--Creación de Tablas

DROP TABLE gasto;

DROP TABLE empleo;

DROP TABLE cliente;

CREATE TABLE cliente( ced NUMBER(8) PRIMARY KEY,

nom VARCHAR2(10) NOT NULL);

INSERT INTO cliente (ced, nom) values (1, 'Daniel');

INSERT INTO cliente (ced, nom) values (2, 'Santa');

CREATE TABLE empleo( ced NUMBER(8) REFERENCES cliente,

nit\_empresa INTEGER,

valor\_mensual NUMBER(6) NOT NULL,

PRIMARY KEY (ced, nit\_empresa));

INSERT INTO empleo (ced, nit\_empresa, valor\_mensual) values (1, 1, 400);

INSERT INTO empleo (ced, nit\_empresa, valor\_mensual) values (2, 2, 200);

INSERT INTO empleo (ced, nit\_empresa, valor\_mensual) values (1, 2, 100);

INSERT INTO empleo (ced, nit\_empresa, valor\_mensual) values (2, 1, 700);

CREATE TABLE gasto( cod\_gasto NUMBER(8) PRIMARY KEY,

ced NUMBER(8) REFERENCES cliente,

valor\_mensual NUMBER(6),

desc\_gasto VARCHAR2(10));

INSERT INTO gasto (cod\_gasto, ced, valor\_mensual, desc\_gasto) values (1, 1, 400, 'Hola');

INSERT INTO gasto (cod\_gasto, ced, valor\_mensual, desc\_gasto) values (2, 1, 400, 'Hola');

INSERT INTO gasto (cod\_gasto, ced, valor\_mensual, desc\_gasto) values (3, 2, 400, 'Hola');

INSERT INTO gasto (cod\_gasto, ced, valor\_mensual, desc\_gasto) values (4, 2, 400, 'Hola');

INSERT INTO gasto (cod\_gasto, ced, valor\_mensual, desc\_gasto) values (5, 1, 200, 'Hola');

--2

--Función que retorna el cliente que más gana

CREATE OR REPLACE FUNCTION ganaMasFuncion

RETURN NUMBER IS

cedulaMas empleo.ced%TYPE;

mayorValor empleo.valor\_mensual%TYPE;

BEGIN

mayorValor := 0;

FOR empleo IN (SELECT ced as cedula, SUM(valor\_mensual) as totalSueldo

FROM empleo GROUP BY ced)

LOOP

IF empleo.totalSueldo > mayorValor THEN

cedulaMas := empleo.cedula;

mayorValor := empleo.totalSueldo;

END IF;

END LOOP;

DBMS\_OUTPUT.PUT\_LINE('La cédula del cliente que más gana es: ' || cedulaMas);

RETURN cedulaMas;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('No hay empleos');

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('Error inesperado: ' || SQLERRM);

END;

/

--3

--Función que retorna el cliente que más gasta

CREATE OR REPLACE FUNCTION gastaMasFuncion

RETURN NUMBER IS

cedulaMas gasto.ced%TYPE;

mayorValor gasto.valor\_mensual%TYPE;

BEGIN

mayorValor := 0;

FOR gasto IN (SELECT ced as cedula, SUM(valor\_mensual) as totalGasto

FROM gasto GROUP BY ced)

LOOP

IF gasto.totalGasto > mayorValor THEN

cedulaMas := gasto.cedula;

mayorValor := gasto.totalGasto;

END IF;

END LOOP;

DBMS\_OUTPUT.PUT\_LINE('La cédula del cliente que más gasta es: ' || cedulaMas || '. Ya que gasta: ' || mayorValor);

RETURN cedulaMas;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('No hay gastos');

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('Error inesperado: ' || SQLERRM);

END;

/

--Creación del paquete del paquete

CREATE OR REPLACE PACKAGE paquete1 IS

FUNCTION ganaMasFuncion RETURN NUMBER;

FUNCTION gastaMasFuncion RETURN NUMBER;

PROCEDURE procedimiento1(

cedulaIng IN cliente.ced%TYPE,

nombreIng IN cliente.nom%TYPE,

nitEmpresaIng IN empleo.nit\_empresa%TYPE,

valorMensualIng IN empleo.valor\_mensual%TYPE);

PROCEDURE ganaMas(cedulaMas OUT cliente.ced%TYPE);

END;

/

CREATE OR REPLACE PACKAGE BODY paquete1 IS

--Subprograma privado - Procedimiento que modifica el nombre del Cliente

PROCEDURE existeCliente(

cedulaIng IN cliente.ced%TYPE,

nombreIng IN cliente.nom%TYPE,

esta OUT BOOLEAN)

IS

cedulaEncontrada cliente.ced%TYPE;

BEGIN

esta := FALSE;

SELECT ced INTO cedulaEncontrada

FROM cliente WHERE ced = cedulaIng;

UPDATE cliente SET nom = nombreIng WHERE ced = cedulaEncontrada;

DBMS\_OUTPUT.PUT\_LINE('El nombre del cliente ha sido actualizado correctamente');

esta := TRUE;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('Cliente no encontrado, se creará un cliente nuevo');

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('Error inesperado: ' || SQLERRM);

END;

--Subprograma privado - Procedimiento que modifica el valor mensual que se gana en el empleo

PROCEDURE existeEmpleo(

cedulaIng IN cliente.ced%TYPE,

nitEmpresaIng IN empleo.nit\_empresa%TYPE,

valorMensualIng IN empleo.valor\_mensual%TYPE,

esta OUT BOOLEAN)

IS

cedulaEncontrada cliente.ced%TYPE;

BEGIN

esta := FALSE;

SELECT ced INTO cedulaEncontrada

FROM empleo WHERE ced = cedulaIng AND nit\_empresa = nitEmpresaIng;

UPDATE empleo SET valor\_mensual = valorMensualIng WHERE ced = cedulaEncontrada AND nit\_empresa = nitEmpresaIng;

esta := TRUE;

DBMS\_OUTPUT.PUT\_LINE('El valor mensual del empleo ha sido actualizado correctamente');

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('Empleo no encontrado, se creará un empleo nuevo');

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('Error inesperado: ' || SQLERRM);

END;

--Subprograma Público - Procedimiento que crea un nuevo cliente y un nuevo empleo

PROCEDURE procedimiento1(

cedulaIng IN cliente.ced%TYPE,

nombreIng IN cliente.nom%TYPE,

nitEmpresaIng IN empleo.nit\_empresa%TYPE,

valorMensualIng IN empleo.valor\_mensual%TYPE)

IS

nombre cliente.nom%TYPE;

estaCliente BOOLEAN;

estaEmpleo BOOLEAN;

nombreLargo EXCEPTION;

BEGIN

IF (LENGTH(nombreIng) > 10) THEN

RAISE nombreLargo;

nombre := SUBSTR(nombreIng, 1, 10);

DBMS\_OUTPUT.PUT\_LINE('El nombre del cliente es demasiado largo, se tomarán los 10 primeros caracteres que son: ' || nombre);

ELSE

nombre := nombreIng;

END IF;

existeCliente(cedulaIng, nombre ,estaCliente);

existeEmpleo(cedulaIng, nitEmpresaIng, valorMensualIng, estaEmpleo);

IF NOT estaCliente THEN

INSERT INTO cliente (ced, nom) values (cedulaIng, nombre);

END IF;

IF NOT estaEmpleo THEN

INSERT INTO empleo (ced, nit\_empresa, valor\_mensual) values (cedulaIng, nitEmpresaIng, valorMensualIng);

END IF;

EXCEPTION

WHEN nombreLargo THEN

DBMS\_OUTPUT.PUT\_LINE('El nombre es demasiado largo, entonces se rechaza el usuario');

END;

--Subprograma Público - Procedimiento que retorna el cliente que más gana

PROCEDURE ganaMas(cedulaMas OUT cliente.ced%TYPE)

IS

mayorValor empleo.valor\_mensual%TYPE;

BEGIN

mayorValor := 0;

FOR empleo IN (SELECT ced as cedula, SUM(valor\_mensual) as totalSueldo

FROM empleo GROUP BY ced)

LOOP

IF empleo.totalSueldo > mayorValor THEN

cedulaMas := empleo.cedula;

mayorValor := empleo.totalSueldo;

END IF;

END LOOP;

DBMS\_OUTPUT.PUT\_LINE('La cédula del cliente que más gana es: ' || cedulaMas);

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('No hay empleos');

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('Error inesperado: ' || SQLERRM);

END;

--Subprograma Público - Función que retorna el cliente que más gana

FUNCTION ganaMasFuncion

RETURN NUMBER IS

cedulaMas empleo.ced%TYPE;

mayorValor empleo.valor\_mensual%TYPE;

BEGIN

mayorValor := 0;

FOR empleo IN (SELECT ced as cedula, SUM(valor\_mensual) as totalSueldo

FROM empleo GROUP BY ced)

LOOP

IF empleo.totalSueldo > mayorValor THEN

cedulaMas := empleo.cedula;

mayorValor := empleo.totalSueldo;

END IF;

END LOOP;

DBMS\_OUTPUT.PUT\_LINE('La cédula del cliente que más gana es: ' || cedulaMas);

RETURN cedulaMas;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('No hay empleos');

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('Error inesperado: ' || SQLERRM);

END;

--Subprograma Público - Función que retorna el cliente que más gasta

FUNCTION gastaMasFuncion

RETURN NUMBER IS

cedulaMas gasto.ced%TYPE;

mayorValor gasto.valor\_mensual%TYPE;

BEGIN

mayorValor := 0;

FOR gasto IN (SELECT ced as cedula, SUM(valor\_mensual) as totalGasto

FROM gasto GROUP BY ced)

LOOP

IF gasto.totalGasto > mayorValor THEN

cedulaMas := gasto.cedula;

mayorValor := gasto.totalGasto;

END IF;

END LOOP;

DBMS\_OUTPUT.PUT\_LINE('La cédula del cliente que más gasta es: ' || cedulaMas || '. Ya que gasta: ' || mayorValor);

RETURN cedulaMas;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('No hay gastos');

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('Error inesperado: ' || SQLERRM);

END;

END; --Fin del paquete

/

DECLARE

cedula NUMBER(8);

BEGIN

cedula := paquete1.ganaMasFuncion;

cedula := paquete1.gastaMasFuncion;

paquete1.procedimiento1(3, 'Daniel Santa', 1, 200);

paquete1.ganaMas(cedula);

END;

/

SELECT \* FROM cliente;

SELECT \* FROM empleo;

SELECT \* FROM gasto;