CENTRE ESPECÍFIC
D´EDUCACIÓ A DISTÀNCIA DE
LA COMUNITAT VALENCIANA

TEMA8
PL/SQL. CURSORES, DISPARADORES Y EXCEPCIONES
EJERCICIOS

Bases de Datos
CFGS DAW

Raquel Torres
raquel.torres@ceedcv.es
Versión:180427.1357

UD08. PL/SQL.CURSORES Y DISPARADORES

1. EJERCICIOS

Para la resolución de estos ejercicios se recomienda consultar el Anexo de Funciones predefinidas, así como el resto de funciones definidas en los manuales online.

Para todos los ejercicios se utilizará la bbdd de ejercicios de este tema BD_ej_PL_SQL, excepto para el Ejercicio 8, donde utilizaremos la bbdd de jardineria_oracle.zip.

1.1 Ejercicio 1

Escribe un cursor que muestre todos los empleados de la tabla con un bucle *WHILE*.

1.2 Ejercicio 2

Escribe un cursor que muestre el nombre y salario de los empleados del departamento 10 utilizando una variable registro.

1.3 Ejercicio 3

Escribe un cursor que muestre los años de antigüedad de los empleados del departamento 10.

(En esta solución se ha empleado la función MONTHS_BETWEEN, pero podríais haber utilizado también otras, como DATEDIFF(f1,f2)).

1.4 Ejercicio 4

Supongamos que queremos hacer un trigger que no permita a un empleado ganar más de 5000€ si no es el presidente.

Para ello has de crear un trigger que antes de que se inserte o se actualice algún dato de la tabla *empleado* se compruebe si el salario es mayor de 5000 y si su tarea es diferente de *presidente* se imprima por pantalla (dado que aún no hemos visto excepciones) que 'No puede ganar tanto si no es el presidente'.

Hacer el trigger y comprobar que funciona correctamente.

1.5 Ejercicio 5

Crea un disparador de fila para impedir que se modifique el *nombre*, el *num_emp* o el *salario* si su nuevo salario es más de un 10% mayor que el anterior. En el caso que se vaya a modificar cualquiera

de estos 3 campos y el nuevo salario sea mayor que el 10% del anterior se imprimirá por pantalla que eso no está permitido (dado que aún no hemos visto excepciones).

1.6 Ejercicio 6

Crea un disparador de sentencia para seguir manteniendo información de los empleados que dejan de trabajar en la empresa, para ello necesitamos un disparador que almacene los empleados borrados en una tabla nueva.

Por tanto antes de crear el trigger, creamos la nueva tabla como copia de empleados y borramos su contenido para dejarla vacía.

```
CREATE TABLE emple_borrados as (select * from empleado);
TRUNCATE TABLE emple_borrados;
```

A continuación añadimos la columna fecha_baja que almacenerá la fecha en la que se produce la baja.

```
ALTER TABLE EMPLE_BORRADOS ADD FECHA_BAJA DATE;
```

1.7 Ejercicio 7

Supongamos que queremos hacer un trigger que impida insertar datos en la tabla de departamentos (*dpto*) fuera del horario normal de oficina.

El horario normal de oficina es de lunes a viernes de 8 a 15:00h.

Dado que todavía no hemos visto las excepciones, cuando se intente insertar un departamento fuera de dicho horario se deberá imprimir por pantalla la fecha del sistema en la que se produjo esa inserción no permitida.

1.8 Ejercicio 8

Para terminar, vamos a realizar un ejercicio, que aunque no está completo, pues no incluye el tratamiento de las excepciones, nos puede dar una idea de la potencia que aporta la programación con PL/SQL.

Se trata de realizar un procedimiento que reciba el código de un cliente y nos muestra su estado, es decir, sus datos de cliente, los pedidos que ha realizado dicho cliente con el importe total, los pagos que ha realizado y el importe que tiene pendiente, así como si ha superado o no el crédito que tiene indicado en su ficha de cliente.

Una vez creado el procedimiento lo podremos ejecutar en la línea de comandos con el comando

EXECUTE seguido del nombre del procedimiento y los parámetros entre paréntesis.

El resultado deberá ser:

```
SQL> @ c:\src\ejer_pl_01.sql

Procedure created.

SQL> EXECUTE ESTADOCLIENTE(1);
CODIGO CLIENTE:  1
NOMBRE CLIENTE:  DGPRODUCTIONS GARDEN
CONTACTO:        Daniel G
TELEFONO:        5556901745
FAX:             5556901746
DIRECCION:       False Street 52 2 A
CIUDAD:          San Francisco
REGION:
PAIS:            USA
COD. POSTAL:     24006
-----------------------------------------
Pedido Num.:8 Importe: 1065
Pedido Num.:9 Importe: 2535
Pedido Num.:11 Importe: 820
Pedido Num.:12 Importe: 290
Pedido Num.:25 Importe: 1455
Total Facturado: 6165
-----------------------------------------
Pago 1 FECHA: 10/11/08 Cantidad: 2000
Pago 2 FECHA: 10/12/08 Cantidad: 2000
Total Pagado: 4000
-----------------------------------------
Su saldo es: -2165
NO ha superado su CREDITO que es de 3000 Euros

PL/SQL procedure successfully completed.
```

(Pista: Crear dos cursores (*Pedidos_Cliente* y *Pagos_Cliente*) y utilizar FOR de cursor para procesar los resultados)