-- Procedimientos Almacenados

-- SubPrograma Almacenado / Acepta Parametros / No Retorna Valores por Defecto

-- se puede ejecutar con la instruccion EXECUTE o en el interior de otro

-- procedimiento o bloque anonimo

create or replace procedure sp\_listado(p\_depto in number,p\_total out number)

is

v\_cant number:=0;

begin

for x in (select \* from employees where department\_id= p\_depto)

loop

dbms\_output.put\_line('Nombre:' || x.first\_name);

v\_cant:=v\_cant+1;

end loop;

p\_total:=v\_cant;

end;

--------------------------------------------

set serveroutput on;

variable cantidad number;

execute sp\_listado(130, :cantidad);

begin

dbms\_output.put\_line(:cantidad);

end;

--------------------------------------------

declare

v\_cantidad\_depto number;

begin

sp\_listado(30, v\_cantidad\_depto);

dbms\_output.put\_line('Cantidad de Deptos:' || v\_cantidad\_depto);

end;

-------------------------------------------------------------------------

-- Funciones

-- Sub Programas Almacenados / Retornan datos por naturaleza / pueden

-- recibir datos como parametros / Al finalizar deben retornar RETURN

create or replace function fn\_cantidad\_empleados(p\_depto in number)

return number

is

v\_cantidad number;

begin

select count(\*) into v\_cantidad

from employees where department\_id = p\_depto;

return v\_cantidad;

end;

----------------------------------------

-- para ejecutarla se llama desde un bloque anonimo / otra funcion / desde

-- un procedimiento almacenado / desde una instruccion SELECT

declare

v\_cantidad number;

begin

v\_cantidad:= fn\_cantidad\_empleados(30);

dbms\_output.put\_line('Cantidad de Empleados:' || v\_cantidad);

end;

select department\_name, fn\_cantidad\_empleados(department\_id)

from departments;

----------------------------------------------------------------------

-- Paquetes

-- Permiten organizar procedimientos almacenados, funciones, variables, constantes

-- etc (bloques de codigo) en base aun nombre en comun

-- Poseen una cabecera y cuerpo

-- Todo lo declarado en cabecera sera de uso publico

-- Todo los declarado en el cuerpo sera privado a menos que se encuentre

-- definido en cabecera

create or replace package pkg\_hr

is

function fn\_cantidad\_empleados(p\_depto in number) return number;

procedure sp\_listado(p\_depto in number,p\_total out number);

c\_iva CONSTANT number:=0.19;

end;

create or replace package body pkg\_hr

is

v\_datos varchar2(20):='hola mundo';

function fn\_cantidad\_empleados(p\_depto in number)

return number

is

v\_cantidad number;

begin

select count(\*) into v\_cantidad

from employees where department\_id = p\_depto;

return v\_cantidad;

end;

procedure sp\_listado(p\_depto in number,p\_total out number)

is

v\_cant number:=0;

begin

for x in (select \* from employees where department\_id= p\_depto)

loop

dbms\_output.put\_line('Nombre:' || x.first\_name);

v\_cant:=v\_cant+1;

end loop;

p\_total:=v\_cant;

end;

function fn\_hola return varchar2

is

begin

return 'hola mundo';

end;

end;

select department\_name, pkg\_hr.fn\_cantidad\_empleados(department\_id)

from departments;

declare

v\_cantidad number;

begin

pkg\_hr.sp\_listado(60, v\_cantidad);

dbms\_output.put\_line('Cantidad de Empleados:' || v\_cantidad);

end;

-----------------------------------------------------------------------

-- Triggers (disparadores)

-- Efectuar un proceso cuando se realice una accion

-- especifica (Delete, Update, Insert)

-- sobre una tabla

create table productos(

idproducto number primary key,

nombre varchar2(45),

stock number

);

insert into productos values(1,'Platano',500);

insert into productos values(2,'Pera', 40);

insert into productos values(3,'Melon',10);

select \* from productos;

create table ventas(

idBoleta number primary key,

idProducto number,

cantidad number

);

alter table ventas add constraint fk\_vent\_pro

foreign key (idProducto) references productos(idProducto);

-- cada vez que se realice una venta (Insertar) se debe

-- actualizar el stock de producto

create or replace trigger trg\_ventas

after insert or delete or update on ventas

for each row

declare

begin

if inserting then

update productos set stock=stock - :new.cantidad

where idproducto = :new.idproducto;

end if;

if deleting then

update productos set stock=stock + :old.cantidad

where idproducto = :old.idproducto;

end if;

if updating then

if :old.cantidad < :new.cantidad then

update productos set stock = stock - (:new.cantidad-:old.cantidad)

where idproducto= :old.idproducto;

else

update productos set stock = stock + (:old.cantidad-:new.cantidad)

where idproducto= :old.idproducto;

end if;

end if;

end;

insert into ventas values(1,3,2);

update ventas set cantidad=cantidad-3 where idboleta=1;

select \* from productos;

insert into ventas values(2,1,100);

delete from ventas where idboleta=1;

delete from ventas where idboleta=2;