Programación en BBDD

Cuando se utiliza un SGBD es necesario en muchas ocasiones programar tareas que nos puedan servir tanto de soporte a la administración como a la gestión de los datos almacenados.
Dependiendo del SGBD podemos encontrar los siguientes tipos de scripts:

Anónimos: Conjunto de instrucciones (en forma de script) que generalmente no suelen ser reutilizables.
Procedimientos almacenados: Conjunto de instrucciones que se guardan en el servidor para utilizarlo cuando sea necesario. Se suelen usar para realizar un conjunto de acciones.
Funciones: Conjunto de instrucciones que se almacenan en el servidor para realizar alguna tarea. Se suelen usar para hacer cálculos.
Triggers: Programa creado para ejecutarse automáticamente cuando ocurra un evento en nuestra base de datos. Veremos más adelante sus usos más habituales.

En Oracle la programación en BBDD se hace con el lenguaje PL (Procedural Language)/SQL. La unidad lógica en PL es el bloque. Todos los programas están compuestos por **bloques** que pueden ser anidados o no.

SQL es no-procedural, lo cual significa que es Oracle quien se preocupará de cómo ejecutar de la mejor manera un requerimiento señalado en una sentencia SQL. No es necesaria la conexión entre varias sentencias porque Oracle las ejecuta una cada vez.

PL/SQL permite una completa manipulación de los datos almacenados en una base Oracle, proporciona comandos de control de transacciones y permite utilizar las funciones de SQL, operadores y pseudocolumnas. Además, PL/SQL soporta tipos de datos de SQL, lo que reduce la necesidad de convertir los datos al pasar de una a otra aplicación.

FUNDAMENTOS PL/SQL

PL/SQL no es un lenguaje creado para interactuar con el usuario, sino para trabajar con la BD. Prueba de ello es que no dispone de órdenes para introducir datos por teclado, ni para mostrarlos por pantalla.

Oracle incorpora el paquete DBMS_OUTPUT, que incluye el procedimiento PUT_LINE, que permite visualizar textos por pantalla: DBMS_OUTPUT.PUT_LINE ('Expresión').

La estructura de un **bloque** de PL es la siguiente:

[DECLARE] – Opcional

- Variables, cursores y excepciones definidas por el usuario.

BEGIN – Obligatorio

- Sentencias SQL.
- Sentencias de control PL/SQL.

[EXCEPTION] – Opcional

- Acciones a realizar cuando se producen errores.

END; - Obligatorio

/

1º Sección Declarativa

Contiene todas las variables, constantes, y excepciones definidas por el usuario a las que hace referencia en las secciones ejecutables y excepciones.

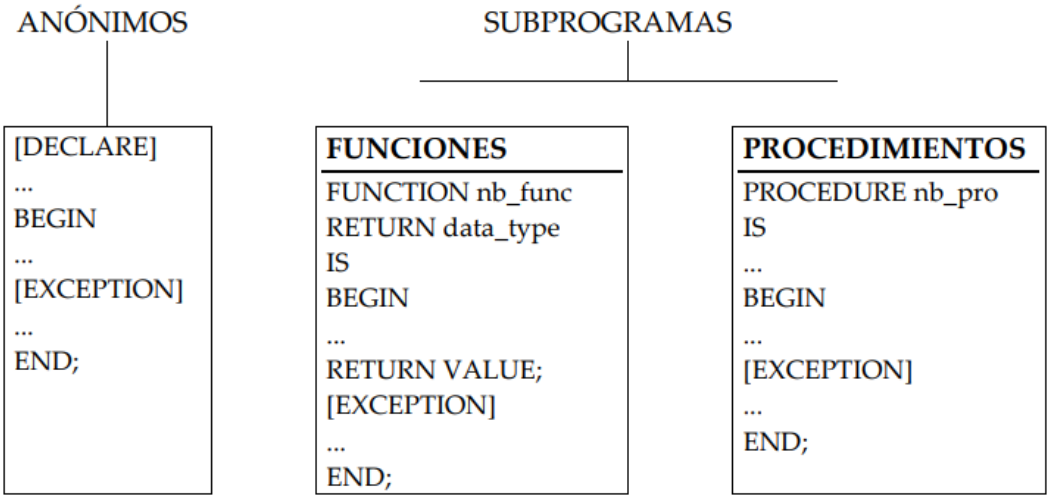
2º Sección Ejecutable

Contiene sentencias SQL para manipular datos de la BD y sentencias PL/SQL para manipular datos del bloque.

3º Sección para Gestión de Excepciones

Especifica las acciones que se han de ejecutar cuando aparecen errores y condiciones anormales de la sección ejecutable.

Tipos de Bloques PL/SQL.



- **Bloques anónimos**: son bloques sin nombrar. Son declarados en una aplicación en la que se van a ejecutar y se transfieren al motor PL/SQL para que se ejecuten al momento.
- **Subprogramas**: son bloques PL/SQL especificados que pueden tomar parámetros y pueden ser invocados. Puede declararlos como procedimientos o como funciones. Generalmente se usan los procedimientos para ejecutar una acción y una función para calcular un valor.

Delimitadores e Identificadores

Un delimitador es un símbolo simple o compuesto que tiene un significado especial dentro de PL/SQL. Algunos de los más utilizados son los siguientes::

Símbolo	Significado
+	operador de suma
'	delimitador de caracteres
/	operador de división
(	expresión o delimitador de lista
)	expresión o delimitador de lista
,	separador de ítems
*	operador de multiplicación
"	delimitador de un identificador entre comillas
=	operador relacional
<	operador relacional

>	operador relacional
;	**terminador de sentencias**
-	negación u operador de substracción
:=	**operador de asignación**
||	operador de concatenación
/*	comienzo de un comentario de varias líneas
***/**	fin de un comentario de varias líneas
<>	operador relacional
!=	operador relacional
<=	operador relacional
>=	operador relacional
--	comentario en una línea

Sintaxis

- Una sentencia puede terminar en una línea y continuar en la otra si ponemos al final de toda la sentencia un punto y coma;

- Los identificadores, literales, comentarios, etc. pueden separarse por uno o más espacios.
- Los identificadores sirven para dar nombre a las unidades y artículos de los programas PL/SQL. Pueden contener hasta 30 caracteres, no pueden contener palabras reservadas, deben comenzar por una letra y el nombre de un identificador no es aconsejable que coincida con el nombre de una columna de una tabla.

- Sintaxis y directrices de los literales: Los números pueden ser simples o ir dentro de una notación científica. Han de ir entre comillas las fechas y las cadenas.

Operadores

Operadores por orden de operación.	**Operación**
** , NOT	Exponenciación, negación lógica.
+ , -	Identidad y negación.
* , /	Multiplicación y división.
+ , - , ||	Suma, resta y concatenación.
= , != , < , >, <= , >= , IS NULL, LIKE, BETWEEN, IN	Comparación.
AND	Conjunción.
OR	Inclusión.

Funciones

Funciones	Descripción
ABS(n)	Devuelve el valor absoluto de un número.
CEIL(n)	Devuelve el entero más pequeño o igual al número.
COS(n)	Coseno de n.
EXP(n)	Número e elevado a n.
FLOOR(n)	Entero más grande o igual a n.
LN(n)	Devuelve el logaritmo neperiano de n.
LOG(n1,n2)	Logaritmo en base n1 de n2.
MOD(n1,n2)	Devuelve el resto de dividir n1 entre n2.
POWER(n1,n2)	Devuelve n1 elevado a n2.
ROUND(n1,n2)	Redondea n1 con n2 decimales.
SVGN(n)	Devuelve –1 si el número es menor de 0, 0 si es igual a 0 y 1 si es mayor de 0.
SIN(n)	Seno de n.
SQRT(n)	Raíz cuadrada de n, que ha de ser siempre mayor de 0.
TAN(n)	Tangente de n.
TRUNC(n1,n2)	Trunca n1 con n2 decimales.

Funciones	Descripción
ASCII(C)	Devuelve el ASCII del carácter.
CHR(n)	Devuelve el carácter correspondiente al número n.
CONCAT (c1, c2)	Concatena 2 cadenas.
INITCAP(c)	Transforma la primera letra a mayúscula.
LENGHT(c)	Longitud de la cadena.
LOWER(c)	Transforma a minúsculas.
LPAD (c1, n, c2)	Justifica a la derecha el valor del carácter c1 n posiciones y los blancos los reemplaza por el carácter c2.
LTRIM(c1, c2)	Suprime caracteres a la izquierda hasta el primer carácter que no está en c2. Si se omite c2 lo pone todo en blanco.
REPLACE(c1,c2,c3)	Devuelve c1 con la cadena c2, reemplazada por c3 y así sucesivamente.
RPAD(c1,n,c2)	Iguala a la derecha.
RTRIM(c1,c2)	Igual que LTRIM pero por la derecha.
SUBSTR(c1,m,n)	Substrae de la cadena c1, y desde el carácter m, n caracteres.
TRANSLATE(c1,c2,c3)	Devuelve la cadena c1 con cada carácter de c2 que contenga reemplazado por el carácter de c3 con el que se corresponda.
UPPER(c)	Convierte c a mayúsculas.

Funciones	Descripción
TO_CHAR(c1,'fmt')	Convierte un número o fecha a cadena de caracteres.
TO_DATE(c1,'fmt')	Convierte una cadena de caracteres a un número.
TO_NUMBER(c1,'fmt')	Convierte una cadena de caracteres a una fecha.

Funciones	Descripción
ADD_MONTH(fecha,n)	Agrega meses a una fecha.
LAST_DAY(fecha)	Último día del mes.
MONTHS_BETWEEN(fecha1,fecha2)	Número de meses entre dos fechas.
NEXT_DAY(fecha, 'caracter')	Próximo día de la fecha especificada.

Funciones	Descripción
DUMP(expr)	Devuelve la expresión en el formato interno de Oracle.
GREATEST(lista de valores)	Da como resultado el valor más grande de la lista.
LEAST (lista de valores)	Da como resultado el valor más pequeño de la lista.
NVL(expr1, expr2)	Reemplaza los nulos por otro valor.
USER	Devuelve al usuario conectado.
USERENV(opción)	Devuelve información sobre el usuario. Las opciones pueden ser: • SESSIONID: Se identifica la sesión. • TERMINAL: Identificador del terminal. • LANGUAGE: Lenguaje activo.
DECODE	Realiza la función de IF/THEN/ELSE.

Bloques anidados

También se pueden anidar bloques de forma que las variables tengan su propio ámbito de ejecución:

```
DECLARE
        X BINARY_INTEGER;
BEGIN
        DECLARE
                Y BINARY_INTEGER;
        BEGIN
                ..............
        EXCEPTION
                .........
        END;
EXCEPTION
        .......
END;
```

Las sentencias pueden ser anidadas cuando el bloque PL/SQL lo requiera. Un bloque anidado se convierte en una sentencia. La sección de excepciones también puede contener bloques, pero generalmente el lugar donde se suele trabajar es en la zona ejecutable. El ámbito de un objeto es la región de programa que hace referencia a dicho objeto.

Las variables tienen como ámbito su propio bloque. En este caso, la variable Y sólo será reconocida por el segundo bloque, y una vez finalizado este el bloque principal no la reconocerá, por el contrario, la variable X tiene como ámbito todo el bloque, ya que tiene vigencia en su propio bloque y en el bloque anidado, ya que este pertenece al bloque principal.

Sentencias DML – COMMIT y ROLLBACK

PL/SQL soporta tanto sentencias DML (select, insert, update y delete) como el uso de COMMIT y ROLLBACK para guardar los cambios hechos en la BBDD o deshacerlos, según necesitemos.