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

Equipo #1

Laboratorio #7: “Implementación de Procedimientos, Funciones y Triggers”

Facilitador:

Ing. Henry Lezcano

Integrantes:

Aguilar, Milagros 3-740-771

Atencio, Anel 8-950-868

Márquez, Paola 8-949-1108

Rojas, Reynaldo 8-950-792

Grupo:

1IF131

II Semestre, 2020

**--TABLAS**

create table Tipo\_ahorro (

ID\_ahorro number not null,

ahorro varchar2(20) not null,

interes\_ahorro number not null,

constraint ID\_ahorro\_pk primary key (ID\_ahorro)

);

create table Ahorros (

Cod\_sucursal number not null,

ID\_Cliente number not null,

ID\_ahorro number not null,

Num\_ahorro number not null,

fecha\_apertura date not null,

letra\_dep number not null,

saldo\_ahorro number not null,

saldo\_interes number not null,

fecha\_deposito date,

fecha\_retiro date,

usuario varchar2(20) not null,

fecha\_modif date not null,

constraint Suc\_ahorro\_fk foreign key (Cod\_sucursal) references Sucursal (Cod\_sucursal),

constraint ID\_aho\_ahor\_fk foreign key (Id\_ahorro) references Tipo\_ahorro (ID\_ahorro),

constraint ID\_cliente\_aho\_fk foreign key (ID\_cliente) references Cliente (ID\_cliente),

constraint Cliente\_ahorro\_pk primary key (Num\_ahorro)

);

create table Tipo\_transaccion (

ID\_tipo\_transc number not null,

transaccion varchar2(20) not null,

constraint ID\_transac\_pk primary key (ID\_tipo\_transc)

);

create table Transadeporeti (

Cod\_sucursal number not null,

id\_transaccion number not null,

id\_cliente number not null,

num\_ahorro number not null,

id\_ahorro number not null,

fechatransaccion date not null,

id\_tipo\_transc number not null,

monto\_dr number not null,

fechainsercion date not null,

usuario varchar2(20) not null,

constraint cod\_suc\_tran\_fk foreign key (Cod\_sucursal) references Sucursal (Cod\_sucursal),

constraint id\_cl\_tran\_fk foreign key (ID\_Cliente) references Cliente (ID\_Cliente),

constraint id\_ahor\_tran\_fk foreign key (id\_ahorro) references Tipo\_ahorro (id\_ahorro),

constraint id\_tipo\_tran\_fk foreign key (id\_tipo\_transc) references Tipo\_transaccion (id\_tipo\_transc),

constraint num\_ahorro\_fk foreign key (num\_ahorro) references Ahorros (num\_ahorro),

constraint id\_tran\_pk primary key (id\_transaccion)

);

create table Auditoria\_saldo (

Id\_transaccion number,

Tabla varchar2(20),

accion char,

Id\_cliente number,

Id\_ahorro number,

saldo\_anterior number,

saldo\_final number,

usuario varchar2(20),

fecha date,

constraint id\_audit\_saldo\_pk primary key (id\_transaccion)

);

create table Auditoria\_transac (

Id\_transaccion number,

Tabla varchar2(20),

accion char,

Id\_cliente number,

Id\_ahorro number,

id\_tipo\_transc number,

monto\_dr number,

usuario varchar2(20),

fecha date,

constraint id\_audit\_transac\_pk primary key (id\_transaccion)

);

create table Sucursal\_ahorro(

upd\_suc number,

cod\_sucursal number,

id\_ahorro number,

monto\_aho\_v number,

monto\_aho\_n number,

usuario varchar2(20),

fecha date,

constraint upd\_pk\_suc primary key (upd\_suc)

);

**--SECUENCIAS**

CREATE SEQUENCE id\_transaccion

start with 1

increment by 1;

CREATE SEQUENCE num\_cuenta

START WITH 1

INCREMENT BY 1;

create sequence trig\_suc

start with 1

increment by 1;

create sequence trig\_transac

start with 1

increment by 1;

create sequence trig\_aud

start with 1

increment by 1;

**--FUNCIONES**

CREATE OR REPLACE FUNCTION interes\_deposito (

p\_monto\_dep transadeporeti.monto\_dr%TYPE,

p\_interes tipo\_ahorro.interes\_ahorro%TYPE )

RETURN NUMBER AS

BEGIN

RETURN (p\_monto\_dep \* p\_interes)/100;

END interes\_deposito;

/

create or replace FUNCTION interes\_corriente (

p\_montod ahorros.saldo\_ahorro%TYPE,

p\_interes tipo\_ahorro.interes\_ahorro%TYPE)

RETURN NUMBER AS

BEGIN

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

END interes\_corriente;

/

**--TRIGGERS**

CREATE OR REPLACE TRIGGER suc\_trig AFTER UPDATE OR INSERT OR DELETE

ON Ahorros FOR EACH ROW

BEGIN

IF INSERTING THEN

INSERT INTO Sucursal\_ahorro (upd\_suc, cod\_sucursal, id\_ahorro, monto\_aho\_v, monto\_aho\_n, usuario, fecha)

VALUES (trig\_suc.nextval, :new.cod\_sucursal, :new.id\_ahorro, :old.saldo\_ahorro, :new.saldo\_ahorro, user, sysdate);

ELSIF UPDATING THEN

UPDATE Sucursal\_ahorro

SET

Upd\_suc= trig\_suc.nextval,

Cod\_sucursal = :NEW.cod\_sucursal,

Id\_ahorro = :new.id\_ahorro,

monto\_aho\_n = :old.saldo\_ahorro +(:new.saldo\_ahorro- :old.saldo\_ahorro),

monto\_aho\_v = :OLD.saldo\_ahorro,

usuario = user,

fecha = sysdate

where id\_ahorro = :new.id\_ahorro AND cod\_sucursal = :new.cod\_sucursal;

END IF;

END suc\_trig;

/

CREATE OR REPLACE TRIGGER Aud\_trig\_saldo

BEFORE UPDATE OR DELETE OR INSERT ON Ahorros

FOR EACH ROW

DECLARE

v\_TipoCambio CHAR(1);

BEGIN

IF UPDATING THEN

v\_TipoCambio := 'U';

INSERT INTO AUDITORIA\_saldo

VALUES (trig\_aud.nextval, 'Ahorros', v\_tipoCambio, :NEW.ID\_cliente, :NEW.id\_ahorro,:OLD.saldo\_ahorro,:NEW.saldo\_ahorro, user, sysdate);

ELSIF INSERTING THEN

v\_TipoCambio := 'I';

INSERT INTO AUDITORIA\_saldo

VALUES (trig\_aud.nextval, 'Ahorros', v\_tipoCambio, :NEW.ID\_cliente, :NEW.id\_ahorro,:OLD.saldo\_ahorro,:NEW.saldo\_ahorro, user, sysdate);

ELSIF DELETING THEN

v\_TipoCambio := 'D';

INSERT INTO AUDITORIA\_saldo

VALUES (trig\_aud.nextval, 'Ahorros', v\_tipoCambio, :NEW.ID\_cliente, :NEW.id\_ahorro,:OLD.saldo\_ahorro,:NEW.saldo\_ahorro, user, sysdate);

END IF;

END Aud\_trig\_saldo;

/

CREATE OR REPLACE TRIGGER Aud\_trig\_transac

BEFORE UPDATE OR DELETE OR INSERT ON transadeporeti

FOR EACH ROW

DECLARE

v\_TipoCambio CHAR(1);

BEGIN

IF UPDATING THEN

v\_TipoCambio := 'U';

INSERT INTO AUDITORIA\_transac

VALUES (trig\_transac.nextval, 'Ahorros', v\_tipoCambio, :NEW.ID\_cliente, :NEW.id\_ahorro,:new.id\_tipo\_transc, :new.monto\_dr, user, sysdate);

ELSIF INSERTING THEN

v\_TipoCambio := 'I';

INSERT INTO AUDITORIA\_transac

VALUES (trig\_transac.nextval, 'Ahorros', v\_tipoCambio, :NEW.ID\_cliente, :NEW.id\_ahorro,:new.id\_tipo\_transc, :new.monto\_dr, user, sysdate);

ELSIF DELETING THEN

v\_TipoCambio := 'D';

INSERT INTO AUDITORIA\_transac

VALUES (trig\_transac.nextval, 'Ahorros', v\_tipoCambio, :NEW.ID\_cliente, :NEW.id\_ahorro,:new.id\_tipo\_transc, :new.monto\_dr, user, sysdate);

END IF;

END Aud\_trig\_transac;

/

**--PROCEDIMIENTOS**

CREATE OR REPLACE PROCEDURE add\_t\_ahorro (

p\_id\_ahorro IN tipo\_ahorro.id\_ahorro%TYPE,

p\_ahorro IN tipo\_ahorro.ahorro%TYPE,

p\_interes IN tipo\_ahorro.interes\_ahorro%TYPE

) AS

BEGIN

INSERT INTO tipo\_ahorro (id\_ahorro, ahorro, interes\_ahorro)

VALUES (p\_id\_ahorro, p\_ahorro, p\_interes);

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

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

COMMIT;

END add\_t\_ahorro;

/

CREATE OR REPLACE PROCEDURE ahorros\_aprob(

p\_cod\_sucursal IN ahorros.cod\_sucursal%TYPE,

P\_ID\_CLIENTE IN ahorros.ID\_CLIENTE%TYPE,

p\_id\_ahorro IN ahorros.id\_ahorro%TYPE,

P\_fecha\_apert IN ahorros.fecha\_apertura%TYPE,

P\_LETRA\_MENSUAL IN ahorros.LETRA\_dep%TYPE,

P\_saldo\_ahorro IN ahorros.saldo\_ahorro%TYPE,

P\_saldo\_interes IN ahorros.saldo\_interes%TYPE,

P\_fecha\_deposito IN ahorros.fecha\_deposito%TYPE,

P\_fecha\_retiro IN ahorros.fecha\_retiro%TYPE

)AS

BEGIN

INSERT INTO Ahorros (cod\_sucursal, ID\_CLIENTE, id\_ahorro, num\_ahorro, fecha\_apertura, LETRA\_dep, saldo\_ahorro, saldo\_interes, fecha\_deposito, fecha\_retiro, usuario, FECHA\_MODIF)

VALUES(P\_cod\_sucursal, P\_ID\_CLIENTE, P\_id\_ahorro, num\_cuenta.nextval, P\_fecha\_apert, P\_LETRA\_MENSUAL, P\_saldo\_ahorro, P\_saldo\_interes, P\_fecha\_deposito, P\_fecha\_retiro, user, sysdate);

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE ('No se encontró la tabla');

COMMIT;

END ahorros\_aprob;

/

create or replace PROCEDURE add\_transadeporeti (

p\_cod\_sucursal IN transadeporeti.cod\_sucursal%TYPE,

p\_id\_cliente IN transadeporeti.id\_cliente%TYPE,

p\_num\_ahorro IN transadeporeti.num\_ahorro%TYPE,

p\_id\_ahorro IN transadeporeti.id\_ahorro%TYPE,

p\_id\_tipo\_transc IN transadeporeti.id\_tipo\_transc%TYPE,

p\_monto\_dr IN transadeporeti.monto\_dr%TYPE

)AS

BEGIN

INSERT INTO transadeporeti (cod\_sucursal, id\_transaccion, id\_cliente, num\_ahorro, id\_ahorro, fechatransaccion, id\_tipo\_transc, monto\_dr, fechainsercion, usuario)

VALUES (p\_cod\_sucursal, id\_transaccion.nextval, p\_id\_cliente, p\_num\_ahorro, p\_id\_ahorro, sysdate, p\_id\_tipo\_transc, p\_monto\_dr, sysdate, user);

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

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

COMMIT;

END add\_transadeporeti;

/

create or replace PROCEDURE update\_dr

AS

CURSOR c\_depret IS

SELECT cod\_sucursal, id\_cliente, id\_ahorro, monto\_dr, id\_tipo\_transc, num\_ahorro

from Transadeporeti;

p\_sucursal transadeporeti.cod\_sucursal%TYPE;

p\_cliente transadeporeti.id\_cliente%TYPE;

p\_transac transadeporeti.id\_transaccion%TYPE;

p\_tipotrans transadeporeti.id\_tipo\_transc%TYPE;

p\_montod transadeporeti.monto\_dr%TYPE;

p\_funcion ahorros.saldo\_interes%TYPE;

p\_tipoa transadeporeti.id\_ahorro%TYPE;

p\_interes tipo\_ahorro.interes\_ahorro%TYPE;

p\_saldo\_actual ahorros.saldo\_ahorro%TYPE;

p\_num\_ahorro transadeporeti.num\_ahorro%TYPE;

BEGIN

OPEN c\_depret;

LOOP

FETCH c\_depret INTO p\_sucursal, p\_cliente, p\_tipoa, p\_montod, p\_tipotrans, p\_num\_ahorro;

SELECT interes\_ahorro INTO p\_interes

FROM tipo\_ahorro WHERE id\_ahorro = p\_tipoa;

SELECT saldo\_ahorro INTO p\_saldo\_actual FROM Ahorros WHERE id\_ahorro = p\_tipoa and id\_cliente =p\_cliente and cod\_sucursal=p\_sucursal;

IF ((p\_tipoa = 1 OR p\_tipoa = 3) AND p\_tipotrans = 1) THEN

p\_funcion := interes\_deposito (p\_montod, p\_interes);

UPDATE Ahorros

SET

saldo\_ahorro =saldo\_ahorro+p\_montod+p\_funcion,

saldo\_interes = saldo\_interes+p\_funcion,

fecha\_deposito = sysdate,

usuario = user,

fecha\_modif = sysdate

WHERE id\_cliente = p\_cliente and num\_ahorro=p\_num\_ahorro;

ELSIF (p\_tipoa = 2 AND p\_tipotrans = 1) THEN

UPDATE Ahorros

SET

saldo\_ahorro =saldo\_ahorro+p\_montod,

fecha\_deposito = sysdate,

usuario = user,

fecha\_modif = sysdate

WHERE id\_cliente = p\_cliente and num\_ahorro=p\_num\_ahorro;

ELSIF ((p\_tipoa=1 OR p\_tipoa=3) AND p\_tipotrans=2) THEN

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

ELSIF (p\_tipoa = 2 AND p\_tipotrans = 2 )THEN

IF (p\_saldo\_actual >= p\_montod) THEN

UPDATE Ahorros

SET

saldo\_ahorro =saldo\_ahorro-p\_montod,

fecha\_retiro = sysdate,

usuario = user,

fecha\_modif = sysdate

WHERE id\_cliente = p\_cliente and num\_ahorro=p\_num\_ahorro;

ELSE

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

END IF;

END IF;

EXIT WHEN c\_depret%NOTFOUND;

END LOOP;

CLOSE c\_depret;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

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

COMMIT;

END update\_dr;

/

CREATE OR REPLACE PROCEDURE update\_corriente

AS

CURSOR c\_corriente IS

SELECT id\_cliente, id\_ahorro, saldo\_ahorro, num\_ahorro

FROM Ahorros where id\_ahorro=2;

v\_cliente Ahorros.id\_cliente%TYPE;

v\_ahorro Ahorros.id\_ahorro%TYPE;

v\_saldoa Ahorros.saldo\_ahorro%TYPE;

v\_funcion Ahorros.saldo\_interes%TYPE;

v\_interes Tipo\_ahorro.interes\_ahorro%TYPE;

v\_num Ahorros.num\_ahorro%TYPE;

BEGIN

OPEN c\_corriente;

LOOP

FETCH c\_corriente INTO v\_cliente, v\_ahorro, v\_saldoa, v\_num;

SELECT interes\_ahorro INTO v\_interes

FROM tipo\_ahorro WHERE id\_ahorro = v\_ahorro;

v\_funcion := interes\_corriente (v\_saldoa, v\_interes);

UPDATE Ahorros

SET

saldo\_ahorro = saldo\_ahorro + v\_funcion,

saldo\_interes = saldo\_interes + v\_funcion,

usuario = user,

fecha\_modif = sysdate

WHERE id\_cliente = v\_cliente AND num\_ahorro = v\_num;

EXIT WHEN c\_corriente%NOTFOUND;

END LOOP;

CLOSE c\_corriente;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

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

COMMIT;

END update\_corriente;

/

CREATE OR REPLACE PROCEDURE add\_t\_transac (

p\_id\_tipo IN tipo\_transaccion.id\_tipo\_transc%TYPE,

p\_transac IN tipo\_transaccion.transaccion%TYPE

) AS

BEGIN

INSERT INTO tipo\_transaccion (id\_tipo\_transc, transaccion)

VALUES (p\_id\_tipo, p\_transac);

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

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

COMMIT;

END add\_t\_transac;

/

**--INVOCACIONES**

BEGIN

ADD\_T\_AHORRO (01,'Navidad', 6);

ADD\_T\_AHORRO (02,'Corriente',4);

ADD\_T\_AHORRO (03,'Escolar', 6);

END;

/

BEGIN

ADD\_T\_TRANSAC (1, 'Depósito');

ADD\_T\_TRANSAC (2, 'Retiro');

END;

/

BEGIN

AHORROS\_APROB (1,1,1,TO\_DATE('27-OCT-20'),100,0,0,NULL,NULL);

AHORROS\_APROB (2,2,2,TO\_DATE('31-OCT-20'),50,0,0,NULL,NULL);

AHORROS\_APROB (3,3,3,TO\_DATE('01-NOV-20'),200,0,0,NULL,NULL);

AHORROS\_APROB (4,5,1,TO\_DATE('01-NOV-20'),100,0,0,NULL,NULL);

AHORROS\_APROB (3,2,2,TO\_DATE('05-NOV-20'),200,0,0,NULL,NULL);

END;

/

select \* from ahorros;

BEGIN

ADD\_TRANSADEPORETI (1,1,1,1,1,100);

ADD\_TRANSADEPORETI (2,2,2,2,1,100);

ADD\_TRANSADEPORETI (3,3,3,3,1,200);

ADD\_TRANSADEPORETI (4,5,4,1,1,100);

ADD\_TRANSADEPORETI (3,2,5,2,1,200);

END;

/

BEGIN

update\_dr;

END;

/

select \* from ahorros;

BEGIN

update\_corriente;

END;

/

select \* from ahorros;

select \* from temp\_suc;

select \* from auditoria\_saldo;

select \* from auditoria\_transac;

**--VISTAS**

create view CLIENTE\_AHORRO as select c.nombre, c.apellido, c.cedula, t.ahorro from cliente c join ahorros a on a.id\_cliente = c.id\_cliente join tipo\_ahorro t on t.id\_ahorro = a.id\_ahorro;

create view Cliente\_montoah as select c.nombre, c.apellido, a.saldo\_ahorro from cliente c join ahorros a on c.id\_cliente = a.id\_cliente;

**CAPTURAS DE TRIGGERS**





