SENA

ANÁLISIS Y DESARROLLO DE SOFTWARE

COMPETENCIA

ANÁLISIS DE LA ESPECIFICACIÓN DE REQUISITOS DEL SOFTWARE.

RESULTADO DE APRENDIZAJE

DESARROLLAR PROCESOS LÓGICOS A TRAVÉS DE LA IMPLEMENTACIÓN DE ALGORITMOS.

EVIDENCIA
PROGRAMACIÓN ESTRUCTURADA DESARROLLADA EN PL-SQL- BLOQUES ANONIMOS

.

TUTOR
RAMON EMILIO GONZALEZ RODRIGUEZ

APRENDIZ
SIMON DAVID LOPEZ CESPEDES
BOGOTA D.C.
SENA CENTRO DE DISEÑO Y METROLOGÍA
OCTUBRE 2022

INTRODUCCIÓN

En el presente se conocerán los lenguajes de programación especializados en Bases de datos como, SQL, MySQL, y PLsql, siendo este último el lenguaje especializado en este documento .

Tener en cuenta que este tipo de lenguajes presenta una sintaxis más compleja debido a que se deben crear a prueba de errores, en el almacenamiento de datos, e interrupciones que pueda presentar

¿Qué es PL/SQL?

PL/SQL es un lenguaje estructurado en bloques. Los programas de PL/SQL son bloques lógicos que pueden contener cualquier número de subbloques anidados. Pl/SQL significa "Extensión de lenguaje de procedimiento de SQL" que se utiliza en Oracle. PL/SQL está integrado con la base de datos Oracle (desde la versión 7). Las funcionalidades de PL/SQL generalmente se amplían después de cada lanzamiento de la base de datos Oracle. Aunque PL/SQL está estrechamente integrado con el lenguaje SQL, agrega algunas restricciones de programación que no están disponibles en SQL.

Funcionalidades PL/SQL

PL/SQL incluye elementos de lenguaje procedimental como condiciones y bucles. Permite declaración de constantes y variables, procedimientos y funciones, tipos y variables de esos tipos y disparadores. Puede admitir Array y manejar excepciones (errores de tiempo de ejecución). Tras la implementación de la versión 8 de la base de datos Oracle se han incluido características asociadas a la orientación a objetos. Puede crear unidades PL/SQL como procedimientos, funciones, paquetes, tipos y disparadores, etc. que se almacenan en la base de datos para que las aplicaciones las reutilicen.

Con PL/SQL, puede utilizar sentencias SQL para manipular datos de Oracle y sentencias de flujo de control para procesar los datos.

El PL/SQL es conocido por su combinación del poder de manipulación de datos de SQL con el poder de procesamiento de datos de los lenguajes de procedimiento. Hereda la solidez, seguridad y portabilidad de Oracle Database.

PL/SQL no distingue entre mayúsculas y minúsculas, por lo que puede usar letras minúsculas o mayúsculas, excepto dentro de cadenas y caracteres literales. Una línea de texto PL/SQL contiene grupos de caracteres conocidos como unidades léxicas. Se puede clasificar de la siguiente manera

Variables PL/SQL

Una variable es un nombre significativo que facilita a un programador almacenar datos temporalmente durante la ejecución del código. Le ayuda a manipular datos en programas PL/SQL. No es más que un

nombre dado a un área de almacenamiento. Cada variable en PL/SQL tiene un tipo de datos específico que define el tamaño y el diseño de la memoria de la variable.

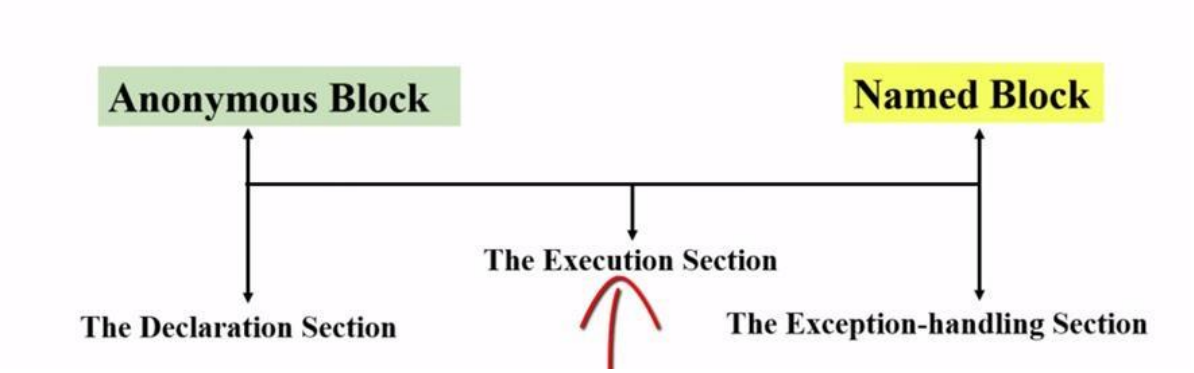
Una variable no debe exceder los 30 caracteres. Su letra seguida opcionalmente por más letras, signos de dólar, números, guión bajo, etc

La siguiente es la sintaxis para declarar una variable:

nombre_variable [CONSTANTE] tipo de datos [NO NULO] [:= | valor_inicial PREDETERMINADO]

Aquí, variable_name es un identificador válido en PL/SQL y el tipo de datos debe ser un tipo de datos PL/SQL válido. Un tipo de datos con límite de tamaño, escala o precisión se denomina declaración restringida. La declaración restringida necesita menos memoria que la declaración no restringida.

TIPOS DE BLOQUES EN PL_SQL



Una sentencia de bloque PL/SQL anónima es una sentencia ejecutable que puede contener sentencias de control PL/SQL y sentencias SQL. Se puede utilizar para implementar lógica procedimental en un lenguaje de script. En contextos PL/SQL, esta sentencia la puede compilar y ejecutar el servidor de datos DB2

```
cta = 1200
nhr = int(input("digite la cantidad"))
total = cta * nhr
print("total", total)
```

Nota: En este manual de procedimientos solamente vamos a trabajar bloques PL/SQL anónimos.

Ejemplo #1: Calcular el monto a pagar en una cabina de Internet si el costo por hora es de 1200 pesos.

Ejemplo 2

```
1  Declare
2  /* Mensaje declarando la variable del mensaje */
3  Mensaje varchar2(15) := 'hola noche';
4  Begin
5  /* Esta es la sintaxis "dbms_output.puy_line" para mostrar la salida */
6  dbms_output.put_line(mensaje);
7  End;
```

Ejemplo #2: Hacer un programa que me sume enteros positivos, las variables deben ser de tipo number

Nota: para poder concatenar una cadena se utiliza los caracter || utilizando la combinación de teclas Alt + 124

Ejercicio #1: Calcular el monto a pagar en una cabina de Internet si el costo por hora es de 1200 pesos

Ejemplo 2

```
Declare
/* Variables a usar */
Horas number;
Costo number;
Pagar number;

Begin
/* Procedimiento */
Horas := 5;
Costo := 1200;
Pagar := Horas*Costo;

dbms_output.put_line('El valor a pagar es ' || Pagar);

End;
```

Ejercicio #2: Si una flota viaja a una velocidad 45km/h al centro de la ciudad y se gasta 2 horas. ¿ A qué velocidad viaja si se gasta N horas?

Ejemplo 2

```
Declare
/* Variables a usar */
Distancia number;
Tiempo number;
Velocidad number;

Begin
/* Procedimiento */
Distancia := 90; -- Distancia fija
Tiempo := 1; -- Horas desplazadas
Velocidad := Distancia/Tiempo;

dbms_output.put_line('Se viaja a una velocidad de ' || Velocidad || ' Km/h');

End;
```

Ejercicio #3: Calcular el nuevo salario de un empleado si obtuvo un incremento del 10 % sobre su salario actual y un descuento de 3, 5% por aportes obligatorios a parafiscales que corresponden al empleador en beneficio de sus trabajadores.

Ejemplo 2

```
Declare
/* Variables a usar */
Incremento number;
descuento number;
Salario number;

Begin
/* Procedimiento */
Incremento := 0.1;
Descuento := 0.035;
Salario := 1000;

Salario := Salario + Salario*Incremento - Salario*Descuento;

dbms_output.put_line('El nuevo salario a pagar es ' || Salario || ' COP');

End;
```

Ejercicio #4: Hacer un programa que me capture por teclado un valor en pesos colombianos y me los convierta a dólares y euros.

Ejemplo 2

```
Declare
/* Variables a usar */
Cambio number;
CambioDolar number;
CambioEuro number;
Dolar number;
Euro number;

Begin
/* Procedimiento */
Cambio := 10000;
Dolar := 4835;
Euro := 4818;

CambioDolar := ROUND(Cambio/Dolar, 3);
CambioEuro := ROUND(Cambio/Euro, 3);

dbms_output.put_line('El cambio en Dolar es: ' || CambioDolar || chr(13)||chr(10) || 'El cambio en Euro es: ' || CambioEuro);

End;
```

Ejercicio #5: Si un usuario desea invertir sus ganancias en un banco y desea saber cuánto dinero ganará después de un año si el banco paga a razón de 1,5% mensual.

Ejercicio 11

```
1  Declare
2  /* Variables a usar */
3  Interes number;
4  Ganancia number;
5
6  Begin
7  /* Procedimiento */
8  Interes := 1.8;
9  Ganancia := 1000;
10
11 Ganancia := ROUND((Ganancia + Ganancia*Interes),3);
12
13 dbms_output.put_line('Su ganancia es: ' || Ganancia );
14
15 End;
```

Ejercicio #6: Si un aprendiz desea saber cuál será su calificación final en programación de software. Dicha calificación se compone de cuatro notas.

Ejercicio 6

```
1  Declare
2  /* Variables a usar */
3  Nota1 number;
4  Nota2 number;
5  Nota3 number;
6  Resultado number;
7
8  Begin
9  /* Procedimiento */
10 Nota1 := 5;
11 Nota2 := 1;
12 Nota3 := 2;
13
14 Resultado := ROUND((Nota1 + Nota2 + Nota3)/3,3) ;
15
16
17 dbms_output.put_line('Su nota final es: ' || Resultado);
18
19 End;
```

Ejercicio #7: Elevar al cubo un número y dividirlo en dos.

Ejercicio 6

```
Declare
/* Variables a usar */
NumIngresado number;
Resultado number;

Begin
/* Procedimiento */
NumIngresado := 8888;
Resultado := ROUND(SQUARE(NumIngresado)/2,3);

dbms_output.put_line('El resultado es: ' || Resultado);

End;
```

Ejercicio #8: Convertir 10 Hm a Km y dm.

Ejercicio 6

```
Declare
/* Variables a usar */
Km number;
dm number;
Conversion number;
Conv_Km number;
Conv_dm number;

Begin
/* Procedimiento */
Conv_Km := 0.1;
Conv_dm := 1000;
Conversion := 10000;

Km := ROUND(Conversion*Conv_Km,3);
dm := ROUND(Conversion*Conv_dm,3);

dbms_output.put_line('Km: ' || Km || chr(13)||chr(10) || 'dm: ' || dm );

End;
```

Ejercicio #9: Si un metro 1m equivale a 0.3048 pies (ft) y 0.0254 pulgadas(in). ¿convertir la distancia en metros a pies y pulgadas?

Ejercicio 6

```
Declare
/* Variables a usar */
Ft number;
IIn number;
Conversion number;
Conv_Ft number;
Conv_In number;

Begin
/* Procedimiento */
Conv_Ft := 0.3048;
Conv_In := 0.0254;
Conversion := 10;

Ft := ROUND(Conversion*Conv_Ft,3);
IIn := ROUND(Conversion*Conv_In,3);

dbms_output.put_line('Ft: ' || Ft || chr(13)||chr(10) || 'In: ' || IIn );

End;
```

Ejercicio #10: Hacer un programa que me convierta 25 Kg a libras y arrobas.

Ejercicio 10

```
Declare
/* Variables a usar */
Kg number;
Arr number;
Conversion number;
Conv_Kg number;
Conv_Arr number;

Begin
/* Procedimiento */
Conv_Kg := 2.20462;
Conv_Arr := 0.088;
Conversion := 10;

Kg := ROUND(Conversion*Conv_Kg,3);
Arr := ROUND(Conversion*Conv_Arr,3);

dbms_output.put_line('Kg: ' || Kg || chr(13)||chr(10) || 'Arr: ' || Arr );

End;
```

Ejercicio #11: Hacer un programa que me permita ingresar el valor de un artículo y la cantidad de una venta. Se le debe calcular el IVA del 19% posteriormente se debe imprimir el total a

Ejercicio 11

```
Declare
/* Variables a usar */
Cantidad number;
Valor number;
Pagar number;

Begin
/* Procedimiento */
Cantidad := 2;
Valor := 100;

Pagar := ROUND(((Valor*0.19)+Valor)*Cantidad,3);

dbms_output.put_line('El monto a pagar es: ' || Pagar );

End;
```

Ejercicio #12: Hacer un programa que me convierta 30Mb a Tb.

Ejercicio 11

```
Declare
/* Variables a usar */
Mb number;
Conversion number;
Conv_Tb number;

Begin
/* Procedimiento */
Conv_Tb := 0.000001;
Conversion := 10000;

Mb := ROUND(Conversion*Conv_Tb,3);

dbms_output.put_line('Mb: ' || Mb );

End;
```

Ejercicio #13: Hacer un programa que me convierta 16000 Tb a Gb.

Ejercicio 11

```
1  Declare
2  /* Variables a usar */
3  Gb number;
4  Conversion number;
5  Conv_Gb number;
6
7  Begin
8  /* Procedimiento */
9  Conv_Gb := 1000;
10 Conversion := 10;
11
12 Gb := ROUND(Conversion*Conv_Gb,3);
13
14 dbms_output.put_line('Gb: ' || Gb );
15
16 End;
```

Ejercicio #14: Si el área de un círculo es π multiplicado por el radio al cuadrado ($A = \pi r^2$). hacer un programa que capture por teclado el diámetro de un círculo y postreramente me muestre el área del círculo.

Ejercicio 11

```
1  Declare
2  /* Variables a usar */
3  Diametro number;
4  Area number;
5
6  Begin
7  /* Procedimiento */
8  Diametro := 10;
9
10 Area := ROUND(3.141592654*SQUARE(Diametro/2),3);
11
12 dbms_output.put_line('El área del circulo es: ' || Area );
13
14 End;
```

Ejercicio #15: A un objeto en reposo se le aplica una fuerza x provocando que este se desplace y adquiera una aceleración de 2.8 m/s^2. Determine la masa de dicho objeto.

Ejercicio 11

```
Declare
/* Variables a usar */
Fuerza number;
Masa number;
Aceleracion number;

Begin
/* Procedimiento */
Aceleracion := 2.8;
Fuerza := 5;

Masa := ROUND(Fuerza/Aceleracion,3);

dbms_output.put_line('La masa del cuerpo es: ' || Masa );

End;
```

SECCIÓN #2:

PL/SQL SI

PL/SQL es compatible con las características del lenguaje de programación, como declaraciones condicionales y declaraciones iterativas. Sus construcciones de programación son similares a las que usa en lenguajes de programación como Java y C++.

Sintaxis para la instrucción IF:

Hay diferentes sintaxis para la instrucción IF-THEN-ELSE.

Hay situaciones en la vida real en las que necesitamos tomar algunas decisiones y, en base a estas decisiones, decidimos qué debemos hacer a continuación. También surgen situaciones similares en la programación en las que necesitamos tomar algunas decisiones y, en base a estas decisiones, ejecutaremos el siguiente bloque de código.

Las declaraciones de toma de decisiones en los lenguajes de programación deciden la dirección del flujo de ejecución del programa. Las declaraciones de toma de decisiones disponibles en pl/SQL son:

- si entonces declaración
- declaraciones si entonces más
- sentencias si-entonces anidadas
- si-entonces-elsif-entonces-otro escalera

si entonces la declaración es la declaración de toma de decisiones más simple. Se utiliza para decidir si una determinada instrucción o bloque de instrucciones se ejecutará o no, es decir, si una determinada co

Operador	Operación
**	exponencial
+, -	identidad, negación
*, /	multiplicación, división
+, -, ||	suma, resta, concatenación
=, <, >, <=, >=, <>, !=, ~=, ^=, IS NULL, LIKE, BETWEEN, IN	comparación
NOT	negación
AND	conjunción
OR	inclusión

Ejercicio #1: Hacer un programa que me capture por teclado el número 86 y un mensaje que diga “Tutela artículo 86” y el número 23 me debe mostrar un mensaje que diga “Derecho de petición artículo 23”

```
Declare
/* Variables a usar */
Numero number;
Cond1 number;
Cond2 number;

Begin
/* Procedimiento */
Numero := 2;

Cond1 := 86;
Cond2 := 23;

IF Cond1 = Numero THEN
        dbms_output.put_line('Tutela artículo 86');
ELSIF Cond2 = Numero THEN
        dbms_output.put_line('Derecho de petición artículo 23');
END IF;


End;
```

CASE #1:

```
Declare
/* Variables a usar */
Numero number := 8;
Cond1 number := 23;
Cond2 number := 86;

Begin
/* Procedimiento */
   CASE

   WHEN Cond1 = Numero THEN dbms_output.put_line('Derecho de petición artículo
23');
   WHEN Cond2 = Numero THEN dbms_output.put_line('Tutela artículo 86');

   END CASE;
/* Fin */
End;
```

Ejercicio #2: En Mega Plaza de Madrid se hace un 25% de descuento a los clientes cuya compra supere los $70000 ¿Cuál será la cantidad que pagará una persona por su compra?

```
Declare
/* Variables a usar */
Vpagar number;
Limite number;

Begin
/* Procedimiento */
Vpagar := 100000;
Limite := 70000;

IF Limite < Vpagar THEN
        dbms_output.put_line('El salario a pagar es: ' || (Vpagar*0.75));
ELSE
   dbms_output.put_line('El salario a pagar es: ' || Vpagar);
END IF;


End;
```

CASE #2:

```
Declare
/* Variables a usar */
Vpagar number := 100000;
Limite number := 70000;

Begin
/* Procedimiento */

   CASE

   WHEN Limite <= Vpagar THEN dbms_output.put_line('El salario a pagar es: ' ||
(Vpagar*0.75));
   ELSE dbms_output.put_line('El salario a pagar es: ' || Vpagar);

   END CASE;

End;
```

Ejercicio #3: Hacer un programa que me capture por teclado el día 10 y me escriba “petición en curso”, el día 15 y me escriba “solicitud de copias “y el 30 día me escriba “consultas”. si el usuario digita un día distinto me debe mostrar un mensaje que diga “día no correspondiente al trámite del derecho de petición”

```
Declare
dia number;

Begin
dia := 1;

if dia = 10 then
   dbms_output.put_line('Petición en curso');
elsif dia = 15 then
   dbms_output.put_line('Solicitud copias');
elsif dia = 30 then
   dbms_output.put_line('consultas');
else
   dbms_output.put_line('dia no correspondiente al trámite del derecho de
petición');
end if;

End;
```

CASE #3:

```
Declare
/* Variables a usar */
dia number := 10;

Begin
/* Procedimiento */

   CASE
   WHEN dia = 10 THEN dbms_output.put_line('Petición en curso');
   WHEN dia = 15 THEN dbms_output.put_line('Solicitud copias');
   WHEN dia = 30 THEN dbms_output.put_line('consultas');
   ELSE dbms_output.put_line('Dia no correspondiente al trámite del derecho de
petición');
   END CASE;

/* Fin */
End;
```

Ejercicio #4: Diseñar un diagrama de flujo que me evalué que se busca con un derecho de petición. Captura por teclado un número que se evalúa de la siguiente manera; si digita el número 1 debe escribir “reconocimiento a un derecho”, 2 “la intervención de una entidad o funcionario”, 3 “la resolución de una situación jurídica”, 4 “la prestación de un servicio”, 5 “requerir información “, 6” consultar, examinar y requerir copias de documento”, 7 “formular consultas, quejas y reclamos” y 8 “interponer reclamos”. Si digita un número distinto mostrar un mensaje que diga no se puede evaluar el derecho de petición.

```
Declare
/* Variables a usar */
Dia number;

Begin
/* Procedimiento */
Dia := 0;

IF Dia = 1 THEN
        dbms_output.put_line('reconocimiento a un derecho');
ELSIF Dia = 2 THEN
   dbms_output.put_line('la intervención de una entidad o funcionario');
ELSIF Dia = 3 THEN
   dbms_output.put_line('la resolución de una situación jurídica');
ELSIF Dia = 4 THEN
        dbms_output.put_line('la prestación de un servicio');
ELSIF Dia = 5 THEN
   dbms_output.put_line('requerir información');
ELSIF Dia = 6 THEN
   dbms_output.put_line('consultar, examinar y requerir copias de documento');
ELSIF Dia = 7 THEN
   dbms_output.put_line('formular consultas, quejas y reclamos');
ELSIF Dia = 8 THEN
   dbms_output.put_line('interponer reclamos');

ELSE
   dbms_output.put_line('no se puede evaluar el derecho de petición');

END IF;


End;
```

CASE #4:

```
Declare
/* Variables a usar */
Dia number := 1;

Begin
/* Procedimiento */

   CASE
   WHEN Dia = 1 THEN dbms_output.put_line('reconocimiento a un derecho');
   WHEN Dia = 2 THEN dbms_output.put_line('la intervención de una entidad o
funcionario');
   WHEN Dia = 3 THEN dbms_output.put_line('la resolución de una situación
jurídica');
   WHEN Dia = 4 THEN dbms_output.put_line('la prestación de un servicio');
   WHEN Dia = 5 THEN dbms_output.put_line('requerir información');
   WHEN Dia = 6 THEN dbms_output.put_line('consultar, examinar y requerir copias
de documento');
   WHEN Dia = 7 THEN dbms_output.put_line('formular consultas, quejas y
reclamos');
   WHEN Dia = 8 THEN dbms_output.put_line('interponer reclamos');

   ELSE dbms_output.put_line('no se puede evaluar el derecho de petición');

   END CASE;

/* Fin */
End;
```

Ejercicio #5: Un aprendiz va a llenar un tanque bajo con agua de 250 litros, primero le suministra 1/5 debe mostrar el mensaje “50Lt”, 2/5 debe mostrar el mensaje “100Lt”, 3/5 debe mostrar el mensaje “150Lt”, 4/5 debe mostrar el mensaje “200Lt”, 5/5 debe mostrar el mensaje “250Lt”. Capturar por teclado el número racional decimal finito según la fracción.

```
Declare
litros varchar(10);

Begin
litros := '1/5';

if litros = '1/5' then
   dbms_output.put_line('Son 50 Lts');
elsif litros = '2/5' then
   dbms_output.put_line('Son 100 Lts');
elsif litros = '3/5' then
   dbms_output.put_line('Son 150 Lts');
elsif litros = '4/5' then
   dbms_output.put_line('Son 200 Lts');
elsif litros = '5/5' then
   dbms_output.put_line('Son 250 Lts');
end if;

End;
```

CASE #5:

```
Declare
/* Variables a usar */
litros varchar(10) := '1/5';

Begin
/* Procedimiento */

   CASE
   WHEN litros = '1/5' THEN dbms_output.put_line('Son 50 Lts');
   WHEN litros = '2/5' THEN dbms_output.put_line('Son 100 Lts');
   WHEN litros = '3/5' THEN dbms_output.put_line('Son 150 Lts');
   WHEN litros = '4/5' THEN dbms_output.put_line('Son 200 Lts');
   WHEN litros = '5/5' THEN dbms_output.put_line('Son 250 Lts');
   END CASE;

/* Fin */
End;
```

Ejercicio #6: Un obrero necesita calcular su salario mensual , el cual se obtiene de la siguiente manera: Si su salario mínimo es menor o igual a $737.717 las horas extras se pagarán a $6.453 y si su salario es dos salarios mínimos mayor o igual $1.475.434 las horas extras se pagarán a $12.908. capturar por teclado el salario, las horas extras y mostrar el salario neto a pagar al obrero.

```
Declare
salario integer;
horasTotales integer;

Begin
salario := 1000000;
horasTotales := 10;

if salario > 737717 then
   dbms_output.put_line('El salario del empleado es ' || (salario +
horasTotales*12908));
elsif salario <= 737717 then
   dbms_output.put_line('El salario del empleado es ' || (salario +
horasTotales*6453));
end if;

End;
```

CASE #6:

```
Declare
salario number := 737715;
horasTotales number := 10;

Begin

   CASE
   WHEN salario >= 1475434 THEN dbms_output.put_line('El salario del empleado es
' || (salario + horasTotales*12908));
   WHEN salario <= 737717 THEN dbms_output.put_line('El salario del empleado es '
|| (salario + horasTotales*6453));
   END CASE;

End;
```

Ejercicio #7: Si un empleado recibe un ingreso mensual, pero el empleador le dice que el primer mes le incrementa 30%, el segundo mes 40% y tercer mes 35%. Se debe capturar por teclado el ingreso del empleado y el mes, postreramente se debe mostrar el valor neto del ingreso mensual del empleado.

```
Declare
/* Variables a usar */
Salario number;
Mes number;

Begin
/* Procedimiento */
Salario := 1000;
Mes := 3;

IF Mes = 1 THEN
        dbms_output.put_line('Su salario es: ' || (Salario+Salario*0.3));
ELSIF Mes = 2 THEN
        dbms_output.put_line('Su salario es: ' || (Salario+Salario*0.4));
ELSIF Mes = 3 THEN
        dbms_output.put_line('Su salario es: ' || (Salario+Salario*0.35));
END IF;

End;
```

CASE #7:

```
Declare
/* Variables a usar */
Salario number := 1000;
Mes number := 2;

Begin
/* Procedimiento */
   CASE
   WHEN Mes = 1 THEN dbms_output.put_line('Su salario es: ' ||
(Salario+Salario*0.3));
   WHEN Mes = 2 THEN dbms_output.put_line('Su salario es: ' ||
(Salario+Salario*0.4));
   WHEN Mes = 3 THEN dbms_output.put_line('Su salario es: ' ||
(Salario+Salario*0.35));
   END CASE;
/* Fin */
End;
```

Ejercicio #8: Hacer un programa que me capture por teclado la unidad de medida en libras y me las convierta a kilos, arrobas y toneladas. Si su valor equivalente no corresponde a la unidad buscada de mostrar el valor que se capturó en libras por teclado.

```
Declare
libras int;

Begin
libras := 2000;

if libras > 0 then
   dbms_output.put_line('las libras a kilos son ' || (libras/2));
   dbms_output.put_line('las libras a arrobas son ' || (libras/15));
   dbms_output.put_line('las libras a toneladas son ' || (libras/2000));
end if;

End;
```

CASE #8:

```
Declare
libras int := 2000;

Begin
   CASE
   WHEN libras > 0 then
      dbms_output.put_line('las libras a kilos son ' || (libras/2));
      dbms_output.put_line('las libras a arrobas son ' || (libras/15));
      dbms_output.put_line('las libras a toneladas son ' || (libras/2000));
   END CASE;
End;
```

Ejercicio #9: Si el diámetro es superior a 1.4 debe mostrarse el mensaje “La rueda es para un vehículo grande”. Si es menor o igual a 1.4 pero mayor que 0.8 debe mostrarse por el mensaje “La rueda es para un vehículo mediano”. Si no se cumplen ninguna de las condiciones anteriores debe mostrar un el mensaje “La rueda es para un vehículo pequeño”.

```
Declare
/* Variables a usar */
Diametro number;

Begin
/* Procedimiento */
Diametro := 1;

IF Diametro > 1.4 THEN
        dbms_output.put_line('La rueda es para un vehículo grande');

ELSIF 0.8 <= Diametro AND Diametro <= 1.4 THEN
        dbms_output.put_line('La rueda es para un vehículo mediano');

ELSE
        dbms_output.put_line('La rueda es para un vehículo pequeño');
END IF;

End;
```

CASE #9:

```
Declare
/* Variables a usar */
Diametro number := 0;

Begin
/* Procedimiento */
   CASE
   WHEN Diametro > 1.4 THEN dbms_output.put_line('La rueda es para un vehículo
grande');
   WHEN 0.8 <= Diametro AND Diametro <= 1.4 THEN dbms_output.put_line('La
rueda es para un vehículo mediano');

   ELSE dbms_output.put_line('La rueda es para un vehículo pequeño');
   END CASE;
/* Fin */
End;
```

Ejercicio #10: Si el diámetro es superior a 1.4 con un grosor inferior a 0.4, ó si el diámetro es menor o igual a 1.4 pero mayor que 0.8, con un grosor inferior a 0.25, deberá mostrarse el mensaje “El grosor para esta rueda es inferior al recomendado”.

```
Declare
diametro float;
grosor float;

Begin
diametro := 1.5;
grosor := 0.3;


if (diametro > 1.4 and grosor < 0.4) or (diametro <= 1.4 and diametro > 0.8 and
grosor < 0.25) then
   dbms_output.put_line('el grosor para esta rueda es inferior');
end if;

End;
```

CASE #10:

```
Declare
diametro float := 1.5;
grosor float := 0.3;

Begin
   CASE
   WHEN (diametro > 1.4 and grosor < 0.4) or (diametro <= 1.4 and diametro > 0.8
and grosor < 0.25) THEN
      dbms_output.put_line('el grosor para esta rueda es inferior');
   END CASE;
End;
```

Ejercicio #11: Capturar por teclado la calificación de un estudiante y posteriormente se debe mostrar los siguientes Conceptos: si la calificación fue del rango de 1.0 a 2.9 debe arrojar un resultado que diga “pierde.” de 3.0 a 4.9 debe arrojar un resultado que diga “bueno” Y si la clasificación es 5.0 debe arrojar un mensaje que diga “excelente”; de lo contrario si alguna de las calificaciones no están dentro del rango aceptado debe arrojar un mensaje que diga “esa nota no es válida” ósea si es mayor de 5.1.

```
Declare
/* Variables a usar */
Nota float;

Begin
/* Procedimiento */
Nota := 3;

IF Nota BETWEEN 1 AND 2.9 THEN
        dbms_output.put_line('Pierde');

ELSIF Nota BETWEEN 3 AND 4.9 THEN
        dbms_output.put_line('Bueno');

ELSIF Nota = 5 THEN
        dbms_output.put_line('Excelente');

END IF;

End;
```

CASE #11:

```
Declare
/* Variables a usar */
Nota float := 5;

Begin
/* Procedimiento */
   CASE
   WHEN Nota BETWEEN 1 AND 2.9 THEN dbms_output.put_line('Pierde');
   WHEN Nota BETWEEN 3 AND 4.9 THEN dbms_output.put_line('Bueno');
   WHEN Nota = 5 THEN dbms_output.put_line('Excelente');
   END CASE;
End;
```

Sección #3

“Bucles”

1. Hacer un programa que muestre la siguiente sucesión: 1,4,7,10,13,16,19,22,25,28,31,34,37 **y** 40.

```
--While

DECLARE
 i NUMBER := 1;
BEGIN
 WHILE i <= 40
 LOOP
  DBMS_OUTPUT.PUT_LINE( 'Contador : ' || i );
  i := i + 3;
 END LOOP;
END;
```

```
--For

DECLARE
 i NUMBER := 1;
BEGIN
 FOR N IN 1..14
 LOOP
  DBMS_OUTPUT.PUT_LINE( 'Contador : ' || i );
  i := i + 3;
 END LOOP;
END;
```

2. Hacer un programa que muestre la siguiente sucesión: 3,8,13,18,23,28,33,38,43, 48 y 53.

```
--While

DECLARE
 i NUMBER := 3;
BEGIN
 WHILE i <= 53
 LOOP
  DBMS_OUTPUT.PUT_LINE( 'Contador : ' || i );
  i := i + 5;
 END LOOP;
END;
```

```
--For

DECLARE
 i NUMBER := 3;
BEGIN
 FOR N IN 1..11
 LOOP
  DBMS_OUTPUT.PUT_LINE( 'Contador : ' || i );
  i := i + 5;
 END LOOP;
END;
```

3. Hacer un programa que muestre la siguiente sucesión geométrica: 3, 9, 27, 81, 243,729 y 2187.

```
--While

DECLARE
 i NUMBER := 3;
 N NUMBER := 2;
BEGIN
  WHILE i <= 2187
  LOOP
    DBMS_OUTPUT.PUT_LINE( 'Contador : ' || i );
    i := 3**n;
    n := n + 1;
  END LOOP;
END;
```

```
--For

DECLARE
 i NUMBER := 3;
 N NUMBER := 2;
BEGIN
  FOR P IN 1..7
  LOOP
    DBMS_OUTPUT.PUT_LINE( 'Contador : ' || i );
    i := 3**n;
    n := n + 1;
  END LOOP;
END;
```

4. Hacer un programa que muestre la sumatoria o “serie” de la sucesión 1,2,3,4,5,6,7,8,9,10,11,12,13,14 y 15.

```
--While

DECLARE
 Contador NUMBER := 0;
 Total NUMBER := 0;
BEGIN
  WHILE Contador <= 14
  LOOP
    Contador := Contador + 1;
    Total := Total + Contador;
  END LOOP;
  DBMS_OUTPUT.PUT_LINE( 'Total : ' || Total );
END;
```

```
--For

DECLARE
 Contador NUMBER := 0;
 Total NUMBER := 0;
BEGIN
  FOR P IN 1..15
  LOOP
    Contador := Contador + 1;
    Total := Total + Contador;
  END LOOP;
  DBMS_OUTPUT.PUT_LINE( 'Total : ' || Total );
END;
```

5. Hacer un programa que muestre la sumatoria o “serie ” de la sucesión 3,8,13,18,23,28,33,38,43,48 y 53.

```
--While

DECLARE
 Contador NUMBER := -2;
 Total NUMBER := 0;
BEGIN
  WHILE Contador < 53
  LOOP
    Contador := Contador + 5;
    Total := Total + Contador;
  END LOOP;
  DBMS_OUTPUT.PUT_LINE( 'Total : ' || Total );
END;
```

```
--FOR

DECLARE
 Contador NUMBER := -2;
 Total NUMBER := 0;
BEGIN
  FOR P IN 1..11
  LOOP
    Contador := Contador + 5;
    Total := Total + Contador;
  END LOOP;
  DBMS_OUTPUT.PUT_LINE( 'Total : ' || Total );
END;
```

6. Hacer un programa que muestre la sumatoria o “serie” de la sucesión 3, 9, 27, 81, 243,729 y 2187.

```
--While

DECLARE
 Contador NUMBER := 0;
 Total NUMBER := 0;
BEGIN
  WHILE Contador <= 6
  LOOP
    Contador := Contador + 1;
    Total := Total + 3**Contador;
  END LOOP;
  DBMS_OUTPUT.PUT_LINE( 'Total : ' || Total );
END;
```

```
--FOR

DECLARE
 Contador NUMBER := 0;
 Total NUMBER := 0;
BEGIN
  FOR P IN 1..7
  LOOP
    Contador := Contador + 1;
    Total := Total + 3**Contador;
  END LOOP;
  DBMS_OUTPUT.PUT_LINE( 'Total : ' || Total );
END;
```

7. Hacer un programa que me muestre las tablas de multiplicar del 1, 2 y 3.

```
--While

DECLARE
 Contador NUMBER := 0;
 Fila NUMBER;
BEGIN
  WHILE Contador < 3
  LOOP
    Contador := Contador + 1;
    DBMS_OUTPUT.PUT_LINE( 'Tabla de multiplicar del ' || Contador );
    Fila := 1;
      WHILE Fila <= 10
      LOOP
        DBMS_OUTPUT.PUT_LINE( Contador || 'x' || Fila || '= ' || Contador*Fila );
        Fila := Fila + 1;
      END LOOP;
    DBMS_OUTPUT.PUT_LINE(' ');
  END LOOP;
END;
```

```
--For

DECLARE
 Contador NUMBER := 0;
 Fila NUMBER;
BEGIN
  FOR Tabla IN 1..3
  LOOP
    Contador := Contador + 1;
    DBMS_OUTPUT.PUT_LINE( 'Tabla de multiplicar del ' || Contador );
    Fila := 1;
      FOR Linea IN 1..10
      LOOP
        DBMS_OUTPUT.PUT_LINE( Contador || 'x' || Fila || '= ' || Contador*Fila );
        Fila := Fila + 1;
```

```
    END LOOP;
  END LOOP;
END;
```

8. Programa que calcula y visualiza por pantalla el factorial de todos los valores numéricos enteros

Entre 1 y 10. Teniendo en cuenta el siguiente sesudo código:

```
Para (n = 1; n <=10; n)
{
        f = 1;
                Para (i = 2; i<=n; i)
                {
                        f *= i; }
                        Salida ("El factorial de ", n);
                        Salida (" es: ", f);
                }
```

```
        }
}
```

```
--While

DECLARE
 n NUMBER := 0;
 f NUMBER := 0;
 i NUMBER := 0;

BEGIN
  WHILE n <= 10
  LOOP
     WHILE i <= n
     LOOP
      f := f * i;
      i := i + 1;
      DBMS_OUTPUT.PUT_LINE( 'El factorial de' || n || 'es' || f );
     END LOOP;
    n := n + 1;
  END LOOP;
END;
```

```
--For

DECLARE
 n NUMBER := 0;
 f NUMBER := 0;
```

```
 i NUMBER := 0;

BEGIN
 FOR n IN 1..9
 LOOP
    FOR n IN (i-n)
    LOOP
     f := f * i;
     i := i + 1;
     DBMS_OUTPUT.PUT_LINE( 'El factorial de' || n || 'es' || f );
    END LOOP;
  n := n + 1;
 END LOOP;
END;
```

```
/*
#TEMA   A: Hacer un programa que me imprima y/o me muestre los números
múltiplos de tres, que están en el rango de uno hasta n, se asume que N se captura
por teclado. y al final cuando se termina la iteración me imprima un mensaje de
cuántos múltiplos de
tres hay en el rango de 1 hasta n.
*/

DECLARE
multi number := 10;
Cmul number := 0;
cont number := 1;

BEGIN
DBMS_OUTPUT.PUT_LINE('Multiplos: ');
WHILE cont < multi
LOOP
   IF MOD(cont, 3) = 0 THEN
      DBMS_OUTPUT.PUT_LINE(cont);
```

```
      Cmul := Cmul + 1;
   END IF;
   cont := cont + 1;
END LOOP;

DBMS_OUTPUT.PUT_LINE('Fueron ' || Cmul || ' multiplos');
END;
```

```
/*
#TEMA B: Hacer un programa que me imprima y/o me muestre los números
impares, que están en el rango de uno hasta n, se asume que N se captura por
teclado. y al final cuando se termina la iteración me imprima un mensaje de cuántos
números impares hay en el rango de 1 hasta n
*/

DECLARE
multi number := 10;
Cmul number := 0;
cont number := 1;

BEGIN
DBMS_OUTPUT.PUT_LINE('Numeros impares: ');
WHILE cont < multi
LOOP
   IF MOD(cont, 2) = 0 THEN
      DBMS_OUTPUT.PUT_LINE(cont);
      Cmul := Cmul + 1;
   END IF;
   cont := cont + 1;
END LOOP;
```

```
DBMS_OUTPUT.PUT_LINE('Fueron ' || Cmul || ' multiplos');
END;
```

REFERENCIAS

Sección #1

CURSOS DE PROGRAMACION https://www.javatpoint.com/

https://livesql.oracle.com/apex/f?p=590:1000

http://profesionghh.blogspot.com/2015/09/tipos-de-datos-plsql.html

https://serchlg.wordpress.com/2018/10/17/introduccion-oracle-plsql/

Sección #2

https://www.jpromero.com/2011/06/oracle-precedencia-de-operadores.html