

Universidad Tecnológica de Panamá

Facultad de Ingeniería de Sistemas Computacionales

Licenciatura en Ingeniería de Sistemas de Información

Departamento de Sistemas de Información, Control y Evaluación de Recursos Informáticos

Sistemas de Bases de Datos II

Laboratorio No.7. Fundamentos del Lenguaje-Cursores-Procedimientos y Funciones y Triggers

Facilitador:

Ing. Henry Lezcano

Integrantes:

David Cedeño 8-951-990

Simon Billeke E-8-154322

Gabriel Grimaldo 8-949-56

Daniel Rodriguez 3-740-2242

Grupo:

1IF131

II Semestre, 2020

1. Agregar una tabla al modelo físico que almacenes las sucursales de la empresa

la financiera con las restricciones correspondientes:

Sucursal (cod\_surcursal, nombresucursal, tipoprestamo,

montoprestam os, tipoaho de ahorro, montoahorrado) usted presente la mejor opción.

* Create Table Sucursal\_Ahorro(

tipoaho number not null,

cod\_sucursal varchar (10) not null,

montoahorrado number not null,

constraint fk\_Suc\_Aho\_tipoA foreign key (tipoaho) References Tipo\_Ahorro (ID\_TipoA),

constraint fk\_Suc\_Aho\_codS foreign key (cod\_sucursal) References Sucursal (cod\_sucursal),

constraint pk\_Suc\_Aho primary key (tipoaho,cod\_sucursal));

1. Agregar la tabla tipos de ahorros donde los tipo cuyos atributos son tipo de ahorro, descripción, y tasa de interés, donde los aprobados fueron:
   1. 01 ahorro de navidad con tasa de interés det 6%.
   2. 02 ahorro corriente con tasa de interés del 4%.
   3. 03 ahorro de escolar con tasa de interés del 6%.

* Create Table Tipo\_Ahorro(

ID\_TipoA number not null,

Nombre varchar (20) not null,

Tasa\_inte number not null,

constraint pk\_Tipo\_Ahorro\_ID primary key (ID\_TipoA));

1. Agregar la tabla de ahorros al modelo de base de datos cuyos atributos serán:
   1. Código de sucursal
   2. Id del cliente
   3. Tipo de ahorro
   4. Número de cuenta(secuencia)
   5. Fecha de apertura
   6. Tasa de interés
   7. Letra Depósito mensual(acuerdo como será incrementada la cuenta)
   8. Saldo del ahorro
   9. Saldo del interés
   10. Fecha de deposito
   11. Fecha de retiro
   12. Usuario
   13. Fecha de modificación

Deberán incluir las restricciones aplicadas al modelo, por otra parte los clientes pueden tener varias cuentas de un mismo tipo.

Create Table Ahorros(

Ncuenta number not null,

cod\_sucursal varchar(5) not null,

Id\_cliente number not null,

tipoaho number not null,

Fecha\_aper date not null,

T\_interes number not null,

LetraDM number not null,

SAhorro number not null,

Sinteres number not null,

Fecha\_dep date null,

Fecha\_ret date null,

Usuario varchar(20) not null,

Fecha\_mod date not null,

constraint pk\_ahorro primary key (Ncuenta),

constraint fk\_ahoro\_suc foreign key (cod\_sucursal) References Sucursal (cod\_sucursal),

constraint fk\_ahoro\_tip foreign key (tipoaho) References Tipo\_Ahorro (ID\_TipoA),

constraint fk\_ahoro\_cli foreign key (Id\_cliente) References Clientes (IDCliente));

1. Agregar una tabla de auditoria que llevara la trazabilidad transaccional de cuentas de ahorros por el momento donde sus atributos serán:
   1. Id\_transaccion(secuencia)
   2. Tabla
   3. Id cliente
   4. Tipo de ahorro
   5. Tipo transacción
   6. Saldo de la cuenta anterior antes de la aplicación
   7. Monto del deposito o retiro
   8. Saldo final después de la aplicación
   9. Usuario
   10. Fecha (diamesaño, hora minuto y segundo).

* Create Table Tipo\_Transaccion(

Id\_tipoT Number not null,

Nombre varchar(15) not null,

constraint pk\_Tipo\_transa primary key(Id\_tipoT));

* Create Table Auditoria(

Id\_transaccion number not null,

Tabla varchar(30),

Id\_cliente number not null,

tipoaho number not null,

tipotransac number DEFAULT 1,

Santerior number DEFAULT 0,

Monto number DEFAULT 0,

Sfinal number DEFAULT 0,

Usuario varchar(20) not null,

Fecha date not null,

constraint pk\_trazabi primary key (Id\_transaccion),

constraint fk\_traza\_tipA foreign key (tipoaho) References Tipo\_Ahorro (ID\_TipoA),

constraint fk\_traza\_cli foreign key (Id\_cliente) References Clientes (IDCliente),

constraint fk\_traza\_tipT foreign key (tipotransac) References Tipo\_Transaccion(Id\_tipoT));

* CREATE SEQUENCE SAditroia

START WITH 1

INCREMENT BY 1;

1. Agregar una tabla transaccional para recibir los depósitos y retiros de los clientes a sus cuentas de ahorros con las restricciones correspondiente que contendrá la siguiente atributos:

Transadeporeti(cod\_sucursal, id\_transaccion, id\_cliente, tipoahorro, fechatransaccion, tipotransac( 1=deposito, 2=retiro), monto del deposito o retiro, fechainsercion(dd/mm/yyyy, hh:mm:ss, usuario).

* Create Table Tipo\_Transaccion(

Id\_tipoT Number not null,

Nombre varchar(15) not null,

constraint pk\_Tipo\_transa primary key(Id\_tipoT));

* Create Table Transadeporeti(

cod\_sucursal varchar(5) not null,

Id\_transaccion number not null,

Id\_cliente number not null,

tipoaho number not null,

Ncuenta number not null,

fechatransaccion date not null,

tipotransac number not null,

monto number not null,

fechainsercion date not null,

usuario varchar(20) not null,

constraint pk\_Transdeporeti primary key(Id\_transaccion),

constraint fk\_deporeti\_cli foreign key (Id\_cliente) References Clientes (IDCliente),

constraint fk\_deporeti\_tipT foreign key (tipotransac) References Tipo\_Transaccion(Id\_tipoT),

constraint fk\_deporeti\_Ncu foreign key (Ncuenta) References Ahorros(Ncuenta),

constraint fk\_deporeti\_tipA foreign key (tipoaho) References Tipo\_Ahorro (ID\_TipoA),

constraint fk\_deporeti\_suc foreign key (cod\_sucursal) References Sucursal (cod\_sucursal));

1. Procedimiento almacenado para la carga o inserción de la tabla paramétrica
   1. Tipo de ahorros

* CREATE OR REPLACE PROCEDURE Crear\_Tipo\_Aho(

p\_ID\_TipoA Tipo\_Ahorro.ID\_TipoA%TYPE,

p\_Nombre Tipo\_Ahorro.Nombre%TYPE,

p\_Tasa\_inte Tipo\_Ahorro.Tasa\_inte%TYPE,

p\_error out varchar2)AS

BEGIN

INSERT INTO Tipo\_Ahorro(ID\_TipoA,Nombre,Tasa\_inte)

VALUES(p\_ID\_TipoA,p\_Nombre,p\_Tasa\_inte);

p\_error := 'Registro Creado';

EXCEPTION

WHEN DUP\_VAL\_ON\_INDEX THEN

p\_error := 'Tipo de ahorro existente';

WHEN OTHERS THEN

p\_error := 'No se creo el registro';

COMMIT;

END Crear\_Tipo\_Aho;

* DECLARE

v\_error varchar2(100);

BEGIN

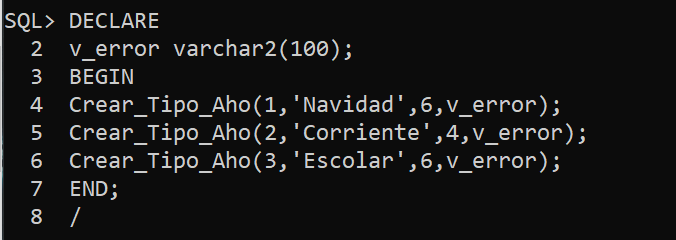
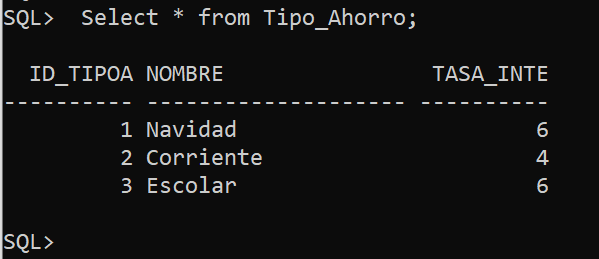
Crear\_Tipo\_Aho(1,'Navidad',6,v\_error);

Crear\_Tipo\_Aho(2,'Corriente',4,v\_error);

Crear\_Tipo\_Aho(3,'Escolar',6,v\_error);

dbms\_output.put\_line(v\_error);

END;

* 
* 

1. Procedimiento almacenado para la apertura o inserción de los ahorros aprobados con toda la información correspondiente.

* CREATE SEQUENCE N\_Cuenta

START WITH 1

INCREMENT BY 1;

* CREATE OR REPLACE PROCEDURE Crear\_Ahorros(

p\_codSuc Ahorros.cod\_sucursal%TYPE,

p\_Idcli Ahorros.Id\_cliente%TYPE,

p\_tipoaho Ahorros.tipoaho%TYPE,

p\_Tint Ahorros.T\_interes%TYPE,

p\_LetraDM Ahorros.LetraDM%TYPE,

p\_SAhorro Ahorros.SAhorro%TYPE,

p\_Sinteres Ahorros.Sinteres%TYPE,

p\_error out varchar2)AS

BEGIN

INSERT INTO Ahorros (Ncuenta,cod\_sucursal,id\_cliente,tipoaho,fecha\_aper,t\_interes,letradm,sahorro,sinteres,usuario,fecha\_mod)

VALUES(N\_Cuenta.NEXTVAL,p\_codSuc,p\_Idcli,p\_tipoaho,SYSDATE,p\_Tint,p\_LetraDM,p\_SAhorro,p\_Sinteres,user,sysdate);

UPDATE Sucursal\_Ahorro

SET montoahorrado = montoahorrado +p\_SAhorro

Where tipoaho=p\_tipoaho AND cod\_sucursal=p\_codSuc;

IF SQL%ROWCOUNT=0 THEN

INSERT INTO Sucursal\_Ahorro(tipoaho,cod\_sucursal,montoahorrado)

VALUES(p\_tipoaho,p\_codSuc,p\_SAhorro);

END IF;

p\_error := 'Registro Creado';

EXCEPTION

WHEN DUP\_VAL\_ON\_INDEX THEN

p\_error := 'Ahorro existente';

WHEN OTHERS THEN

p\_error := 'No se creo el registro';

COMMIT;

END Crear\_Ahorros;

* DECLARE

v\_error varchar2(100);

BEGIN

Crear\_Ahorros('S-BDG',1,1,6,100,0,0,v\_error);

Crear\_Ahorros('S-VL',2,2,4,50,0,0,v\_error);

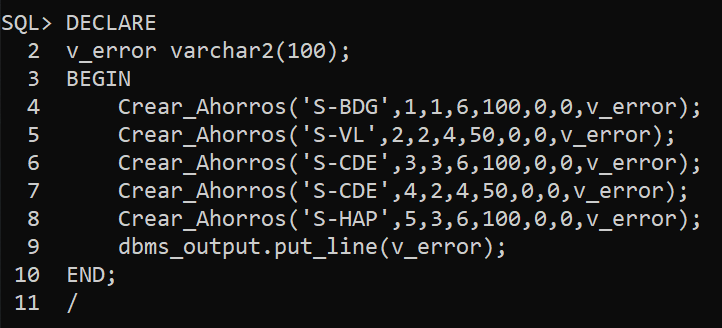
Crear\_Ahorros('S-CDE',3,3,6,100,0,0,v\_error);

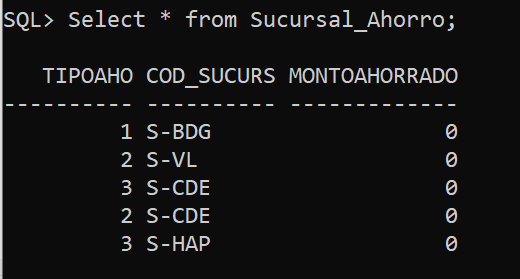
Crear\_Ahorros('S-CDE',4,2,4,50,0,0,v\_error);

Crear\_Ahorros('S-HAP',5,3,6,100,0,0,v\_error);

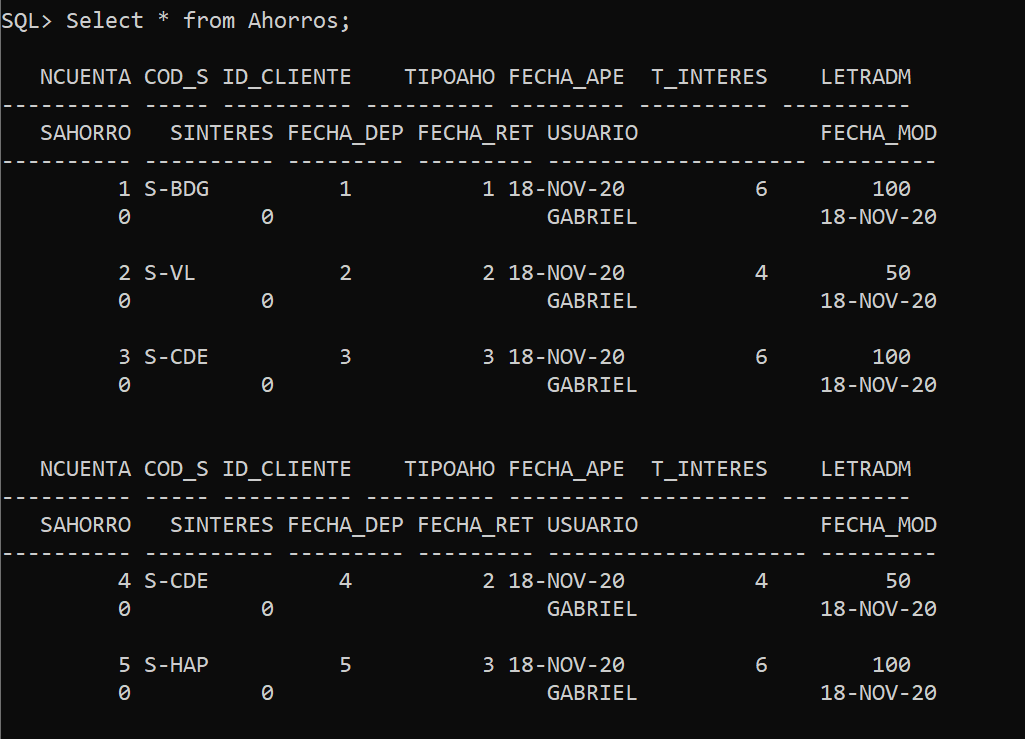
dbms\_output.put\_line(v\_error);

END;

* 
* Tabla sucursal ahorro



* Tabla ahorros



1. Procedimiento almacenado para la carga o inserción de los depósitos o retiros recibidos de los clientes para sus cuentas de ahorros que se almacena en la tabla transadeporeti. Por lo menos uno para cada tipo de ahorro.

* CREATE SEQUENCE ID\_Transaccion

START WITH 1

INCREMENT BY 1;

* CREATE OR REPLACE PROCEDURE Crear\_TipoTrans(

p\_idtipo tipo\_transaccion.id\_tipot%type,

p\_Nombre tipo\_transaccion.Nombre%type,

p\_error out varchar2)AS

BEGIN

INSERT INTO tipo\_transaccion(id\_tipot,Nombre)

Values (p\_idtipo,p\_Nombre);

p\_error := 'Registro Creado';

EXCEPTION

WHEN DUP\_VAL\_ON\_INDEX THEN

p\_error := 'Ahorro existente';

WHEN OTHERS THEN

p\_error := 'No se creo el registro';

COMMIT;

END Crear\_TipoTrans;

/

* DECLARE

v\_error varchar2(100);

BEGIN

Crear\_TipoTrans(1,'Deposito',v\_error);

Crear\_TipoTrans(2,'Retiro',v\_error);

dbms\_output.put\_line(v\_error);

END;

/

* CREATE OR REPLACE PROCEDURE Crear\_Transadeporeti(

p\_codsuc Transadeporeti.cod\_sucursal%TYPE,

p\_Idcli Transadeporeti.Id\_cliente%TYPE,

p\_tipoaho Transadeporeti.tipoaho%TYPE,

p\_ttransac Transadeporeti.tipotransac%TYPE,

p\_monto Transadeporeti.monto%TYPE,

p\_Ncuenta Transadeporeti.Ncuenta%TYPE,

p\_error out varchar2)AS

BEGIN

INSERT INTO Transadeporeti(cod\_sucursal,Id\_transaccion,Id\_cliente,tipoaho,fechatransaccion,tipotransac,monto,fechainsercion,usuario,Ncuenta)

VALUES(p\_codsuc,ID\_Transaccion.NEXTVAL,p\_Idcli,p\_tipoaho,sysdate,p\_ttransac,p\_monto,sysdate,user,p\_Ncuenta);

p\_error := 'Registro Creado';

EXCEPTION

WHEN DUP\_VAL\_ON\_INDEX THEN

p\_error := 'Transaccion existente';

WHEN OTHERS THEN

p\_error := 'No se creo el registro';

COMMIT;

END Crear\_Transadeporeti;

* DECLARE

v\_error varchar2(100);

BEGIN

Crear\_Transadeporeti('S-BDG',1,1,1,1000,1,v\_error);

Crear\_Transadeporeti('S-VL',2,2,1,500,2,v\_error);

Crear\_Transadeporeti('S-VL',2,2,2,50,2,v\_error);

Crear\_Transadeporeti('S-CDE',3,3,1,1000,3,v\_error);

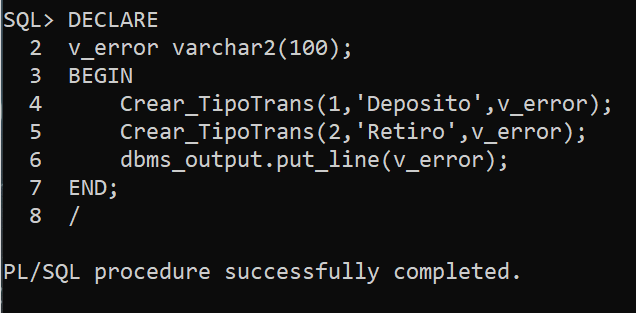
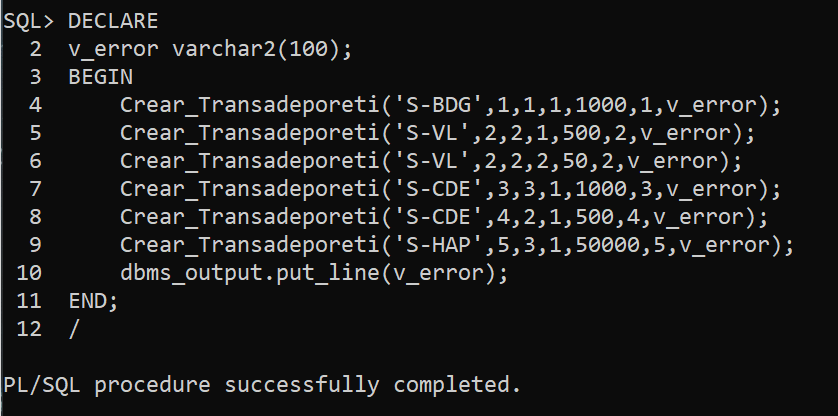
Crear\_Transadeporeti('S-CDE',4,2,1,500,4,v\_error);

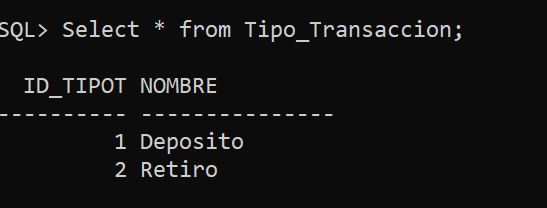
Crear\_Transadeporeti('S-HAP',5,3,1,50000,5,v\_error);

dbms\_output.put\_line(v\_error);

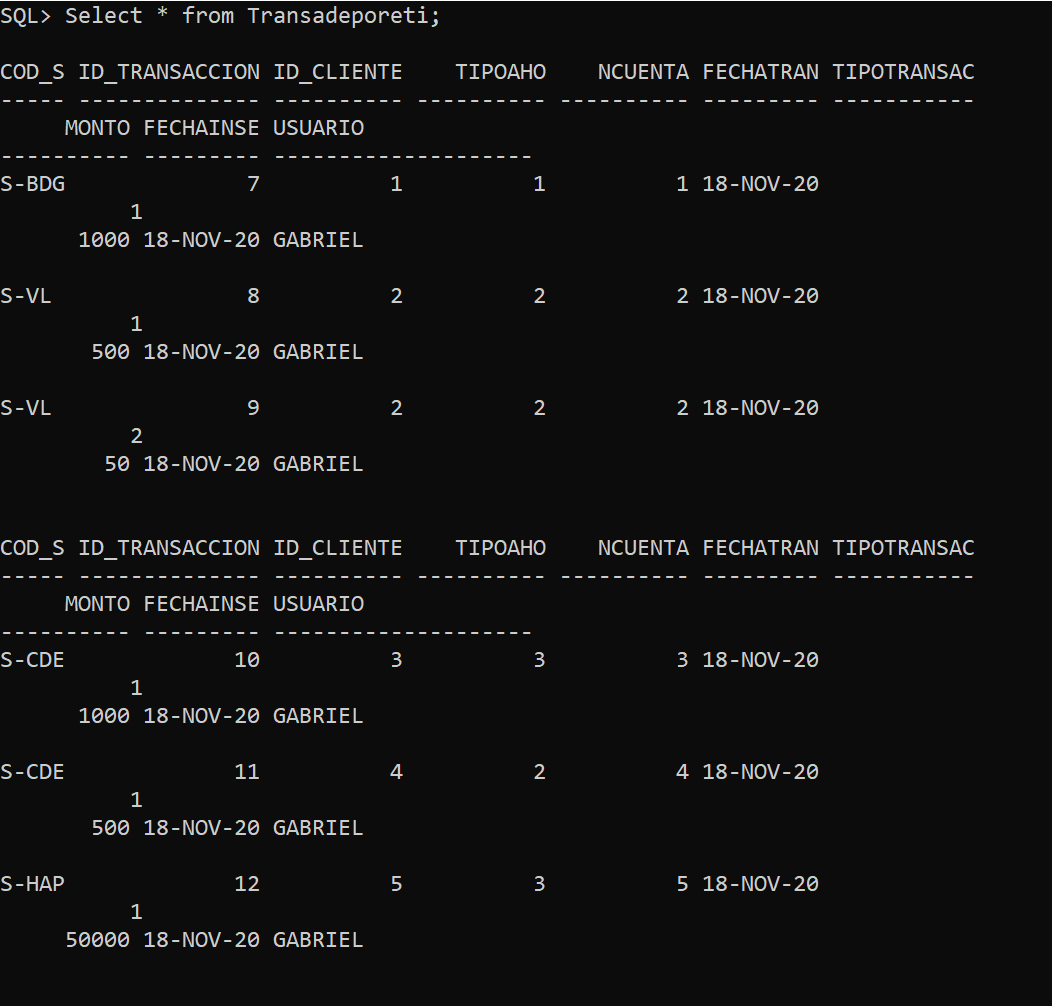
END;

/

* 
* 
* Tabla tipo de transaccion



* Tabla Transadeporeti



1. Procedimiento almacenado que actualice los depósitos o retiros de las cuentas de ahorro correspondientes. Deberá implementar un cursor que busque el depósitos/retiros insertados en la tabla uno a uno y los vaya actualizando en la tabla de ahorros de cada cliente (proceso en lote o proceso en línea usted decide). De la siguiente forma:
   1. Si el tipo de ahorro es navidad o escolar por cada deposito realizado debe calcular el interés que corresponde montodeposito \* tasade interes% que calculo que lo debe realizar una función diseñada previamente. El procedimiento debe actualizar el saldo de ahorro y el saldo interés de la cuenta de ahorro de los clientes.
   2. Si el tipo de ahorro es corriente simplemente se realizar la aplicación del depósito o retiro a la cuenta de ahorro del cliente correspondiente. De las únicas cuentas que se puede realizar retiros es de la cuenta de ahorro corriente por lo tanto el procedimiento debe controlar esta situación.

* CREATE OR REPLACE FUNCTION Calc\_Interes(

p\_monto Transadeporeti.monto%TYPE,

p\_tasaI Tipo\_Ahorro.Tasa\_inte%TYPE)RETURN NUMBER AS

BEGIN

RETURN (p\_monto\*p\_tasaI)/100;

END Calc\_Interes;

* CREATE OR REPLACE PROCEDURE Actualizar\_DR

(p\_error out varchar2)AS

CURSOR c\_Deporeti IS

SELECT cod\_sucursal,Id\_cliente,tipotransac,tipoaho,monto,Ncuenta

From Transadeporeti;

p\_codsuc Transadeporeti.cod\_sucursal%TYPE;

p\_Ttrans Transadeporeti.tipotransac%TYPE;

p\_Idcli Transadeporeti.Id\_cliente%TYPE;

p\_monto Transadeporeti.monto%TYPE;

p\_tipoaho Transadeporeti.tipoaho%TYPE;

p\_tasaI Tipo\_Ahorro.Tasa\_inte%TYPE;

p\_calcinte Ahorros.Sinteres%TYPE;

p\_SaldoA Ahorros.SAhorro%TYPE;

p\_Ncuenta Transadeporeti.Ncuenta%TYPE;

BEGIN

OPEN c\_Deporeti;

LOOP

FETCH c\_Deporeti INTO p\_codsuc,p\_Idcli,p\_Ttrans,p\_tipoaho,p\_monto,p\_Ncuenta;

Select Tasa\_inte INTO p\_tasaI

From Tipo\_Ahorro WHERE (ID\_TipoA=p\_tipoaho);

SELECT SAhorro INTO p\_SaldoA

FROM Ahorros

WHERE Id\_cliente=p\_Idcli AND cod\_sucursal=p\_codsuc AND tipoaho=p\_tipoaho;

IF ((p\_tipoaho = 1 OR p\_tipoaho = 3) AND p\_Ttrans=1) THEN

p\_calcinte := Calc\_Interes(p\_monto,p\_tasaI);

UPDATE Ahorros

SET

SAhorro =SAhorro+p\_monto,

Sinteres = Sinteres+p\_calcinte,

Usuario=user,

Fecha\_mod=sysdate,

Fecha\_dep=sysdate

WHERE p\_Idcli=Id\_cliente AND p\_Ncuenta=Ncuenta;

ELSIF ((p\_tipoaho = 1 OR p\_tipoaho = 3) AND p\_Ttrans=2) THEN

DBMS\_OUTPUT.PUT\_LINE('NO SE PUEDE RETIRAR DE ESTE TIPO DE CUENTA');

ELSIF (p\_tipoaho = 2 AND p\_Ttrans=1) THEN

UPDATE Ahorros

SET

SAhorro =SAhorro+p\_monto,

Usuario=user,

Fecha\_mod=sysdate,

Fecha\_dep=sysdate

WHERE p\_Idcli=Id\_cliente AND p\_Ncuenta=Ncuenta;

ELSIF (p\_tipoaho = 2 AND p\_Ttrans=2) THEN

IF(p\_monto <= p\_SaldoA) THEN

UPDATE Ahorros

SET

SAhorro =SAhorro-p\_monto,

Usuario=user,

Fecha\_mod=sysdate,

Fecha\_ret = sysdate

WHERE p\_Idcli=Id\_cliente AND p\_Ncuenta=Ncuenta;

ELSE

DBMS\_OUTPUT.PUT\_LINE('Fondos insuficiente');

END IF;

END IF;

EXIT WHEN c\_Deporeti%notfound;

END LOOP;

CLOSE c\_Deporeti;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

p\_error := 'No se encontraron datos';

WHEN OTHERS THEN

p\_error := 'No se actualizo la tabla ahorros';

COMMIT;

END Actualizar\_DR;

/

* DECLARE

v\_error varchar2(100);

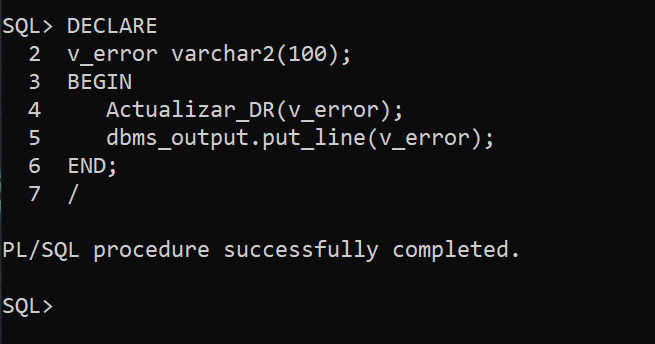
BEGIN

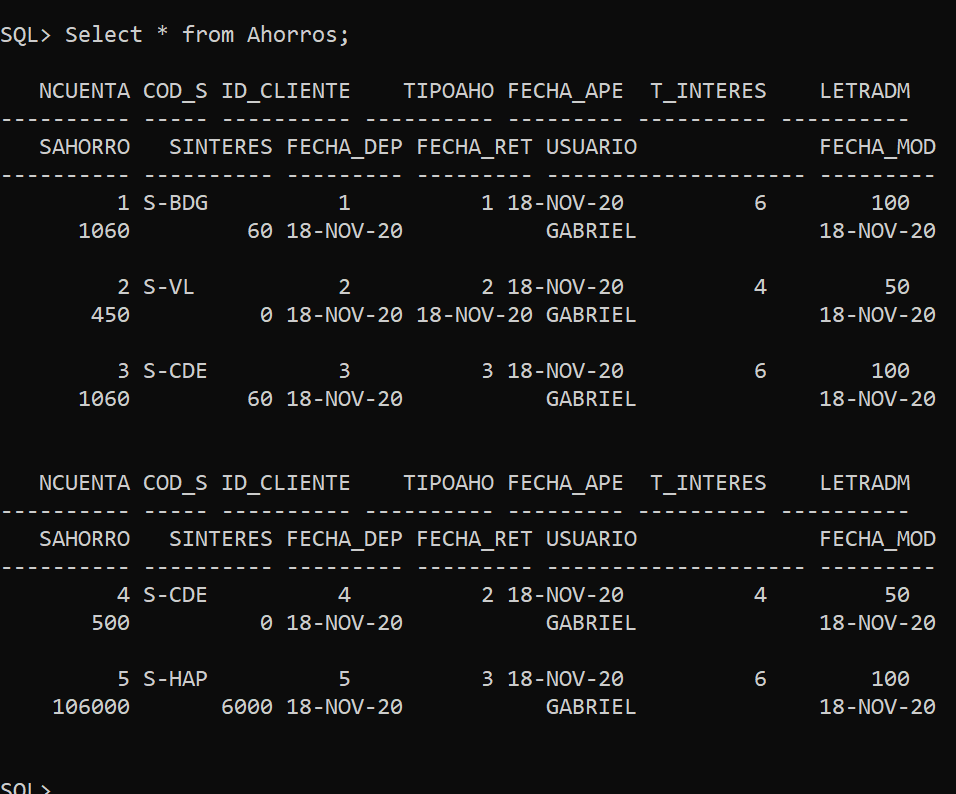
Actualizar\_DR(v\_error);

dbms\_output.put\_line(v\_error);

END;

/

* 
* Tabla ahorros



1. Triggers. Para las actualizaciones en la tabla de sucursales.

* CREATE OR REPLACE TRIGGER T\_SucAhorro

AFTER UPDATE OR DELETE ON Ahorros

FOR EACH ROW

BEGIN

UPDATE sucursal\_ahorro

Set

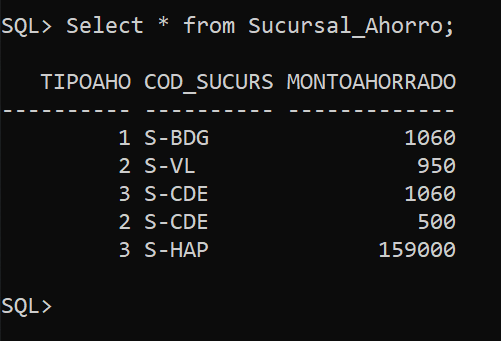
montoahorrado = montoahorrado+:new.SAhorro

Where cod\_sucursal=:NEW.cod\_sucursal AND tipoaho=:NEW.tipoaho;

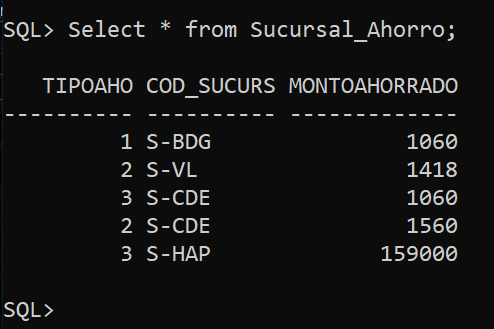
END T\_SucAhorro;

/

* Luego de ejecutar el proceso de actualizar los depositos y retiros



* Luego de ejecutar el proceso de pago de iteres mensual



1. Triggers. Para inserciones en la tabla de auditoria.

* CREATE OR REPLACE TRIGGER T\_Auditoria

AFTER UPDATE OR DELETE ON Ahorros

FOR EACH ROW

BEGIN

IF(:NEW.tipoaho = 1 OR :NEW.tipoaho = 3)THEN

INSERT INTO auditoria

Values (SAditroia.NEXTVAL,'Ahorros',:NEW.Id\_cliente,:NEW.tipoaho,1,:OLD.SAhorro,:NEW.SAhorro-:OLD.SAhorro,:NEW.SAhorro,user,Sysdate);

ELSIF (:NEW.SAhorro-:OLD.SAhorro>0)then

INSERT INTO auditoria

Values (SAditroia.NEXTVAL,'Ahorros',:NEW.Id\_cliente,:NEW.tipoaho,1,:OLD.SAhorro,:NEW.SAhorro-:OLD.SAhorro,:NEW.SAhorro,user,Sysdate);

ELSIF (:NEW.SAhorro-:OLD.SAhorro<0)then

INSERT INTO auditoria

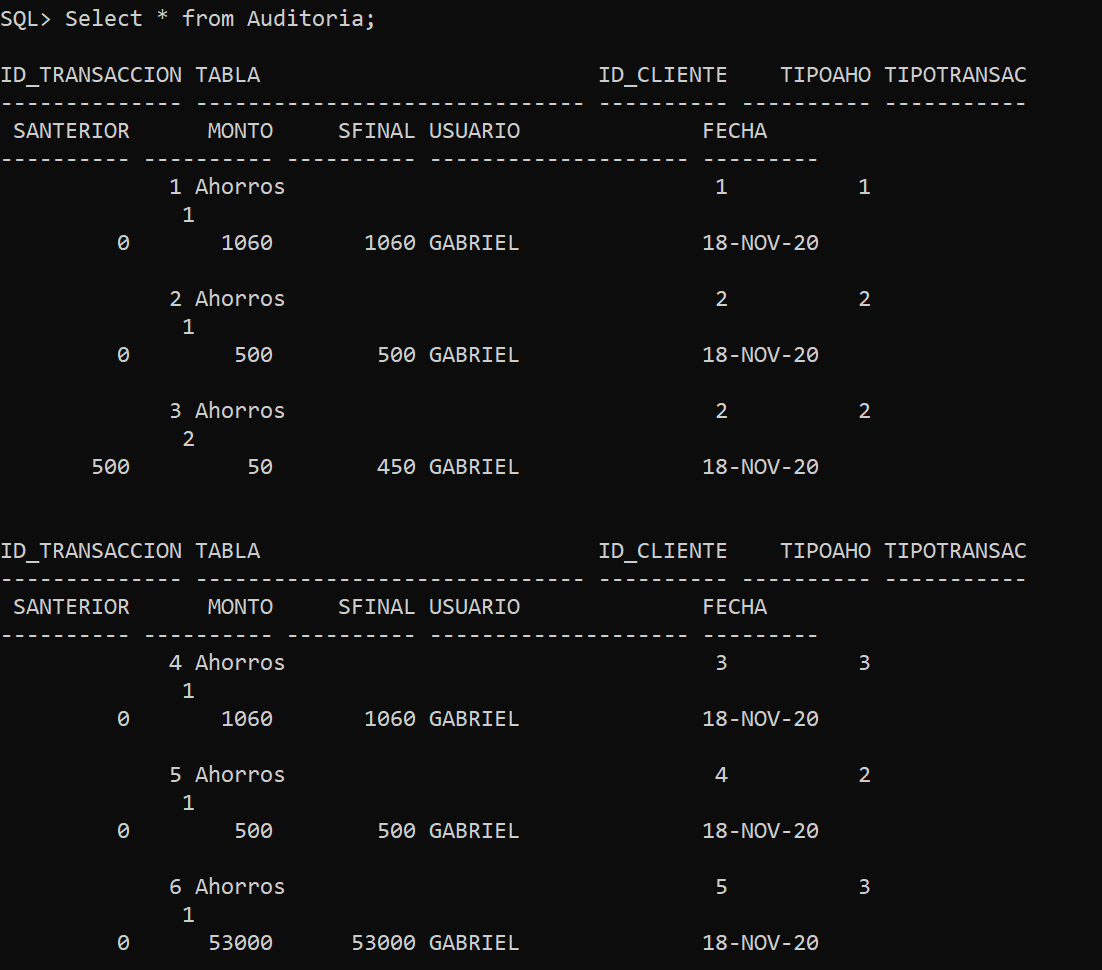
Values (SAditroia.NEXTVAL,'Ahorros',:NEW.Id\_cliente,:NEW.tipoaho,2,:OLD.SAhorro,:OLD.SAhorro-:NEW.SAhorro,:NEW.SAhorro,user,Sysdate);

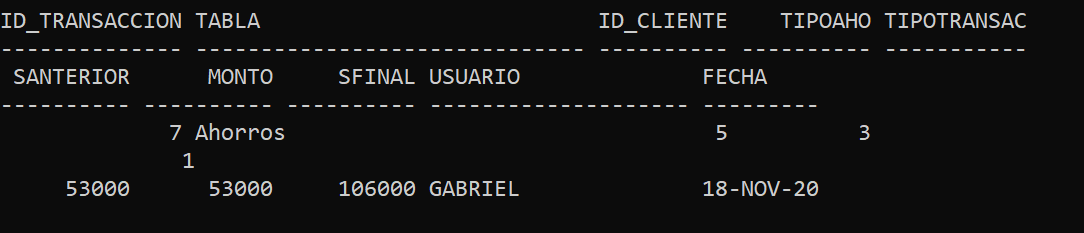
END IF;

END T\_Auditoria;

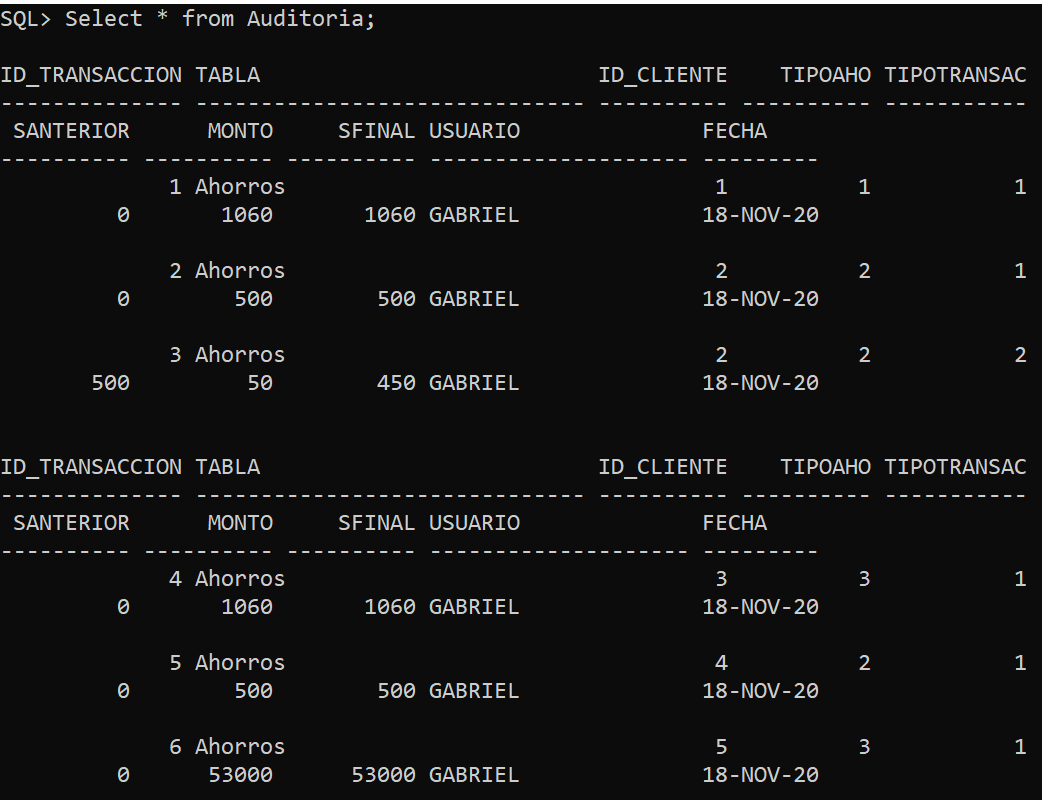
/

* Luego de ejecutar el proceso de actualizar los depositos y retiros





* Luego de ejecutar el proceso de pago de iteres mensual





1. Procedimiento. Llegamos a fin de mes y hay pagar los interés a la cuenta de ahorro corriente. Deberá diseñar un cursor que consulte todos los ahorros corrientes de forma controlada y calcule los interés que habrá que pagar mensualmente a estos ahorros (saldo de ahorros por la tasade interes% ) este cálculo lo realizar una función. El procedimiento deberá actualizar el saldo del ahorro y saldo de interés de las cuentas.’ El interés es sumado al saldo de ahorro’.

* CREATE OR REPLACE FUNCTION Interes\_Acorriente(

p\_SaldoT Ahorros.SAhorro%TYPE,

p\_interes tipo\_ahorro.tasa\_inte%type)

RETURN NUMBER AS

BEGIN

RETURN (p\_SaldoT\*p\_interes)/100;

END Interes\_Acorriente;

* CREATE OR REPLACE PROCEDURE Pago\_Interes

(p\_error out varchar2)AS

CURSOR c\_InCorriente IS

SELECT Id\_cliente,Ncuenta,SAhorro,tipoaho,cod\_sucursal

FROM ahorros WHERE tipoaho=2;

p\_idclien Ahorros.Id\_cliente%TYPE;

p\_Ncuenta Ahorros.Ncuenta%TYPE;

p\_Sahorro Ahorros.SAhorro%TYPE;

p\_intneres Tipo\_Ahorro.Tasa\_inte%TYPE;

p\_tipoah Ahorros.tipoaho%TYPE;

p\_Cinteres Ahorros.Sinteres%TYPE;

p\_codsuc Ahorros.cod\_sucursal%TYPE;

BEGIN

OPEN c\_InCorriente;

LOOP

FETCH c\_InCorriente INTO p\_idclien,p\_Ncuenta,p\_Sahorro,p\_tipoah,p\_codsuc;

Select Tasa\_inte INTO p\_intneres

From Tipo\_Ahorro WHERE (ID\_TipoA=p\_tipoah);

p\_Cinteres:= Interes\_Acorriente(p\_Sahorro,p\_intneres);

UPDATE ahorros

SET

SAhorro=SAhorro+ p\_Cinteres,

Sinteres=p\_Cinteres,

Fecha\_mod=sysdate,

Usuario =user

WHERE Id\_cliente=p\_idclien AND Ncuenta=p\_Ncuenta;

UPDATE Sucursal\_Ahorro

SET

montoahorrado = montoahorrado+ p\_Cinteres

Where cod\_sucursal=p\_codsuc AND tipoaho= p\_tipoah;

EXIT WHEN c\_InCorriente%NOTFOUND;

END LOOP;

CLOSE c\_InCorriente;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

p\_error := 'No se encontraron datos';

WHEN OTHERS THEN

p\_error := 'No se actualizo la tabla ahorros';

COMMIT;

END Pago\_Interes;

* DECLARE

v\_error varchar2(100);

BEGIN

Pago\_Interes(v\_error);

dbms\_output.put\_line(v\_error);

END;

/

