp.empno=increase\_sal.empno;  
      
    IF curr\_sal IS NULL THEN   
       RAISE no\_sal;  
    ELSE  
       UPDATE emp SET sal = sal + increase\_sal.sal\_incr --当前工资加新增的工资   
       WHERE emp.empno = increase\_sal.empno;  
    END IF;  
    EXCEPTION  
       WHEN NO\_DATA\_FOUND THEN   
          DBMS\_OUTPUT.PUT\_LINE('温馨提示:你需要的数据不存在!');  
       WHEN no\_sal THEN   
          DBMS\_OUTPUT.PUT\_LINE('温馨提示:此员工的工资不存在!');  
       WHEN OTHERS THEN   
          DBMS\_OUTPUT.PUT\_LINE('温馨提示:发生系统错误!');  
  END increase\_sal;  
    
  --给指定员工增加指定数量的奖金  
  PROCEDURE increase\_comm(empno NUMBER, comm\_incr NUMBER)   
  IS  
    curr\_comm NUMBER(7,2);  
  BEGIN   
    --得到指定员工的当前资金  
    SELECT comm INTO curr\_comm   
    FROM emp WHERE emp.empno = increase\_comm.empno;  
      
    IF curr\_comm IS NULL THEN   
       RAISE no\_comm;  
    ELSE  
      UPDATE emp SET comm = comm + increase\_comm.comm\_incr  
      WHERE emp.empno=increase\_comm.empno;  
    END IF;  
  EXCEPTION  
     WHEN NO\_DATA\_FOUND THEN   
        DBMS\_OUTPUT.PUT\_LINE('温馨提示:你需要的数据不存在!');  
     WHEN no\_comm THEN   
        DBMS\_OUTPUT.PUT\_LINE('温馨提示:此员工的奖金不存在!');  
     WHEN OTHERS THEN   
        DBMS\_OUTPUT.PUT\_LINE('温馨提示:发生系统错误!');  
  END increase\_comm;  
END MANAGE\_EMP\_PKG;--创建包体结束  
  
--调用  
SQL> variable empno number  
SQL>execute  :empno:= manage\_emp\_pkg.hire\_emp('HUYONG',PM,1455,5500,14,10)  
  
PL/SQL procedure successfully completed  
empno  
---------  
105**

**例4：**利用游标变量创建包 CURROR\_VARIBAL\_PKG。由于游标变量指是一个指针，其状态是不确定的，因此它不能随同包存储在数据库中，既不能在PL/SQL包中声明游标变量。但在包中可以创建游标变量参照类型，并可向包中的子程序传递游标变量参数。

**-- \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
  -- 创建包体  
  -- 包   名：CURROR\_VARIBAL\_PKG   
  -- 功能描述：在包中引用游标变量  
  -- 创建人员：胡勇  
  -- 创建日期：2010-05-19  
  -- Q     Q: 80368704  
  -- E-mail : 80368704@yahoo.com.cn  
  -- WebSite: http://www.cnblogs.com/huyong  
-- \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
CREATE OR REPLACE PACKAGE CURROR\_VARIBAL\_PKG AS  
  TYPE DeptCurType IS REF CURSOR   
  RETURN dept%ROWTYPE; --强类型定义  
    
  TYPE CurType IS REF CURSOR;-- 弱类型定义  
    
  PROCEDURE OpenDeptVar(  
    Cv IN OUT DeptCurType,  
    Choice INTEGER DEFAULT 0,  
    Dept\_no NUMBER DEFAULT 50,  
    Dept\_name VARCHAR DEFAULT '%');  
END;  
  
-- \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
  -- 创建包体  
  -- 包   名：CURROR\_VARIBAL\_PKG   
  -- 功能描述：在包中引用游标变量  
  -- 创建人员：胡勇  
  -- 创建日期：2010-05-19  
  -- Q     Q: 80368704  
  -- E-mail : 80368704@yahoo.com.cn  
  -- WebSite: http://www.cnblogs.com/huyong  
-- \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
CREATE OR REPLACE PACKAGE BODY CURROR\_VARIBAL\_PKG  
AS  
  PROCEDURE OpenDeptvar(  
    Cv IN OUT DeptCurType,  
    Choice INTEGER DEFAULT 0,  
    Dept\_no NUMBER DEFAULT 50,  
    Dept\_name VARCHAR DEFAULT ‘%’)  
  IS   
  BEGIN  
    IF choice =1 THEN  
      OPEN cv FOR SELECT \* FROM dept WHERE deptno <= dept\_no;  
    ELSIF choice = 2 THEN  
      OPEN cv FOR SELECT \* FROM dept WHERE dname LIKE dept\_name;  
    ELSE  
      OPEN cv FOR SELECT \* FROM dept;  
    END IF;  
  END OpenDeptvar;  
END CURROR\_VARIBAL\_PKG;  
  
--定义一个过程  
CREATE OR REPLACE PROCEDURE UP\_OpenCurType(  
  Cv IN OUT CURROR\_VARIBAL\_PKG.CurType,  
  FirstCapInTableName CHAR)   
AS  
BEGIN  
  --CURROR\_VARIBAL\_PKG.CurType采用弱类型定义  
  --所以可以使用它定义的游标变量打开不同类型的查询语句  
  IF FirstCapInTableName = 'D' THEN  
    OPEN cv FOR SELECT \* FROM dept;  
  ELSE  
    OPEN cv FOR SELECT \* FROM emp;  
  END IF;  
END UP\_OpenCurType;  
  
  
--定义一个应用  
DECLARE   
  DeptRec Dept%ROWTYPE;  
  EmpRec Emp%ROWTYPE;  
  Cv1 CURROR\_VARIBAL\_PKG.deptcurtype;  
  Cv2 CURROR\_VARIBAL\_PKG.curtype;  
BEGIN  
  DBMS\_OUTPUT.PUT\_LINE('游标变量强类型定义应用');  
  CURROR\_VARIBAL\_PKG.OpenDeptVar(cv1, 1, 30);  
  FETCH cv1 INTO DeptRec;  
  WHILE cv1%FOUND LOOP  
    DBMS\_OUTPUT.PUT\_LINE(DeptRec.deptno||':'||DeptRec.dname);  
    FETCH cv1 INTO DeptRec;  
  END LOOP;  
  CLOSE cv1;  
  
  DBMS\_OUTPUT.PUT\_LINE('游标变量弱类型定义应用');  
  CURROR\_VARIBAL\_PKG.OpenDeptvar(cv2, 2, dept\_name => 'A%');  
  FETCH cv2 INTO DeptRec;  
  WHILE cv2%FOUND LOOP  
    DBMS\_OUTPUT.PUT\_LINE(DeptRec.deptno||':'||DeptRec.dname);  
    FETCH cv2 INTO DeptRec;  
  END LOOP;  
  
  DBMS\_OUTPUT.PUT\_LINE('游标变量弱类型定义应用—dept表');  
  UP\_OpenCurType(cv2, 'D');  
  FETCH cv2 INTO DeptRec;  
  WHILE cv2%FOUND LOOP  
    DBMS\_OUTPUT.PUT\_LINE(deptrec.deptno||':'||deptrec.dname);  
    FETCH cv2 INTO deptrec;  
  END LOOP;  
  
  DBMS\_OUTPUT.PUT\_LINE('游标变量弱类型定义应用—emp表');  
  UP\_OpenCurType(cv2, 'E');  
  FETCH cv2 INTO EmpRec;  
  WHILE cv2%FOUND LOOP  
    DBMS\_OUTPUT.PUT\_LINE(emprec.empno||':'||emprec.ename);  
    FETCH cv2 INTO emprec;  
  END LOOP;  
  CLOSE cv2;  
END;  
----------运行结果-------------------  
游标变量强类型定义应用  
10:ACCOUNTING  
20:RESEARCH  
30:SALES  
游标变量弱类型定义应用  
10:ACCOUNTING  
游标变量弱类型定义应用—dept表  
10:ACCOUNTING  
20:RESEARCH  
30:SALES  
40:OPERATIONS  
50:50abc  
60:Developer  
游标变量弱类型定义应用—emp表  
7369:SMITH  
7499:ALLEN  
7521:WARD  
7566:JONES  
7654:MARTIN  
7698:BLAKE  
7782:CLARK  
7788:SCOTT  
7839:KING  
7844:TURNER  
7876:ADAMS  
7900:JAMES  
7902:FORD  
7934:MILLER  
   
PL/SQL procedure successfully completed**

**7.5  子程序重载**

PL/SQL 允许对包内子程序和本地子程序进行重载。所谓重载时指两个或多个子程序有相同的名称，但拥有不同的参数变量、参数顺序或参数数据类型。

**例5：**

**-- \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
  -- 创建包说明  
  -- 包   名：DEMO\_PKG1   
  -- 功能描述：创建包对子程序重载进行测试  
  -- 创建人员：胡勇  
  -- 创建日期：2010-05-22  
  -- Q     Q: 80368704  
  -- E-mail : 80368704@yahoo.com.cn  
  -- WebSite: http://www.cnblogs.com/huyong  
-- \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
CREATE OR REPLACE PACKAGE DEMO\_PKG1  
IS  
    DeptRec dept%ROWTYPE;  
    V\_sqlcode NUMBER;  
    V\_sqlerr VARCHAR2(2048);  
    
  --两个子程序名字相同,但参数类型不同  
    FUNCTION query\_dept(dept\_no IN NUMBER)  
    RETURN INTEGER;  
    
    FUNCTION query\_dept(dept\_no IN VARCHAR2)  
    RETURN INTEGER;  
END DEMO\_PKG1;  
  
-- \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
  -- 创建包体  
  -- 包   名：DEMO\_PKG1   
  -- 功能描述：创建包对子程序重载进行测试  
  -- 创建人员：胡勇  
  -- 创建日期：2010-05-22  
  -- Q     Q: 80368704  
  -- E-mail : 80368704@yahoo.com.cn  
  -- WebSite: http://www.cnblogs.com/huyong  
-- \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
CREATE OR REPLACE PACKAGE BODY DEMO\_PKG1  
IS   
  FUNCTION check\_dept(dept\_no NUMBER)  
  RETURN INTEGER  
  IS  
    deptCnt INTEGER; --指定部门号的部门数量  
  BEGIN  
    SELECT COUNT(\*) INTO deptCnt FROM dept WHERE deptno = dept\_no;  
    IF deptCnt > 0 THEN  
      RETURN 1;  
    ELSE  
      RETURN 0;  
    END IF;  
  END check\_dept;  
  
  FUNCTION check\_dept(dept\_no VARCHAR2)  
  RETURN INTEGER  
  IS  
    deptCnt INTEGER;  
  BEGIN  
    SELECT COUNT(\*) INTO deptCnt FROM dept WHERE deptno=dept\_no;  
    IF deptCnt > 0 THEN  
      RETURN 1;  
    ELSE  
      RETURN 0;  
    END IF;  
  END check\_dept;  
  
  FUNCTION query\_dept(dept\_no IN NUMBER)  
  RETURN INTEGER  
  IS  
  BEGIN  
    IF check\_dept(dept\_no) =1 THEN  
      SELECT \* INTO DeptRec FROM dept WHERE deptno=dept\_no;  
      RETURN 1;  
    ELSE  
      RETURN 0;  
    END IF;  
  END query\_dept;  
  
  FUNCTION query\_dept(dept\_no IN VARCHAR2)  
    RETURN INTEGER  
  IS  
  BEGIN  
    IF check\_dept(dept\_no) =1 THEN  
      SELECT \* INTO DeptRec FROM dept WHERE deptno = dept\_no;  
      RETURN 1;  
    ELSE  
      RETURN 0;  
    END IF;  
  END query\_dept;  
  
END DEMO\_PKG1;**

**7.6  加密实用程序**

ORACLE 提供了一个实用工具来加密或者包装用户的PL/SQL，它会将用户的PL/SQL改变为只有ORACLE能够解释的代码版本．

WRAP 实用工具位于$ORACLE\_HOME/BIN.

格式为：

WRAP INAME=<input\_file\_name> [ONAME=<output\_file\_name>]

wrap iname=e:\sample.txt

**注意：**在加密前，请将PL/SQL程序先保存一份，以备后用。

**7.7  删除包**

可以使用 DROP PACKAGE 命令对不需要的包进行删除，语法如下：

**DROP PACKAGE [BODY] [user.]package\_name;  
  
DROP PROCEDURE OpenCurType; --删除存储过程  
--删除我们实例中创建的各个包  
DROP PACKAGE demo\_pack;  
DROP PACKAGE demo\_pack1;  
DROP PACKAGE emp\_mgmt;  
DROP PACKAGE emp\_package;**

**7.8  包的管理**

包与过程、函数一样，也是存储在数据库中的，可以随时查看其源码。若有需要，在创建包时可以随时查看更详细的编译错误。不需要的包也可以删除。

同样，为了避免调用的失败，在更新表的结构后，一定要记得重新编译依赖于它的程序包。在更新了包说明或包体后，也应该重新编译包说明与包体。语法如下：

ALTER PACKAGE package\_name COMPILE [PACKAGE|BODY|SPECIFICATION];

也可以通过以下数据字典视图查看包的相关。

DBA\_SOURCE, USER\_SOURCE, USER\_ERRORS, DBA-OBJECTS

如，我们可以用：select text from user\_source where name = 'DEMO\_PKG1';来查看我们创建的包的源码。