UNIVERSIDAD TECNOLÓGICA DE PANAMÁ

FACULTAD DE INGENIERÍA EN SISTEMAS COMPUTACIONALES

LIC. ING. EN SISTEMAS DE INFORMACIÓN

LABORATORIO 7

CURSO: SISTEMA BASE DE DATOS 2

PROFESOR:

HENRY LEZCANO

ESTUDIANTES:

MILAGROS CAMPOS 8-948-227

GUADALUPE CASTILLO 8-929-2252

ELIONAYS ROSAS 9-756-2182

ALEJANDRO URRIOLA 9-755-1141

GRUPO:

1IF131

II SEMESTRE, 2020

**Laboratorio 7**

--CREACIÓN DE LA TABLA TIPO\_AHORRO

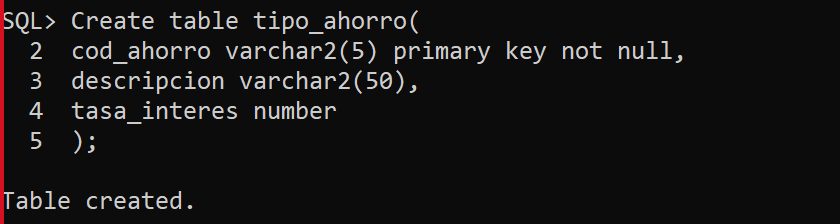
Create table tipo\_ahorro(

cod\_ahorro varchar2(5) primary key not null,

descripcion varchar2(50),

tasa\_interes number

);



Create table Sucursal (

Cod\_Sucursal varchar2(5) primary key not null,

Nom\_Sucursal varchar2(25)

);

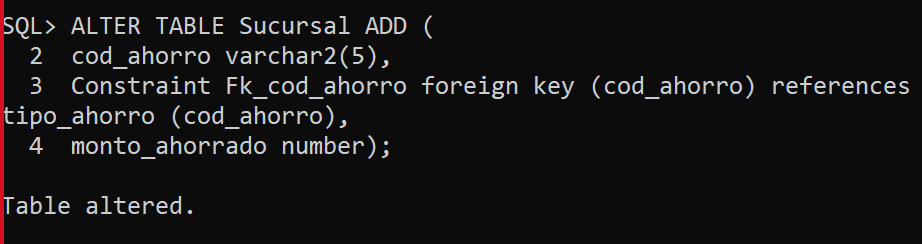
--MODIFICACIONES A LA TABLA SUCURSAL

ALTER TABLE Sucursal ADD (

cod\_ahorro varchar2(5),

Constraint Fk\_cod\_ahorro foreign key (cod\_ahorro) references tipo\_ahorro (cod\_ahorro),

monto\_ahorrado number);



--CREACION DE LA TABLA AHORROS

Create table Ahorros(

cod\_ahorro varchar2(5) ,

num\_cta number, --secuencia---

fecha\_open date,

interes number,

letra number,

sal\_ahorro number,

sal\_interes number,

fecha\_dep date,

fecha\_ret date,

usuario number,

fecha\_modif date,

ID\_Cliente number,

Constraint Fk\_otro\_id\_cliente foreign key (ID\_Cliente) references Banco\_Clientes (ID\_Cliente),

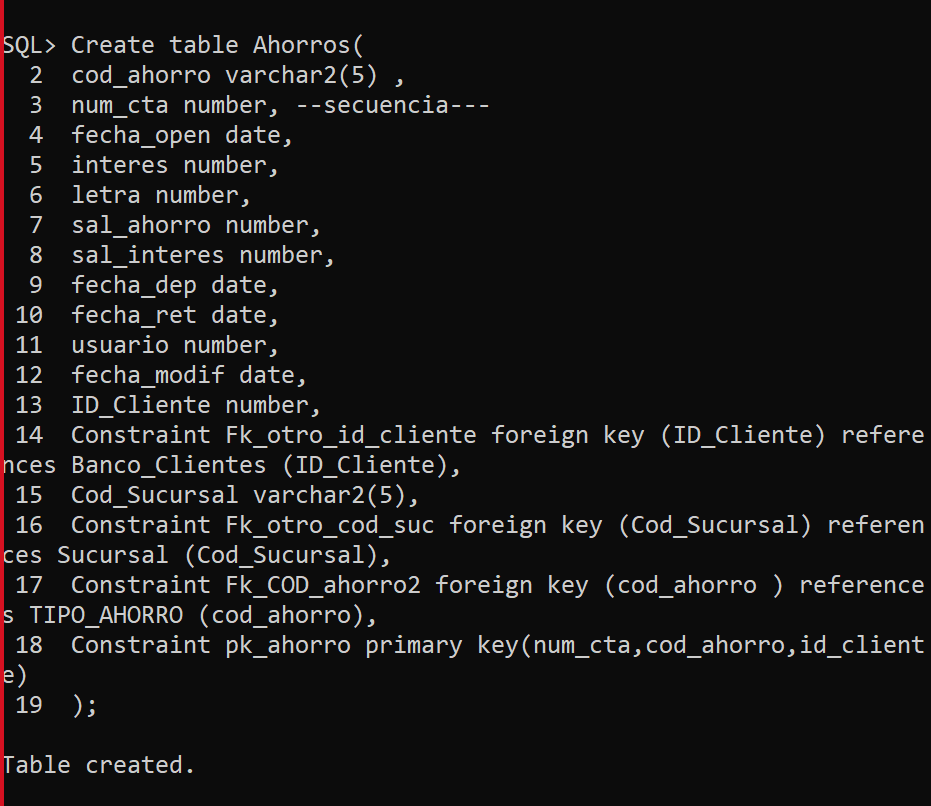
Cod\_Sucursal varchar2(5),

Constraint Fk\_otro\_cod\_suc foreign key (Cod\_Sucursal) references Sucursal (Cod\_Sucursal),

Constraint Fk\_COD\_ahorro2 foreign key (cod\_ahorro ) references TIPO\_AHORRO (cod\_ahorro),

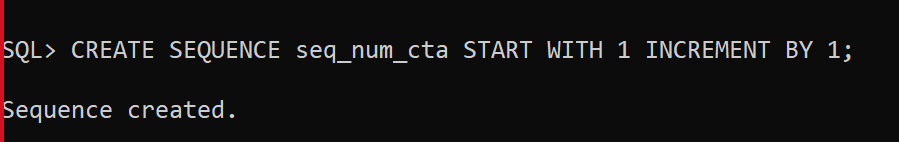
Constraint pk\_ahorro primary key(num\_cta,cod\_ahorro,id\_cliente)

);



--CREACION DE SECUENCIA PARA NUMERO DE CUENTA

CREATE SEQUENCE seq\_num\_cta START WITH 1 INCREMENT BY 1;



--Creación de la tabla Transac\_dr

Create table Transac\_dr(

ID\_Transaccion number primary key,

tipo\_transac varchar2(10),

fecha\_transac date,

monto\_dr number,

fecha\_inser date,

usuario number,

num\_cta number,

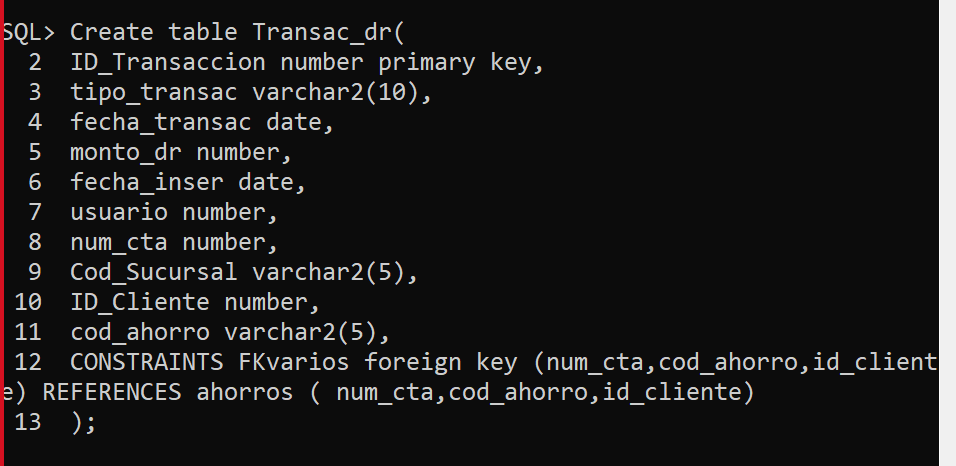
Cod\_Sucursal varchar2(5),

ID\_Cliente number,

cod\_ahorro varchar2(5),

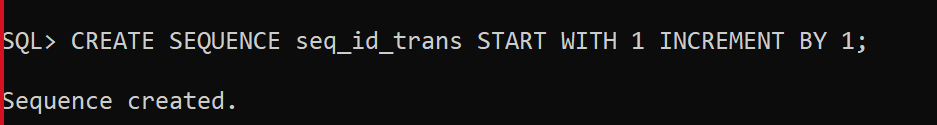
CONSTRAINTS FKvarios foreign key (num\_cta,cod\_ahorro,id\_cliente) REFERENCES ahorros ( num\_cta,cod\_ahorro,id\_cliente)

);



--CREACION DE SECUENCIA PARA ID\_Transaccion

CREATE SEQUENCE seq\_id\_trans START WITH 1 INCREMENT BY 1;



--CREACIÓN DE LA TABLA AUDITORÍA

Create table Auditoria(

ID\_Transaccion number PRIMARY KEY, --secuencia

tipo\_transac varchar2(10),

saldo\_cta\_anterior number,

monto\_transac number,

saldo\_final number,

num\_cta number,

usuario number,

fecha date,

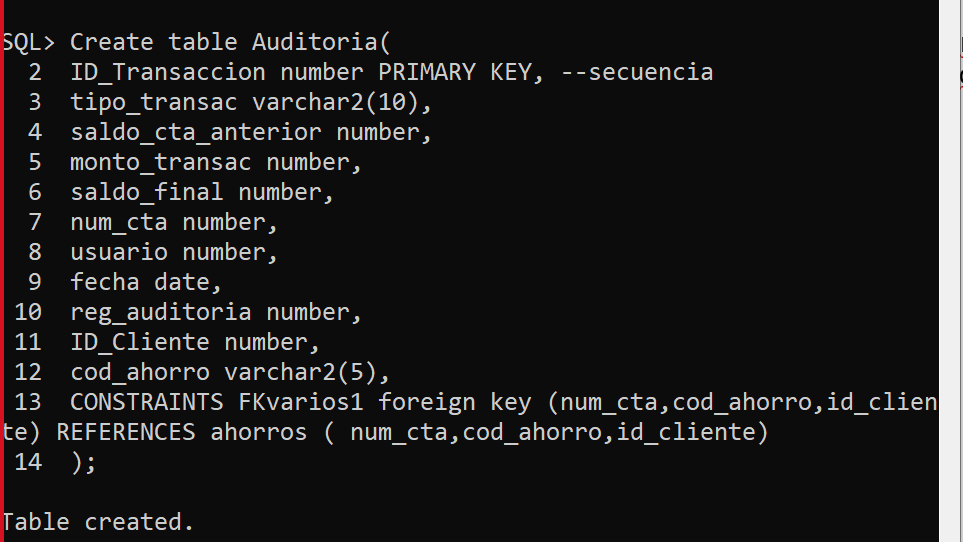
reg\_auditoria number,

ID\_Cliente number,

cod\_ahorro varchar2(5),

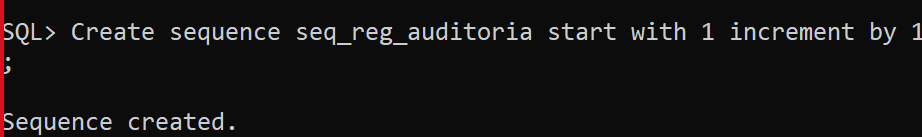
CONSTRAINTS FKvarios1 foreign key (num\_cta,cod\_ahorro,id\_cliente) REFERENCES ahorros ( num\_cta,cod\_ahorro,id\_cliente)

);

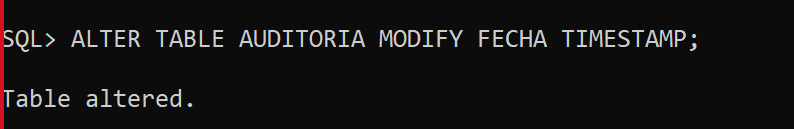


--creación de la secuencia para reg\_auditoria

Create sequence seq\_reg\_auditoria start with 1 increment by 1;



ALTER TABLE AUDITORIA MODIFY FECHA TIMESTAMP;

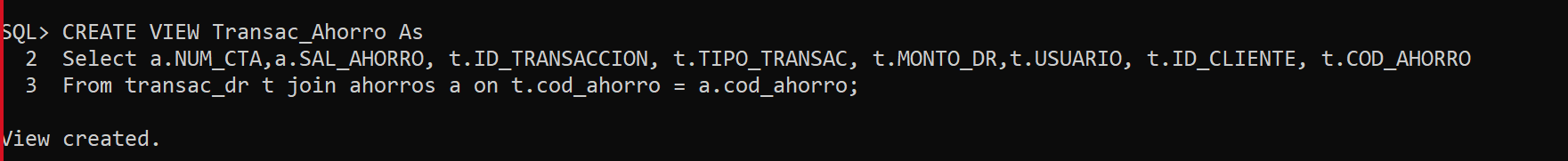


---Vista para TRIGGER Auditoria

CREATE VIEW Transac\_Ahorro As

Select a.NUM\_CTA,a.SAL\_AHORRO, t.ID\_TRANSACCION, t.TIPO\_TRANSAC, t.MONTO\_DR,t.USUARIO, t.ID\_CLIENTE, t.COD\_AHORRO

From transac\_dr t join ahorros a on t.cod\_ahorro = a.cod\_ahorro;



--PROCEDIMIENTO PARA ALMACENAMIENTO DE LA TABLA TIPO AHORRO

CREATE OR REPLACE PROCEDURE p\_Tipo\_Ahorro (

p\_cod\_ahorro tipo\_ahorro.cod\_ahorro%TYPE,

p\_descripcion tipo\_ahorro.descripcion%TYPE,

p\_tasa\_interes tipo\_ahorro.tasa\_interes%TYPE

)

AS

BEGIN

INSERT INTO tipo\_ahorro (cod\_ahorro, descripcion, tasa\_interes)

        Values (p\_cod\_ahorro, p\_descripcion, p\_tasa\_interes);

EXCEPTION

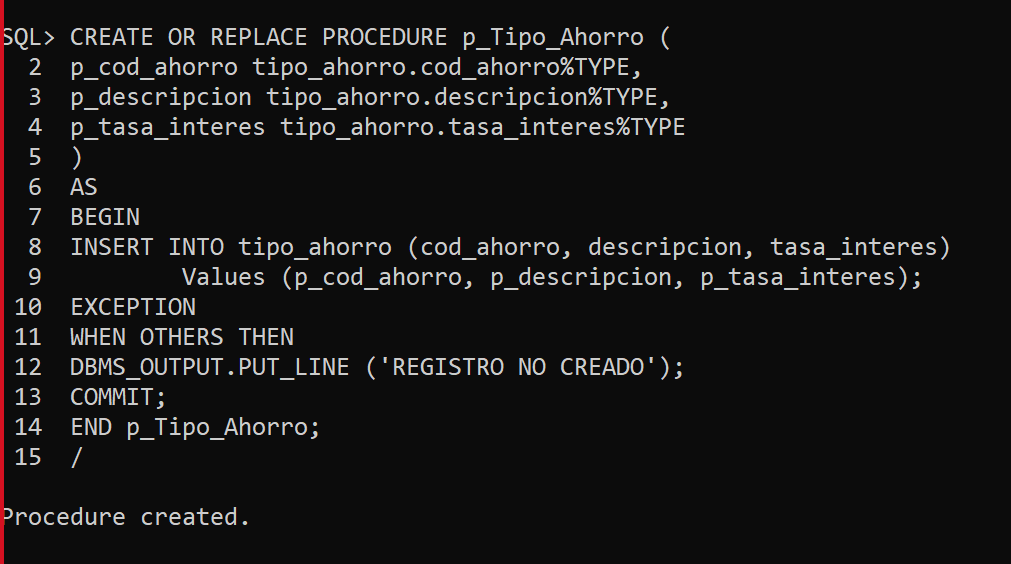
WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE ('REGISTRO NO CREADO');

COMMIT;

END p\_Tipo\_Ahorro;

/



--PROCEDIMIENTO PARA ALMACENAMIENTO DE LA TABLA AHORRO APROBADO

CREATE OR REPLACE PROCEDURE p\_Ahorro\_Aprobado (

p\_cod\_ahorro Ahorros.cod\_ahorro%TYPE,

p\_num\_cta   Ahorros.num\_cta%TYPE,

p\_fecha\_open Ahorros.fecha\_open%TYPE,

p\_interes Ahorros.interes%TYPE,

p\_letra Ahorros.letra%TYPE,

p\_sal\_ahorro Ahorros.sal\_ahorro%TYPE,

p\_sal\_interes Ahorros.sal\_interes%TYPE,

p\_fecha\_dep Ahorros.fecha\_dep%TYPE,

p\_fecha\_ret Ahorros.fecha\_ret%TYPE,

p\_usuario Ahorros.usuario%TYPE,

p\_fecha\_modif Ahorros.fecha\_modif%TYPE,

p\_ID\_Cliente Ahorros.ID\_Cliente%TYPE,

p\_Cod\_Sucursal Ahorros.Cod\_Sucursal%TYPE

)

AS

BEGIN

INSERT INTO Ahorros (cod\_ahorro, num\_cta, fecha\_open , interes,

letra, sal\_ahorro, sal\_interes , fecha\_dep, fecha\_ret, usuario, fecha\_modif, ID\_Cliente, Cod\_Sucursal)

Values (p\_cod\_ahorro, p\_num\_cta, p\_fecha\_open , p\_interes , p\_letra, p\_sal\_ahorro, p\_sal\_interes , p\_fecha\_dep, p\_fecha\_ret, p\_usuario, p\_fecha\_modif, p\_ID\_Cliente, p\_Cod\_Sucursal);

/\*EXCEPTION

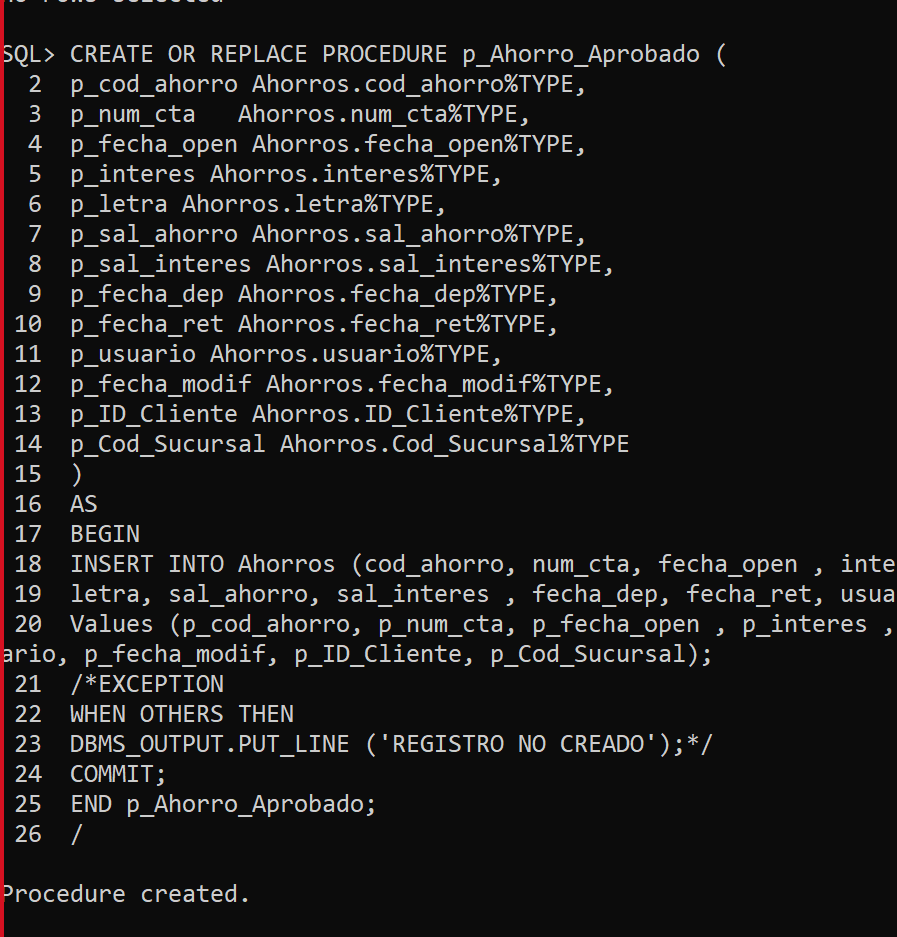
WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE ('REGISTRO NO CREADO');\*/

COMMIT;

END p\_Ahorro\_Aprobado;

/



--PROCEDIMIENTO PARA ALMACENAMIENTO DE LA TABLA TRANSAC\_dr

CREATE OR REPLACE PROCEDURE p\_Transacciones (

p\_ID\_Transaccion  Transac\_dr.ID\_Transaccion%TYPE,

p\_tipo\_transac Transac\_dr.tipo\_transac%TYPE,

p\_fecha\_transac Transac\_dr.fecha\_transac%TYPE,

p\_monto\_dr Transac\_dr.monto\_dr%TYPE,

p\_fecha\_inser Transac\_dr.fecha\_inser%TYPE,

p\_usuario Transac\_dr.usuario%TYPE,

p\_Cod\_Sucursal Transac\_dr.Cod\_Sucursal%TYPE,

p\_ID\_Cliente Transac\_dr.ID\_Cliente%TYPE,

p\_cod\_ahorro Transac\_dr.cod\_ahorro%TYPE,

p\_num\_cta Transac\_dr.num\_cta%TYPE

)

AS

BEGIN

INSERT INTO Transac\_dr (ID\_Transaccion, tipo\_transac , fecha\_transac , monto\_dr , fecha\_inser,usuario, num\_cta , Cod\_Sucursal , ID\_Cliente , cod\_ahorro)

Values (p\_ID\_Transaccion, p\_tipo\_transac , p\_fecha\_transac , p\_monto\_dr , p\_fecha\_inser, p\_usuario, p\_num\_cta, p\_Cod\_Sucursal , p\_ID\_Cliente ,p\_cod\_ahorro);

EXCEPTION

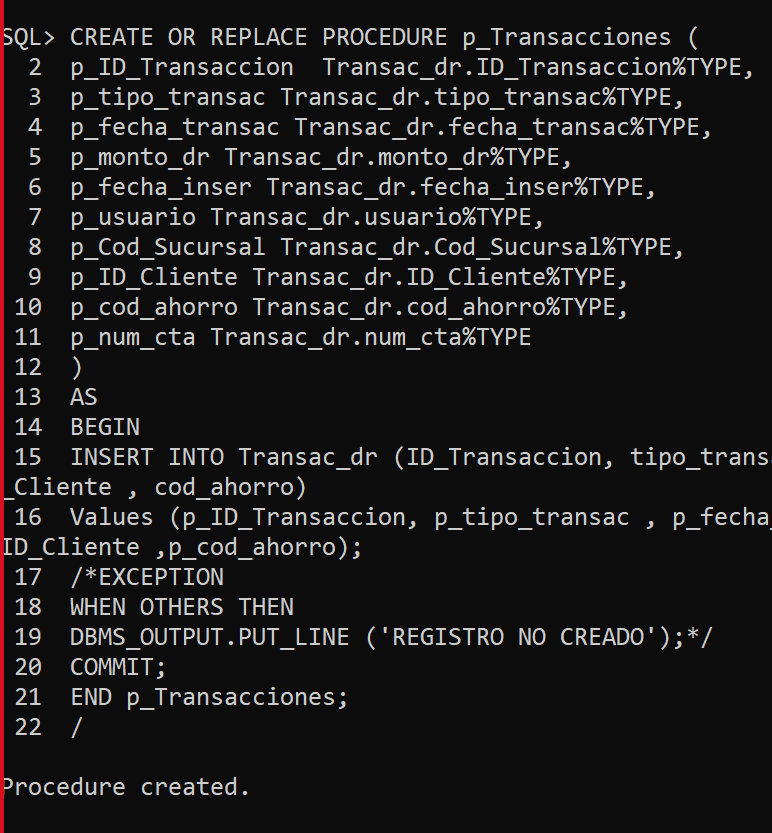
WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE ('REGISTRO NO CREADO');

COMMIT;

END p\_Transacciones;

/



--Procedimiento que actualiza las transacciones en la cuenta

CREATE OR REPLACE PROCEDURE update\_transac

AS

CURSOR Recorrido IS

Select monto\_dr, ID\_Cliente, Cod\_Sucursal,tipo\_transac , cod\_ahorro, num\_cta from Transac\_dr;

v\_sal\_ahorro Ahorros.sal\_ahorro%TYPE;

v\_cal\_int Ahorros.sal\_interes%TYPE;

v\_monto\_dr Transac\_dr.monto\_dr%TYPE;

v\_ID\_Cliente Transac\_dr.ID\_Cliente%TYPE;

v\_Cod\_Sucursal Transac\_dr.Cod\_Sucursal%TYPE;

v\_tipo\_transac Transac\_dr.tipo\_transac%TYPE;

v\_id\_trans Transac\_dr.id\_transaccion%TYPE;

v\_cod\_ahorro Transac\_dr.cod\_ahorro%TYPE;

v\_tasa\_interes tipo\_ahorro.tasa\_interes%TYPE;

v\_num\_cta Transac\_dr.num\_cta%TYPE;

BEGIN

OPEN Recorrido;

LOOP

Fetch Recorrido INTO v\_monto\_dr, v\_ID\_Cliente ,v\_Cod\_Sucursal, v\_tipo\_transac, v\_cod\_ahorro, v\_num\_cta;

Select tasa\_interes INTO v\_tasa\_interes

from tipo\_ahorro where cod\_ahorro= v\_cod\_ahorro;

Select sal\_ahorro INTO v\_sal\_ahorro

FROM Ahorros

where num\_cta=v\_num\_cta AND cod\_ahorro= v\_cod\_ahorro

and ID\_Cliente = v\_ID\_Cliente and Cod\_Sucursal= v\_Cod\_Sucursal;

IF(( v\_cod\_ahorro= ‘01’ OR v\_cod\_ahorro= ‘03’) AND v\_tipo\_transac= ‘DEPOSITO’) THEN

v\_cal\_int:= f\_calcular\_interes(v\_monto\_dr, v\_tasa\_interes);

UPDATE Ahorros

SET

sal\_ahorro = sal\_ahorro + v\_monto\_dr + v\_cal\_int,

sal\_interes = sal\_interes + v\_cal\_int,

fecha\_dep= sysdate,

fecha\_modif= sysdate

WHERE num\_cta=v\_num\_cta AND cod\_ahorro= v\_cod\_ahorro

and ID\_Cliente = v\_ID\_Cliente ;

ELSIF (v\_cod\_ahorro= ‘02’ AND v\_tipo\_transac= ‘DEPOSITO’) THEN

UPDATE Ahorros

SET

sal\_ahorro= sal\_ahorro + v\_monto\_dr,

fecha\_dep= sysdate,

fecha\_modif= sysdate

WHERE num\_cta=v\_num\_cta AND cod\_ahorro= v\_cod\_ahorro

and ID\_Cliente = v\_ID\_Cliente ;

ELSIF ((v\_cod\_ahorro= ‘01’ AND v\_cod\_ahorro= ‘03’) AND v\_tipo\_transac= ‘RETIRO’) THEN

DBMS\_OUTPUT.PUT\_LINE ('NO PUEDES RETIRAR DE ESTA CUENTA');

ELSIF (v\_cod\_ahorro= ‘02’ AND v\_tipo\_transac= ‘RETIRO’) THEN

  IF (v\_sal\_ahorro >= v\_monto\_dr) THEN

UPDATE Ahorros

SET

     sal\_ahorro= sal\_ahorro -v\_monto\_dr,

fecha\_ret= sysdate,

fecha\_modif= sysdate

WHERE num\_cta=v\_num\_cta AND cod\_ahorro= v\_cod\_ahorro

and ID\_Cliente = v\_ID\_Cliente ;

ELSE

DBMS\_OUTPUT.PUT\_LINE ('NO SE PUDO REALIZAR LA TRANSACCION');

END IF;

END IF;

EXIT WHEN Recorrido%NOTFOUND;

END LOOP;

CLOSE Recorrido;

END update\_transac;

/

--procedimiento de consulta de ahorros corrientes

 CREATE OR REPLACE PROCEDURE ahorro\_corriente (

v\_cod\_ahorro Ahorros.cod\_ahorro%TYPE,

v\_sal\_interes Ahorros.sal\_interes%TYPE,

v\_sal\_ahorro Ahorros.sal\_ahorro%TYPE,

v\_interes Ahorros.interes%TYPE )

AS

 CURSOR act\_ahorro IS

 SELECT cod\_ahorro, sal\_interes, sal\_ahorro, interes

FROM Ahorros;

BEGIN

 FOR i IN act\_ahorro LOOP

 --SELECT interes into v\_interes from Ahorros

 --where cod\_ahorro = i.cod\_ahorro;

 IF (v\_cod\_ahorro ='02') THEN

 UPDATE Ahorros

 SET

 sal\_ahorro= v\_sal\_ahorro + f\_interes(v\_sal\_ahorro, v\_interes),

 sal\_interes= v\_sal\_interes

 where cod\_ahorro = i.cod\_ahorro;

 END IF;

 END LOOP;

 EXCEPTION

 WHEN NO\_DATA\_FOUND THEN

 DBMS\_OUTPUT.PUT\_LINE ('No se encontro datos');

 COMMIT;

 END ahorro\_corriente;

 /

**Creacion de Funciones**

--función que calcula el interés

CREATE OR REPLACE FUNCTION f\_calcular\_interes(

f\_interes tipo\_ahorro.tasa\_interes%TYPE,

f\_saldo Transac\_dr.monto\_dr%TYPE

)return number

AS

interes\_ahorro number(5,3);

BEGIN

interes\_ahorro := (f\_saldo \* (f\_interes/100));

RETURN interes\_ahorro;

END f\_calcular\_interes;

/

CREATE OR REPLACE FUNCTION f\_interes(

f\_intereses Ahorros.interes%TYPE,

f\_sal\_ahorro Ahorros.sal\_ahorro%TYPE

)return number

AS

interes\_mensual number(5,3);

BEGIN

interes\_mensual := (f\_sal\_ahorro \* (f\_intereses/100));

RETURN interes\_mensual;

END f\_interes;

/

--trigger para las actualizaciones en la tabla sucursales

CREATE OR REPLACE TRIGGER act\_sucursal

BEFORE INSERT OR DELETE OR UPDATE ON Ahorros FOR EACH ROW

BEGIN

IF INSERTING THEN

INSERT INTO temp\_sucursal (Cod\_sucursal, Cod\_ahorro,monto\_ahorrado,fecha\_modify,Usuario\_modify)

VALUES ( :new.cod\_sucursal,:new.cod\_ahorro, :new.sal\_ahorro, SYSDATE, :new.Usuario);

ELSIF UPDATING THEN

UPDATE temp\_sucursal

SET monto\_ahorrado= :new.sal\_ahorro,

Usuario\_modify = :new.Usuario

WHERE Cod\_ahorro=:new.cod\_ahorro AND Cod\_sucursal = :new.cod\_sucursal;

END IF;

END act\_sucursal ;

/

--trigger para inserciones en la tabla auditoria

CREATE OR REPLACE TRIGGER t\_aud

INSTEAD OF INSERT ON Transac\_Ahorro FOR EACH ROW

BEGIN

INSERT INTO temp\_auditoria (ID\_TRANSACCION, TIPO\_TRANSAC, SALDO\_CTA\_ANTERIOR, MONTO\_TRANSAC, SALDO\_FINAL,USUARIO,FECHA, REG\_AUDITORIA, ID\_CLIENTE)

VALUES ( :new.ID\_TRANSACCION,:new.TIPO\_TRANSAC, :old.Sal\_ahorro, :new.Monto\_DR, :new.sal\_ahorro,:new.usuario,sysdate,seq\_reg\_auditoria.NEXTVAL,:new.ID\_cliente);

--END IF;

END t\_aud;

/

--INSERCION tabla tipo ahorro

BEGIN

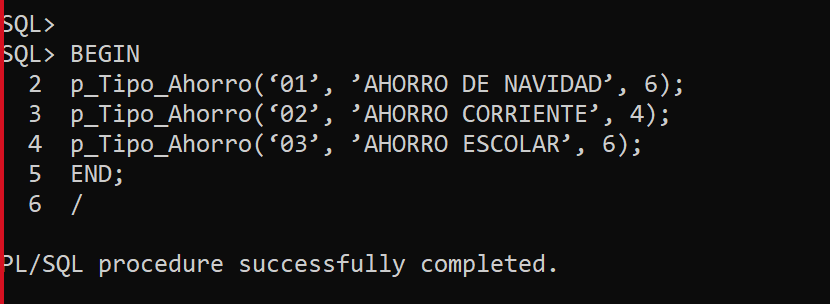
p\_Tipo\_Ahorro(‘01’, ’AHORRO DE NAVIDAD’, 6);

p\_Tipo\_Ahorro(‘02’, ’AHORRO CORRIENTE’, 4);

p\_Tipo\_Ahorro(‘03’, ’AHORRO ESCOLAR’, 6);

END;

/



--INSERCION tabla ahorros

BEGIN

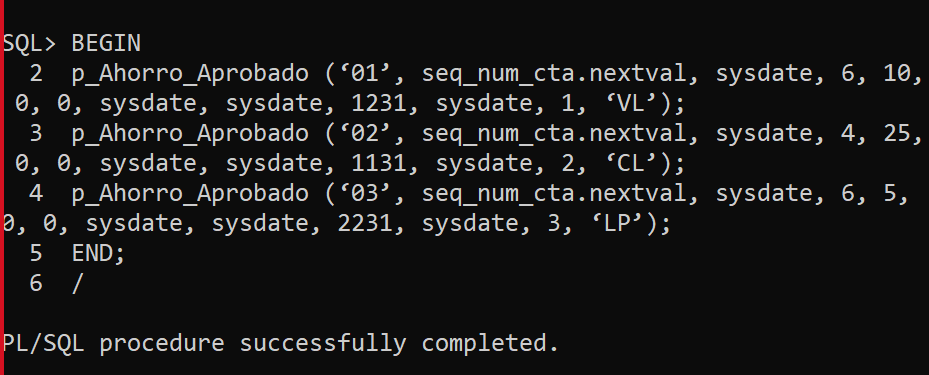
p\_Ahorro\_Aprobado (‘01’, seq\_num\_cta.nextval, sysdate, 6, 10, 0, 0, sysdate, sysdate, 1231, sysdate, 1, ‘VL’);

p\_Ahorro\_Aprobado (‘02’, seq\_num\_cta.nextval, sysdate, 4, 25, 0, 0, sysdate, sysdate, 1131, sysdate, 2, ‘CL’);

p\_Ahorro\_Aprobado (‘03’, seq\_num\_cta.nextval, sysdate, 6, 5, 0, 0, sysdate, sysdate, 2231, sysdate, 3, ‘LP’);

END;

/



--INSERCION tabla transacciones

BEGIN

p\_Transacciones (seq\_id\_trans.nextval, ‘DEPOSITO’, sysdate, 25,sysdate, 1243,4,‘VL’,1, ‘01’ );

p\_Transacciones (seq\_id\_trans.nextval, ‘DEPOSITO’, sysdate, 10,sysdate, 6578,5,‘CL’,2,’02’);

p\_Transacciones (seq\_id\_trans.nextval, ‘DEPOSITO’, sysdate, 5,sysdate, 9807,6,‘LP’, 3,’03’);

END;

/

BEGIN

Update\_transac;

END;

/

BEGIN

Ahorro\_corriente;

END;

/