PAQUETES EN PL/SQL
SQL DINAMICO

Presentado por: Daniel Alejandro Guerrero Suárez

PAQUETES EN PL/SQL

DEFINICION

Un paquete es una estructura que agrupa objetos de PL/SQL compilados (procedimientos, funciones, variables, tipos ...) en la base de datos. Esto nos permite agrupar la funcionalidad de los procesos en programas.

Lo primero que se debe tener en cuenta que los paquetes están formados por dos partes: la **especificación** y el **cuerpo**. La especificación del un paquete y su cuerpo se crean por separado.

PAQUETES EN PL/SQL

ESPECIFICACION

La **especificación** es la interfaz con las aplicaciones. En ella es posible declarar los tipos, variables, constantes, excepciones, cursores y subprogramas disponibles para su uso posterior desde fuera del paquete. En la especificación del paquete sólo se declaran los objetos (procedures, funciones, variables ...), no se implementa el código.

Los objetos declarados en la especificación del paquete son accesibles desde fuera del paquete por otro script de PL/SQL o programa.

Para crear la cabecera del paquete utilizaremos la siguiente instrucción:

```
CREATE {OR REPLACE} PACKAGE nombre_de_paquete IS
-- Declaraciones
END;
```

PAQUETES EN PL/SQL

CUERPO (IMPLEMENTACION)

El **cuerpo** es la implementación del paquete. El cuerpo del paquete debe implementar lo que se declaró inicialmente en la especificación. Es el donde debemos escribir el código de los subprogramas. En el cuerpo de un package podemos declarar nuevos subprogramas y tipos, pero estos seran privados para el propio package.

La sintaxis general para crear el cuerpo de un paquete es muy parecida a la de la especificación, tan solo se añade la palabra clave **BODY**, y se implementa el código de los subprogramas.

```
CREATE {OR REPLACE} PACKAGE BODY nombre_paquete IS
--Bloques de código
END;
```

PAQUETES EN PL/SQL

LLAMADAS INTERNAS Y EXTERNAS

Para realizar llamadas a objetos dentro de un paquete, habría que diferenciar si la llamada es desde un subprograma dentro del mismo paquete o si la llamada es externa al paquete:

Llamada interna: Se pone el nombre del subprograma y entre paréntesis los parámetros que se deben pasar si es necesario.

Por ej. para llamar a la función "Multiplica(real r1, real r2)" desde la función "CalculaBeneficios", se pondría "Multiplica(2.5,3.5)“

Llamada externa: Se debe preceder el nombre del paquete al nombre del procedimiento o función. Además, el subprograma llamado debe ser público para que pueda ser referenciado.

Por ejemplo para llamar a la función Multiplica que se encuentra en el paquete "OperacionesMatematicas" desde un procedimiento externo al paquete sería "OperacionesMatematicas.Multiplica(2.5,3.5)"

PAQUETES EN PL/SQL

EJEMPLO 1 - ESPECIFICACION

```
CREATE OR REPLACE PACKAGE OPERACIONES_BASICAS IS

  PROCEDURE SUMAR(A NUMBER, B NUMBER);
  PROCEDURE RESTAR(A NUMBER, B NUMBER);
  FUNCTION MULTIPLICAR(A NUMBER, B NUMBER)RETURN NUMBER;

 END OPERACIONES_BASICAS;
/
```

PAQUETES EN PL/SQL

EJEMPLO 1 - IMPLEMENTACION

```
CREATE OR REPLACE PACKAGE BODY OPERACIONES_BASICAS AS

PROCEDURE SUMAR(A NUMBER,B NUMBER) IS
        RESULTADO NUMBER(10);
 BEGIN
        RESULTADO := A + B;
        DBMS_OUTPUT.PUT_LINE('EL RESULTADO DE LA SUMA ES : '||RESULTADO);
 END SUMAR;

 PROCEDURE RESTAR(A NUMBER,B NUMBER) IS
        RESULTADO NUMBER(10);
 BEGIN
         RESULTADO := A - B;
         DBMS_OUTPUT.PUT_LINE('EL RESULTADO DE LA RESTA ES : '||RESULTADO);
 END RESTAR;

FUNCTION MULTIPLICAR(A NUMBER, B NUMBER) RETURN NUMBER
 IS
        RESULTADO NUMBER(30);
  BEGIN
         RESULTADO := A * B;
        return(RESULTADO);
        END MULTIPLICAR;

END OPERACIONES_BASICAS;
/
```

PAQUETES EN PL/SQL

EJEMPLO 1 – LLAMADO EXTERNO

```
EXECUTE OPERACIONES_BASICAS.SUMAR(25,10);


EXECUTE OPERACIONES_BASICAS.RESTAR(25,10);


DECLARE
  Valor NUMBER(30);
BEGIN
        Valor := OPERACIONES_BASICAS.MULTIPLICAR(2,10);
        DBMS_OUTPUT.PUT_LINE('EL RESULTADO DE LA MULTIPLICACION ES : ' | | Valor);
END;
/
```

PAQUETES EN PL/SQL

EJEMPLO 2 - ESPECIFICACION

```
CREATE OR REPLACE PACKAGE OPERACIONES_BASICAS IS

  PROCEDURE SUMAR(A NUMBER, B NUMBER);
  PROCEDURE RESTAR(A NUMBER, B NUMBER);
  FUNCTION MULTIPLICAR(A NUMBER, B NUMBER)RETURN NUMBER;

 END OPERACIONES_BASICAS;
/
```

PAQUETES EN PL/SQL

EJEMPLO 2 - IMPLEMENTACION

```
CREATE OR REPLACE PACKAGE BODY OPERACIONES_BASICAS AS

PROCEDURE SUMAR(A NUMBER,B NUMBER) IS
        RESULTADO NUMBER(10);
 BEGIN
         RESULTADO := A + B;
        DBMS_OUTPUT.PUT_LINE('EL RESULTADO DE LA SUMA ES : ' | | RESULTADO);
        RESTAR(25,10);
 END SUMAR;

 PROCEDURE RESTAR(A NUMBER,B NUMBER) IS
        RESULTADO NUMBER(10);
        Valor NUMBER(30);
 BEGIN
        RESULTADO := A - B;
         DBMS_OUTPUT.PUT_LINE('EL RESULTADO DE LA RESTA ES : ' | | RESULTADO);
         Valor := MULTIPLICAR(2,10);
        DBMS_OUTPUT.PUT_LINE('EL RESULTADO DE LA MULTIPLICACION ES : ' | | Valor);
   END RESTAR;

FUNCTION MULTIPLICAR(A NUMBER, B NUMBER) RETURN NUMBER
  IS
        RESULTADO NUMBER(30);
  BEGIN
         RESULTADO := A * B;
        return(RESULTADO);
        END MULTIPLICAR;

END OPERACIONES_BASICAS;
/
```

PAQUETES EN PL/SQL

EJEMPLO 2 – LLAMADO INTERNO

EXECUTE OPERACIONES_BASICAS.SUMAR(25,10);

Al ejecutar el primer porcedimiento, se llaman automaticamente el procedimiento de resta y la function de Multiplicar

PAQUETES EN PL/SQL

OTRAS SENTENCIAS SQL

LISTA DE PAQUETES DBMS (otras utilidades del Data Base Manager System)

```
SELECT * FROM ALL_OBJECTS
 WHERE OBJECT_NAME LIKE '%DBMS_%'
    AND OBJECT_TYPE = 'PACKAGE'
```

Lista de paquetes guardados en el diccionario de Datos de Oracle:

```
SELECT OBJECT_NAME FROM ALL_OBJECTS
    WHERE OBJECT_TYPE = 'PACKAGE';
```

Para eliminar un paquete de la base de datos tanto la especificación como el cuerpo:

```
DROP PACKAGE nombre_paquete
```

Ver resultados de PL/SQL:

```
SET SERVEROUTPUT ON SIZE UNLIMITED;
```

PAQUETES EN PL/SQL

CONCLUSIONES

Es posible modificar el cuerpo de un paquete sin necesidad de alterar la especificación del mismo.

Permite modularizar el diseño de la aplicación: El uso de Paquetes permite encapsular elementos relacionados entre sí (tipos, variables, procedimientos, funciones) en un único módulo PL/Sql que llevará un nombre que identifique la funcionalidad del conjunto.

Permite ocultar los detalles de implementación: Pueden especificarse cuáles tipos, variables y sub-programas dentro del paquete son públicos (visibles y accesibles por otras aplicaciones y sub-programas fuera del paquete) o privados (ocultos e inaccesibles fuera del paquete). Por ejemplo, dentro del paquete pueden existir procedimientos y funciones que serán invocados por uso interno del paquete que no tendrán posibilidad otros programas, así como también otras rutinas de ser accedidas fuera del mismo. Esto asegura que cualquier cambio en la definición de estas rutinas internas afectará sólo al paquete donde se encuentran, simplificando el mantenimiento y protegiendo la integridad del conjunto.

PAQUETES EN PL/SQL

CONCLUSIONES

Agrega mayor funcionalidad al desarrollo: Las definiciones públicas de tipos, variables y cursores hechas en la especificación de un paquete persisten a lo largo de una sesión. Por lo tanto pueden ser compartidas por todos los sub-programas y/o paquetes que se ejecutan en ese entorno durante esa sesión. Por ejemplo, puede utilizarse esta técnica (en dónde sólo se define una especificación de paquete y no un cuerpo) para mantener tipos y variables globales a todo el sistema.

Introduce mejoras al rendimiento: En relación a su ejecución, cuando un procedimiento o función que está definido dentro de un paquete es llamado por primera vez, todo el paquete es ingresado a memoria. Por lo tanto, posteriores llamadas al mismo u otros sub-programas dentro de ese paquete realizarán un acceso a memoria en lugar de a disco. Esto no sucede con procedimientos y funciones estándares.

Los paquetes pueden llegar a ser programas muy complejos y suelen almacenar gran parte de la lógica de negocio.

SQL DINAMICO

DEFINICION

SQL Dinámico permite ejecutar sentencias SQL a partir de cadenas de caracteres. Para ello debemos emplear la instrucción EXECUTE IMMEDIATE.

```
DECLARE
      str_sql VARCHAR2(255);
      resultado   VARCHAR2(20);
BEGIN
    str_sql := 'SELECT count(*) FROM mecanicos';
    EXECUTE IMMEDIATE str_sql INTO resultado;
    dbms_output.put_line(resultado);
END;
/
```

SQL DINAMICO

VENTAJAS Y DESVENTAJAS

VENTAJAS:

- Creación de sentencias en tiempo de ejecución
- Interactivo con el usuario
- Acceder a objetos no existentes en tiempo de compilación
- Gestión de permisos de usuarios de forma dinámica
- Permite ejecutar instrucciones DDL (create, alter, drop, grant, …)

DESVENTAJAS:

- No siempre se forman las consultas más optimas
- Problemas de seguridad por ataques de inyección SQL