ACTIVIDADES TEMA 6 PARTE I BLOQUES SOLUCIONES– PL/SQL

BLOQUES E INTRODUCCIÓN A SUBPROGRAMAS

CREAR UN USUARIO U6, CON LA MISMA CONTRASEÑA y lanzar dentro de este esquema el script TABLAS_U4.SQL para partir de una tabla sin duplicidades. Lanar el script TABLA_U4.SQL y

1. *Escribir un bloque PL/SQL que escriba el texto 'Hola'.*

```
BEGIN
    DBMS_OUTPUT.PUT_LINE('HOLA');
END;
```

2. *¿Qué hace el siguiente bloque PL/SQL?*

```
DECLARE
 V_num NUMBER;
BEGIN
 SELECT count(*) INTO v_num FROM EMPLE;
 DBMS_OUTPUT.PUT_LINE (v_num);
END;
```

Cuenta el número de filas que hay en la tabla productos, deposita el resultado en la variable v_num, y visualiza su contenido.

3. *Introducir el bloque anterior y guardarlo en un fichero llamado EJER6_3.SQL*

```
SAVE A:\PROG01.SQL REPLACE
```

4. *Ejecutar el bloque anterior y comprobar el resultado*

Desde el SQLPlus o desde el ARCHIVO DE COMANDOS

```
Start ejer5_3.sql
```

5. *Diseñar un pequeño bloque que nos muestre el departamento al que pertenece el empleado cuyo apellido es 'SALA', en la tabla EMPLE.*

```
DECLARE
   v_dept_no   emple.DEPT_NO%TYPE;
BEGIN
   SELECT dept_no INTO v_dept_no FROM emple where APELLIDO='SALA';
   DBMS_OUTPUT.PUT_LINE (v_dept_no);
END;
```

6. Diseñar un bloque de programa muestre el emp_no que le correspondería a un nuevo empleado, sabiendo que sería siguiente valor al mayor de todos los emp_no de la tabla EMPLE.

```
declare
  ult_emp_no   emple.emp_no%type;
  sgte_emp_no  emple.emp_no%type;
begin
 select max(emp_no) into ult_emp_no from emple;
 sgte_emp_no:= ult_emp_no+1;
 dbms_output.put_line('ultimo emp_no '|| ult_emp_no||'   emp_no del nuevo empleado:
'||sgte_emp_no);
end;
```

7. Bloque que calcule cual es el departamento cuyo salario medio es menor. Visualizar el dept_no, y su salario medio

```
declare
   v_dept_no              emple.dept_no%type;
   v_salario_medio        real;
begin
   select dept_no,avg(salario) INTO v_dept_no,v_salario_medio from emple
         group by dept_no
         having avg(salario)= (select min(avg(salario)) from emple group by dept_no);
   dbms_output.put_line('Código Departamento con salario medio menor'||
               v_dept_no||'Salario medio'||to_char(v_salario_medio,'999G999D99'));
end;
```

8. Modificar el bloque anterior para que además nos muestre el nombre del departamento

```
declare
   v_dept_no              emple.dept_no%type;
   v_salario_medio        real;
   v_nom_dep              depart.dnombre%type
begin
   select dept_no,avg(salario) INTO v_dept_no,v_salario_medio from emple
         group by dept_no
         having avg(salario)= (select min(avg(salario)) from emple group by dept_no);

   --sabiendo el dept_no ya puedo consultar en la tabla DEPART el nombre
   select dnombre into v_nom_dep from DEPART where dept_no = v_dept_no;
   dbms_output.put_line('Código Departamento con salario medio menor '||v_dept_no||
               ' Nombre Departamento '||v_nom_dep||
               ' Salario medio '||to_char(v_salario_medio,'999G999D99'));
end;
```

9. Diseñar un bloque PL/SQL que permita insertar una fila con la siguiente información:

- Emp_no: siguiente valor al mayor de todos los emp_no
- Apellido: se introducirá por teclado tu nombre
- Oficio: ANALISTA y se introducirá por teclado
- Dept_no y salario: se calcularán a partir del ejercicio anterior
- Jefe (DIR): emp_no del DIRECTOR de departamento donde se dará de alta
- Fecha de alta: la fecha actual
- El resto de los campos se dejan en blanco

```
accept nombre prompt 'Introduce tu nombre: ';
accept oficio prompt 'Introduce el oficio: ';

declare
  ult_emp_no         emple.emp_no%type;
  sgte_emp_no        emple.emp_no%type;
  v_dept_no               emple.dept_no%type;
  v_salario_medio    real;
  v_jefe             emple.dir%type;

begin
 select max(emp_no) into ult_emp_no from emple;
 sgte_emp_no:= ult_emp_no+1;
 select dept_no,avg(salario) INTO v_dept_no,v_salario_medio from emple
        group by dept_no
        having avg(salario)= (select min(avg(salario)) from emple
                                                    group by dept_no);
 select emp_no into v_jefe from emple where dept_no=v_dept_no and oficio='DIRECTOR';

 --insertamos en la tabla
 insert into emple (emp_no,apellido,oficio,dir,salario,dept_no,fecha_alt)
 values(sgte_emp_no,'&nombre','&oficio',v_jefe,v_salario_medio,v_dept_no,sysdate);

 commit;

 --visualizacion de datos
 dbms_output.put_line('ultimo emp_no existente'|| ult_emp_no||
                          '   emp_no del nuevo empleado: '||sgte_emp_no);
 dbms_output.put_line('Nombre: '||'&nombre');
 dbms_output.put_line('Oficio: '||'&oficio');
 dbms_output.put_line('Código Departamento con salario medio menor'|| v_dept_no);
 dbms_output.put_line('Salario medio de este departamento'||
                          to_char(v_salario_medio,'999G999D99'));
 dbms_output.put_line('El director de este nuevo empleado será : '||v_jefe);
end;
```

10. Crear una tabla llamada DATOS_EMPLE con la siguiente información:

- Emp_no clave primaria
- Apellido
- Sueldo
- Estrellas: donde por cada 500€ se le asignará una estrella al empleado

11. Crear un bloque PL/SQL que una vez introducir un número de empleado, localice dicho empleado en la tabla EMPLE e inserte en la tabla DATOS_EMPLE la información correspondiente al emp_no, apellido, sueldo. Utilizar un registro

```
accept cod prompt 'Introduce el codigo de empleado:';

DECLARE
   v_emp_no          emple.emp_no%type;
   v_apellido        emple.apellido%type ;
   v_sueldo          emple.salario%type ;
   v_estrellas       datos_emple.estrellas%type default null;
   cont              integer;

BEGIN
  select emple.emp_no,emple.apellido,(emple.salario+nvl(emple.comision,0))
         into v_emp_no, v_apellido,v_sueldo from emple
         where emple.emp_no=&cod;

  cont:=floor(v_sueldo/500);

  FOR i IN 1..cont LOOP
       v_estrellas := concat(v_estrellas,'*');
  END LOOP;

  dbms_output.put_line (v_emp_no||' ' || v_apellido||' ' ||v_sueldo||' '
                        ||v_estrellas);
   insert into datos_emple  values (v_emp_no, v_apellido,v_sueldo,v_estrellas);
   commit;
end;
```

12. Indicar los errores que aparecen en las siguientes instrucciones y la forma de corregirlos:

```
DECLARE
 num1 number(8,2):=0
 num2 number(8,2)  NOT NULL DEFAULT O;
 num3 NUMBER(8,2) NOT NULL;
 cantidad INTEGER(3);
 precio, descuento NUMBER(6);
 num4 num1%ROWTYPE;
 dto CONSTANT INTEGER;
BEGIN
 .....
END;
```

```
Num3 NUMBER(8,2) NOT NULL DEFAULT 0;  → debe ser el numero 0 no la letra O
Cantidad INTEGER;                     → los integer no tienen precisión
Precio NUMBER(6);                     → cada variable se declara de forma individual
Descuento NUMBER(6);
Num4 Num1%TYPE;                       → Num1 no es un registro de una TABLA, sino
                                        una variable definida como number
```

13. Escribir un procedimiento que reciba dos números y visualice su suma. Guardar el procedimiento en el catalogo de la base de datos para ser invocado en cualquier momento, es decir en la carpeta SUBPROGRAMAS creada anteriormente.

```
CREATE OR REPLACE PROCEDURE sumar_numeros (num1 NUMBER,num2 NUMBER)
IS
suma NUMBER(6);
BEGIN
   suma := num1 + num2;
   DBMS_OUTPUT.PUT_LINE('Suma: '|| suma);
END sumar_numeros;
```

14. Diseñar un bloque que permita invocar al procedimiento anterior pasándole dos números cualesquiera

```
Begin
   Sumar_numeros(4,5);
End;
```

15. Codificar un procedimiento que reciba una cadena y la visualice al revés, añadir el procedimiento al catalogo. A continuación realizar un bloque para comprobar que funciona

```
CREATE OR REPLACE PROCEDURE cadena_reves(
   vcadena VARCHAR2)
AS
   vcad_reves VARCHAR2(80);
BEGIN
   FOR i IN REVERSE 1..LENGTH(vcadena) LOOP
         vcad_reves := vcad_reves || SUBSTR(vcadena,i,1);
   END LOOP;
   DBMS_OUTPUT.PUT_LINE(vcad_reves);
END cadena_reves;

--
Begin
  Cadena_reves('HOla mundo');
End;
```

16. Escribir una función que reciba una fecha y devuelva, en número, solo el año correspondiente a esa fecha. Añadirla al catálogo.

```
//function que devuelve el año de una fecha que se pasa como parámetro
CREATE OR REPLACE FUNCTION f_anio      (fecha DATE)   RETURN NUMBER
AS
   v_anio NUMBER(4);
BEGIN
   v_anio := TO_NUMBER(TO_CHAR(fecha, 'YYYY'));
   RETURN v_anio;
END f_anio;
```

17. Escribir un bloque PL/SQL que haga uso de la función anterior. Por ejemplo, a partir de la fecha de hoy, que devuelve el año y lo visualice.

```
DECLARE
      anio        NUMBER(4);     -- variable donde se recogerá el año
      fecha       date;
 BEGIN
      fecha:= SYSDATE;           -- como prueba introduzco la fecha del sistema actual
      anio := f_anio(fecha);     --invoco a la función f_anio pasándole la fecha
      DBMS_OUTPUT.PUT_LINE('En la fecha: '|| fecha || ' el año es : '|| anio);
 END;
```