**UNIVERSIDAD TECNOLÓGICA DE PANAMÁ**

**FACULTAD DE INGENIERÍA DE SISTEMAS COMPUTACIONALES**

**DEPARTAMENTO DE SISTEMAS DE INFORMACIÓN, CONTROL Y EVALUACIÓN DE RECURSOS INFORMÁTICOS**

**LIC. EN ING. DE SISTEMAS DE INFORMACIÓN**

**SISTEMAS DE BASE DE DATOS II**

**Laboratorio N°7**

**Profesor:**

**Ing. Henry Lezcano**

**Integrantes:**

Cutire, Fernando (8-972-906)

Díaz, Gabriel (20-53-5198)

Escobar, Jorge (2-747-1772)

Feng, William (8-977-446)

**Grupo: 1IF131**

**Caso:**

Tomando en cuenta el Modelo proporcionado por la compañía Financiera SAEZMONTERO, S.A del Laboratorio No.2 y Modificado en laboratorio No. 6 que indicaba lo siguiente:

‘Compañía Financiera SAEN-MONTERO S.A necesita levantar un modelo físico de datos a partir de un modelo lógico relacional para una sección de su proceso de negocio (otorgamiento de préstamos), en lo corresponde a la entrada de la información de los cliente y los préstamos e información relacionada a esta sección del proceso a dicha base de datos’.

La Cía ha crecido lo suficiente para innovar en nuevos productos para la gran gama de cliente que la visitan. Y los dueños de esta quieren innovar en los productos de ahorros pero con los controles, que corresponden.

I. Realizar las siguientes tareas que afectan el Modelo Físico implementado previamente que permitirá optimizar la sección del proceso de negocio implementado una nueva área Proceso de Negocio’ ya que se han aperturado varias sucursales, se han aprobado préstamos y estos cliente son buenos candidatos para iniciar un programa de ahorro:

1. Agregar una tabla al modelo físico que almacenes las sucursales de la empresa la financiera con las restricciones correspondientes:

(tipoaho de ahorro, montoahorrado) usted debe realizar el planteamiento correcto para administrar los totales por sucursales para los ahorros.

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:

01 ahorro de navidad con tasa de interés det 6%

02 ahorro corriente con tasa de interés del 4%

03 ahorro de navidad con tasa de interés del 6%

1. Agregar la tabla de ahorros al modelo de base de datos cuyos atributos serán: Docente: Ing. Henry Lezcano MAS martes, 26 de Octubre de 2021

* Código de sucursal
* Id del cliente
* Tipo de ahorro
* Número de cuenta(secuencia)
* Fecha de apertura
* Tasa de interés
* Letra Depósito mensual(acuerdo como será incrementada la cuenta)
* Saldo del ahorro
* Saldo del interés
* Fecha de depósito
* Fecha de retiro
* Usuario
* Fecha de modificación

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

1. Agregar una tabla de auditoría que llevará la trazabilidad transaccional de cuentas de ahorros por el momento donde sus atributos serán:

* Id\_transaccion(secuencia)
* Tabla (relación critica que será auditada)
* Tipo de Operación (I=Insert, U=update, D=delete)
* Id cliente
* Tipo de ahorro
* Tipo transacción (1=deposito, 2=Retiro)
* Saldo de la cuenta anterior antes de la aplicación
* Monto del deposito o retiro
* Saldo final después de la aplicación
* Usuario
* Fecha (diamesaño, hora minuto y segundo

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,

Status (control de aplicación de la transacción)

fechainsercion(dd/mm/yyyy, hh:mm:ss),

usuario)

Importante en el Modelo Físico validar la Integridad referencial para los nuevos atributos de ser necesarios y para las nuevas estructuras que formarán parte del Modelo Físico Optimizado. Estos cambios impactan el modelo Lógico Relacional de la Base de Datos.

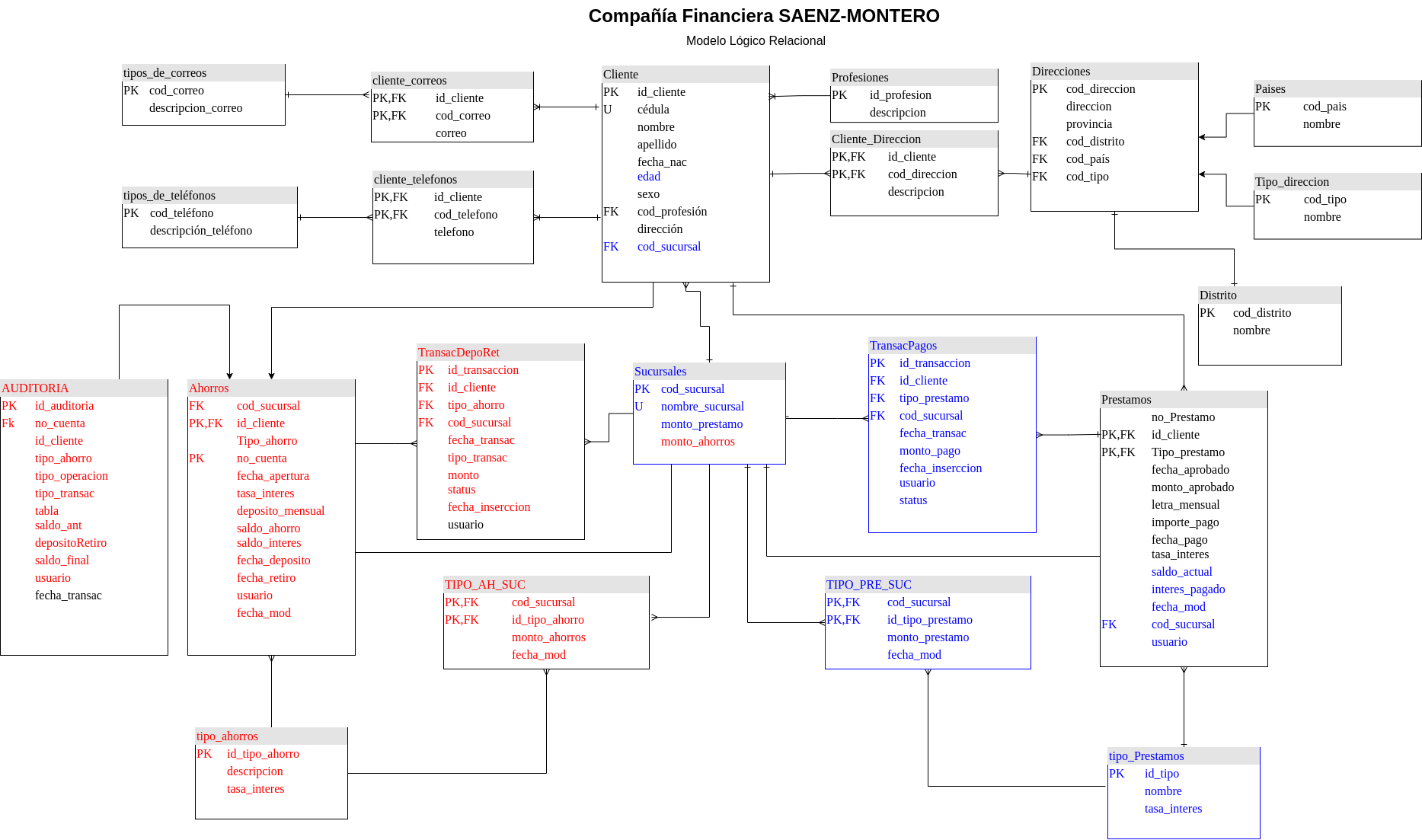
Procesos que deben ser implementado con la programación almacenada y que serán probados en la sustentación con las invocaciones correspondiente:

* Procedimiento almacenado para la carga o inserción de la tabla paramétrica
  + Tipo de ahorros
* Procedimiento almacenado para la apertura o inserción de los ahorros aprobados con toda la información correspondiente..
* 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.
* 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:

* Si el tipo de ahorro es navidad o escolar por cada depósito realizado debe calcular el interés que corresponde montodeposito \* tasade interes% que Calculo que 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
* Si el tipo de ahorro es corriente simplemente se realiza 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.
* Triggers. Para las actualizaciones en la tabla de sucursales los montos de los ahorros.
* Triggers Para las actualizaciones en la tabla de sucursales tipo de ahorros, los montos de los ahorros correspondientes.
* Triggers. Para inserciones en la tabla de auditoría.
* Procedimiento. Llegamos a fin de mes y hay que pagar los intereses 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 realiza 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’.

**Modelo**



**Codigo**

| set serveroutput on;  *----LABORATORIO 7------*  *-- 1 Tipo de ahorros* CREATE TABLE tipos\_ahorros (  id\_tipo\_ahorro NUMBER NOT NULL,  descripcion VARCHAR2(30),  tasa\_interes NUMBER (15,2),  CONSTRAINT tp\_ahorro\_pk PRIMARY KEY (id\_tipo\_ahorro) );    *-- 2 Tipo de AH Sucursal* CREATE TABLE TIPO\_AH\_SUC (  cod\_sucursal NUMBER NOT NULL,  id\_tipo\_ahorro NUMBER NOT NULL,  monto\_ahorros NUMBER(15,2) DEFAULT 0,  fecha\_mod DATE,   CONSTRAINT tipo\_ah\_suc\_pk PRIMARY KEY (cod\_sucursal, id\_tipo\_ahorro),  CONSTRAINT tipo\_suc\_fk FOREIGN KEY (cod\_sucursal)   REFERENCES SUCURSALES (cod\_sucursal),  CONSTRAINT tipo\_ah\_fk FOREIGN KEY (id\_tipo\_ahorro)   REFERENCES TIPOS\_AHORROS (id\_tipo\_ahorro) );  *-- 3 Ahorros* CREATE TABLE ahorros (  no\_cuenta NUMBER NOT null,  id\_cliente NUMBER NOT NULL,  tipo\_ahorro NUMBER NOT NULL,  cod\_sucursal NUMBER NOT NULL,  fecha\_apertura DATE,  tasa\_interes NUMBER(2, 2) DEFAULT 0,  deposito\_mensual NUMBER(15, 2),   saldo\_ahorro NUMBER(15, 2),   saldo\_interes NUMBER(15, 2),   usuario VARCHAR2(45),  fecha\_deposito NUMBER,  fecha\_retiro NUMBER,   fecha\_mod DATE,  CONSTRAINT ahorros\_pk PRIMARY KEY (no\_cuenta),  CONSTRAINT ahorros\_sucursales\_fk FOREIGN KEY (cod\_sucursal)  REFERENCES SUCURSALES (cod\_sucursal),  CONSTRAINT ahorros\_tipo\_ahorros\_fk FOREIGN KEY (tipo\_ahorro)  REFERENCES tipos\_ahorros (id\_tipo\_ahorro),  CONSTRAINT ahorros\_cliente\_fk FOREIGN KEY (id\_cliente)  REFERENCES clientes (id\_cliente) );    *-- 4 Transacciones Depo Reti* CREATE TABLE transaDepoReti (  id\_transaccion NUMBER NOT NULL,  no\_cuenta NUMBER NOT NULL,  id\_cliente NUMBER NOT NULL,  tipo\_ahorro NUMBER NOT NULL,  cod\_sucursal NUMBER NOT NULL,  fecha\_transac DATE,  tipo\_transac NUMBER,  monto NUMBER(15, 2) DEFAULT 0,  fecha\_inserccion DATE,  status VARCHAR2(2) NOT NULL,  usuario VARCHAR2(45),  CONSTRAINT tipo\_transac\_ck CHECK (tipo\_transac in(1, 2)),  CONSTRAINT status\_ck CHECK (status in('PE', 'PR')),  CONSTRAINT transaDepoReti\_pk PRIMARY KEY ( id\_transaccion ),   CONSTRAINT transadeporeti\_ahorros\_fk FOREIGN KEY (no\_cuenta)  REFERENCES ahorros(no\_cuenta),  CONSTRAINT transaDepoReti\_cliente\_fk FOREIGN KEY (id\_cliente)  REFERENCES clientes(id\_cliente),  CONSTRAINT transaDepoReti\_tipoahorro\_fk FOREIGN KEY (tipo\_ahorro)  REFERENCES tipos\_ahorros(id\_tipo\_ahorro),   CONSTRAINT transaDepoReti\_sucursales\_fk FOREIGN KEY ( cod\_sucursal )  REFERENCES sucursales ( cod\_sucursal ) );    -- 5 AUDITORIA  CREATE TABLE AUDITORIA (  id\_auditoria NUMBER NOT NULL,  no\_cuenta NUMBER,  id\_cliente NUMBER,  id\_tipo\_ahorro NUMBER,  tipo\_operacion CHAR(2),  tipo\_transac NUMBER,  tabla VARCHAR2(25),  saldo\_anterior NUMBER (15, 2) DEFAULT 0,  monto\_deposito NUMBER (15, 2) DEFAULT 0,  saldo\_final NUMBER (15, 2) DEFAULT 0,  usuario VARCHAR2(42),  fecha\_transaccion date,  CONSTRAINT tipo\_operacion\_ck CHECK ( tipo\_operacion IN ('I', 'U', 'D')),  CONSTRAINT auditoria\_tipo\_transac\_ck CHECK ( tipo\_transac IN(1, 2)),  CONSTRAINT auditoria\_pk PRIMARY KEY (id\_auditoria),  CONSTRAINT auditoria\_AHORROS\_fk FOREIGN KEY (NO\_CUENTA)  REFERENCES AHORROS (NO\_CUENTA)  );    *-- 6 ALTER TABLA SUCURSAL* ALTER TABLE   SUCURSALES   ADD monto\_ahorros NUMBER(15,2) DEFAULT 0 NOT NULL;    --- SECUENCIAS LABORATORIO 7 ---  --SECUENCIAS DE ID TIPO AHORRO --  CREATE SEQUENCE sec\_tipo\_aho  INCREMENT BY 1  START WITH 1  MAXVALUE 99999  MINVALUE 1;    --SECUENCIAS DE NUMERO CUENTA AHORRO  CREATE SEQUENCE sec\_no\_cuenta  INCREMENT BY 100  START WITH 100  MAXVALUE 99999  MINVALUE 100;    --SECUENCIAS DE ID TRANSACCION DEPOSITO RETIRO  CREATE SEQUENCE sec\_transacdeporeti  INCREMENT BY 1  START WITH 1  MAXVALUE 99999  MINVALUE 1;    --SECUENCIAS DE AUDITORIA-  CREATE SEQUENCE sec\_cod\_aut  INCREMENT BY 1  START WITH 1  MAXVALUE 99999  MINVALUE 1;    *----Procedimientos----*  *--TIPOS AHORRO* CREATE OR REPLACE PROCEDURE Nuevo\_tipoAhorro(  p\_ahorro\_descripcion IN tipos\_ahorros.descripcion%TYPE,  p\_ahorro\_tasa\_interes IN tipos\_ahorros.tasa\_interes%TYPE  )  IS  intSeqVal number(10);  BEGIN  select sec\_tipo\_aho.nextval into intSeqVal from dual;  INSERT into tipos\_ahorros (id\_tipo\_ahorro, descripcion, tasa\_interes)  VALUES (intSeqVal, p\_ahorro\_descripcion, p\_ahorro\_tasa\_interes);  COMMIT;  EXCEPTION  WHEN DUP\_VAL\_ON\_INDEX THEN  DBMS\_OUTPUT.PUT\_LINE('💣 Error: El tipo de ahorro ya existe.');  END Nuevo\_tipoAhorro;  /    --INSERT AHORRO --  CREATE OR REPLACE PROCEDURE insertAhorro(  p\_id\_cliente IN ahorros.id\_cliente%TYPE,  p\_tipo\_ahorro IN ahorros.tipo\_ahorro%TYPE,  p\_cod\_sucursal IN ahorros.cod\_sucursal%TYPE,  p\_depo\_mensual IN ahorros.deposito\_mensual%TYPE,  p\_fecha\_deposito IN ahorros.fecha\_deposito%TYPE,  p\_fecha\_retiro IN ahorros.fecha\_retiro%TYPE  )  IS  intSeqVal number(10);  v\_fecha\_ap date := SYSDATE;  v\_saldo\_ah number := 10;  v\_interes NUMBER;  v\_saldoInteres NUMBER := 10;  v\_fecha\_deposito number := p\_fecha\_deposito;  v\_fecha\_retiro NUMBER := p\_fecha\_retiro;    BEGIN  select sec\_no\_cuenta.nextval into intSeqVal from dual;  SELECT tasa\_interes INTO v\_interes FROM TIPOS\_AHORROS WHERE id\_tipo\_ahorro = p\_tipo\_ahorro;  INSERT INTO AHORROS  VALUES (  intSeqVal,  p\_id\_cliente,  p\_tipo\_ahorro,  p\_cod\_sucursal,  to\_date(v\_fecha\_ap,'DD-MM-YYY HH24:MI:SS'),  v\_interes,  p\_depo\_mensual,  v\_saldo\_ah,  v\_saldoInteres,  user,  v\_fecha\_deposito,  v\_fecha\_retiro,  to\_date(v\_fecha\_ap,'DD-MM-YYY HH24:MI:SS')  );  COMMIT;  EXCEPTION  WHEN DUP\_VAL\_ON\_INDEX THEN  DBMS\_OUTPUT.PUT\_LINE('💣 Error: El numero de cuenta ya existe.');  END insertAhorro;  /    -- INSERT TRANSADEPORETI --  CREATE OR REPLACE PROCEDURE insertTransaDeporeti (  p\_id\_cliente IN transaDepoReti.id\_cliente%TYPE,  p\_no\_cuenta IN transaDepoReti.no\_cuenta%TYPE,  p\_tipo\_ahorro IN transaDepoReti.tipo\_ahorro%TYPE,  p\_cod\_sucursal IN transaDepoReti.cod\_sucursal%TYPE,  p\_tipo\_transac IN transaDepoReti.tipo\_transac%TYPE,  p\_monto IN transaDepoReti.monto%TYPE  )  IS  intSeqVal number(10);  --v\_fecha\_ap date := SYSDATE;  --v\_usuario VARCHAR2(45) := USER;  v\_status CHAR(2) := 'PE';  v\_monto NUMBER(15,2) := p\_monto;  --v\_exception VARCHAR2(250) := EXCEPTION;  BEGIN  select sec\_transacdeporeti.nextval into intSeqVal from dual;  -- CONDICION DE TRANSACCION 1= DEPOSITO, 2=RETIRO solo de cuenta corriente se puede retirar.  IF p\_tipo\_ahorro = 2 AND (p\_tipo\_transac = 1 OR p\_tipo\_transac =2 ) THEN  INSERT INTO transaDepoReti  VALUES(  intSeqVal,  p\_no\_cuenta,  p\_id\_cliente,  p\_tipo\_ahorro,  p\_cod\_sucursal,  --to\_date(v\_fecha\_ap,'DD-MM-YYY HH24:MI:SS'),  SYSDATE,  p\_tipo\_transac,  v\_monto,  --to\_date(v\_fecha\_ap,'DD-MM-YYY HH24:MI:SS'),  SYSDATE,  v\_status,  user  );  ELSIF (p\_tipo\_ahorro = 1 OR p\_tipo\_ahorro = 3) AND p\_tipo\_transac = 1 THEN  INSERT INTO transaDepoReti  VALUES(  intSeqVal,  p\_no\_cuenta,  p\_id\_cliente,  p\_tipo\_ahorro,  p\_cod\_sucursal,  --to\_date(v\_fecha\_ap,'DD-MM-YYY HH24:MI:SS'),  SYSDATE,  p\_tipo\_transac,  v\_monto,  --to\_date(v\_fecha\_ap,'DD-MM-YYY HH24:MI:SS'),  SYSDATE,  v\_status,  user  );  ELSE  DBMS\_OUTPUT.PUT\_LINE('💣 Error: El tiempo de retiro no puede realizarse en este momento. Verifique su tipo de cuenta.');  END IF;  COMMIT;  EXCEPTION  WHEN DUP\_VAL\_ON\_INDEX THEN  DBMS\_OUTPUT.PUT\_LINE('💣 Error: La transacción ya existe.');  END insertTransaDeporeti;  /    -- ------------------ Proc 4 ---------------------------- --  CREATE OR REPLACE FUNCTION calcularInteresDelAhorro(  p\_tipoInteres number,  p\_monto number  )  RETURN NUMBER IS  V\_interes\_calculado NUMBER;  v\_monto NUMBER := p\_monto;  v\_interes NUMBER;  --v\_exeption EXCEPTION;  BEGIN  -- ASIGNA EL VALOR DEL INTERES EN BASE AL TIPO DE AHORRO  IF p\_tipoInteres = 1 THEN  v\_interes := 0.06;  ELSIF p\_tipoInteres = 2 THEN  v\_interes := 0.04;  ELSIF p\_tipoInteres = 3 THEN  v\_interes := 0.06;  END IF;  -- Interes calculado mediante el valor depositado y el tipo de interes  v\_interes\_calculado := (v\_monto \* v\_interes) + v\_monto;      RETURN v\_interes\_calculado;  EXCEPTION  WHEN NO\_DATA\_FOUND THEN  DBMS\_OUTPUT.PUT\_LINE('💣 Error: El préstamo no ha sido encontrado.');  END calcularInteresDelAhorro;  /    CREATE OR REPLACE PROCEDURE actualizarAhorros  IS  v\_id\_transac NUMBER;  v\_no\_cuenta NUMBER;  v\_id\_cliente NUMBER;  v\_tipo\_ahorro NUMBER;  v\_cod\_sucursal NUMBER;  v\_fecha\_transac date;  v\_tipo\_transac NUMBER;  v\_monto NUMBER(15, 2) DEFAULT 0;  v\_status char(2) := 'PE'; -- SOLO SE PROCESARAN LOS PENDIENTES  CURSOR c\_transaDepoReti IS  SELECT  id\_transaccion,  no\_cuenta,  id\_cliente,  tipo\_ahorro,  cod\_sucursal,  fecha\_transac,  tipo\_transac,  monto  FROM transaDepoReti  WHERE  status = v\_status;  BEGIN  --- SI TIPO = CORRIENTE NO SE REALIZA CALCULO DE INTERES EN EL SALDO Y EL INTERES.  ---- SI TIPO = NAVIDAD Y ESCOLAR SI SE LE REALIZA CALCULO DE INTERES EN EL SALDO.  --IF to\_char(CURRENT\_DATE, 'dd') = '1' OR to\_char(CURRENT\_DATE, 'dd') = '15' THEN  OPEN c\_transaDepoReti;  LOOP  FETCH c\_transaDepoReti INTO  v\_id\_transac,  v\_no\_cuenta,  v\_id\_cliente,  v\_tipo\_ahorro,  v\_cod\_sucursal,  v\_fecha\_transac,  v\_tipo\_transac,  v\_monto;  EXIT  WHEN c\_transadeporeti%NOTFOUND;    -- TIPO DE AHORRO,Navidad 1, Corriente 2, Escolar 3  -- CONDICION DE TRANSACCION 1= DEPOSITO, 2=RETIRO  IF v\_tipo\_transac = 1  THEN  IF v\_tipo\_ahorro = 2  THEN  UPDATE AHORROS  SET  saldo\_ahorro = saldo\_ahorro + v\_monto,  fecha\_mod = SYSDATE  WHERE  no\_cuenta = v\_no\_cuenta;  ELSE  UPDATE AHORROS  SET  saldo\_ahorro = calcularInteresDelAhorro(v\_tipo\_ahorro,v\_monto) + saldo\_ahorro,  saldo\_interes = saldo\_interes + (calcularInteresDelAhorro(v\_tipo\_ahorro,v\_monto) - v\_monto),  fecha\_mod = SYSDATE  WHERE  no\_cuenta = v\_no\_cuenta;  END IF;  ELSE  UPDATE AHORROS  SET  saldo\_ahorro = saldo\_ahorro - v\_monto,  fecha\_mod = SYSDATE  WHERE  no\_cuenta = v\_no\_cuenta;  END IF;  --ACTUALIZA EL ESTADO DEL DEPOSITO O RETIRO PARA QUE NO SE VUELVA A REPETIR  UPDATE TRANSADEPORETI  SET  status = 'PR'  WHERE  id\_transaccion = v\_id\_transac;  END LOOP;  CLOSE c\_transadeporeti;  EXCEPTION  WHEN NO\_DATA\_FOUND THEN  DBMS\_OUTPUT.PUT\_LINE('💣 Error: Los datos suministrados no existen');  END actualizarAhorros;  /    ---PROCEDIMIENTO 5 --  CREATE OR REPLACE PROCEDURE calcularInteresDeCorriente  IS  v\_no\_cuenta NUMBER;  v\_tipo\_ahorro NUMBER := 2;  v\_saldo\_ahorro NUMBER;  v\_saldo\_interes NUMBER;  CURSOR c\_ahorros IS  SELECT  no\_cuenta,  tipo\_ahorro,  saldo\_ahorro,  saldo\_interes  FROM AHORROS  WHERE  tipo\_ahorro = v\_tipo\_ahorro;  BEGIN    OPEN c\_ahorros;  LOOP  FETCH c\_ahorros INTO  v\_no\_cuenta,  v\_tipo\_ahorro,  v\_saldo\_ahorro,  v\_saldo\_interes;  EXIT  WHEN c\_ahorros%NOTFOUND;  -- IF to\_char(CURRENT\_DATE, 'dd') = '01' OR to\_char(CURRENT\_DATE, 'dd') = '15' THEN  IF to\_char(CURRENT\_DATE, 'dd') = '08' THEN  --IF v\_tipo\_ahorro = 2  --THEN  UPDATE AHORROS  SET  saldo\_interes = (calcularInteresDelAhorro(v\_tipo\_ahorro,v\_saldo\_ahorro) - v\_saldo\_ahorro),  saldo\_ahorro = calcularInteresDelAhorro(v\_tipo\_ahorro,v\_saldo\_ahorro),  fecha\_mod = SYSDATE  WHERE no\_cuenta = v\_no\_cuenta;  END IF;  END LOOP;  CLOSE c\_ahorros;  EXCEPTION  WHEN NO\_DATA\_FOUND THEN  DBMS\_OUTPUT.PUT\_LINE('💣 Error: Los datos suministrados no existen');  END calcularInteresDeCorriente;  /    -- TRIGGER 1 --  CREATE OR REPLACE TRIGGER actualizarSaldosSucursales  -- Inicio de la sección declarativa  AFTER UPDATE OF saldo\_ahorro  ON AHORROS  FOR EACH ROW  BEGIN  -- Inicio de la sección ejecutable  IF to\_char(CURRENT\_DATE, 'dd') = '08' AND :NEW.tipo\_ahorro = 2  THEN  IF :NEW.saldo\_interes > 0 THEN  UPDATE SUCURSALES  SET monto\_ahorros = monto\_ahorros + :NEW.saldo\_interes  WHERE COD\_SUCURSAL = :NEW.COD\_SUCURSAL;  ELSE  UPDATE SUCURSALES  SET monto\_ahorros = monto\_ahorros + :NEW.saldo\_ahorro  WHERE COD\_SUCURSAL = :NEW.COD\_SUCURSAL;  END IF;  ELSE  UPDATE SUCURSALES  SET monto\_ahorros = monto\_ahorros + :NEW.saldo\_ahorro  WHERE COD\_SUCURSAL = :NEW.COD\_SUCURSAL;  END IF;  EXCEPTION WHEN dup\_val\_on\_index THEN  null;  END actualizarSaldosSucursales;  /    -- TRIGGER 2  CREATE OR REPLACE TRIGGER actuSaldosSucTipoAho  -- Inicio de la sección declarativa  AFTER UPDATE OF saldo\_ahorro  ON AHORROS  FOR EACH ROW  BEGIN  -- Inicio de la sección ejecutable  IF to\_char(CURRENT\_DATE, 'dd') = '08' AND :NEW.tipo\_ahorro = 2  THEN  IF :NEW.saldo\_interes > 0 THEN  UPDATE TIPO\_AH\_SUC  SET monto\_ahorros = monto\_ahorros + :NEW.saldo\_interes  WHERE COD\_SUCURSAL = :NEW.COD\_SUCURSAL  AND id\_tipo\_ahorro = :NEW.tipo\_ahorro;  ELSE  UPDATE TIPO\_AH\_SUC  SET monto\_ahorros = monto\_ahorros + :NEW.saldo\_ahorro  WHERE COD\_SUCURSAL = :NEW.COD\_SUCURSAL  AND id\_tipo\_ahorro = :NEW.tipo\_ahorro;  END IF;  ELSE  UPDATE TIPO\_AH\_SUC  SET monto\_ahorros = monto\_ahorros + :NEW.saldo\_ahorro  WHERE COD\_SUCURSAL = :NEW.COD\_SUCURSAL  AND id\_tipo\_ahorro = :NEW.tipo\_ahorro;  END IF;  EXCEPTION WHEN dup\_val\_on\_index THEN  null;  END actuSaldosSucTipoAho;  /    --- TRIGGER 3 ----  CREATE OR REPLACE TRIGGER INSERT\_AUDITORIA  AFTER INSERT OR UPDATE  ON AHORROS  FOR EACH ROW  DECLARE  l\_tipo\_transac NUMBER := CASE  WHEN :new.saldo\_ahorro > :old.saldo\_ahorro THEN 1  ELSE 2 END;  BEGIN  IF INSERTING THEN  INSERT INTO AUDITORIA(ID\_AUDITORIA,NO\_CUENTA,ID\_CLIENTE,id\_tipo\_ahorro,TIPO\_OPERACION,TIPO\_TRANSAC,TABLA,saldo\_anterior,monto\_deposito,saldo\_final,USUARIO,fecha\_transaccion)  VALUES(sec\_cod\_aut.nextval,:NEW.NO\_CUENTA,:NEW.id\_cliente,:NEW.tipo\_ahorro,'I',1,'AHORROS',:new.saldo\_ahorro,:new.saldo\_ahorro,:new.saldo\_ahorro,USER,SYSDATE);  END IF;  IF UPDATING THEN  IF l\_tipo\_transac = 1 THEN  INSERT INTO AUDITORIA(ID\_AUDITORIA,NO\_CUENTA,ID\_CLIENTE,id\_tipo\_ahorro,TIPO\_OPERACION,TIPO\_TRANSAC,TABLA,saldo\_anterior,monto\_deposito,saldo\_final,USUARIO,fecha\_transaccion)  VALUES(sec\_cod\_aut.nextval,:NEW.NO\_CUENTA,:NEW.id\_cliente,:NEW.tipo\_ahorro,'U',l\_tipo\_transac,'AHORROS',:old.saldo\_ahorro,:new.saldo\_ahorro + :old.saldo\_ahorro,:new.saldo\_ahorro,USER,SYSDATE);  ELSE  INSERT INTO AUDITORIA(ID\_AUDITORIA,NO\_CUENTA,ID\_CLIENTE,id\_tipo\_ahorro,TIPO\_OPERACION,TIPO\_TRANSAC,TABLA,saldo\_anterior,monto\_deposito,SALDO\_FINAL,USUARIO,fecha\_transaccion)  VALUES (sec\_cod\_aut.nextval,:NEW.NO\_CUENTA,:NEW.id\_cliente,:NEW.tipo\_ahorro,'U',l\_tipo\_transac,'AHORROS',:OLD.SALDO\_AHORRO,:new.saldo\_ahorro - :old.saldo\_ahorro,:new.saldo\_ahorro ,USER,SYSDATE);  END IF;  END IF;  END INSERT\_AUDITORIA;  /    -- INVOCACION PROCEDIMIENTO NUEVOS TIPOS AHORROS  EXECUTE Nuevo\_tipoAhorro('Ahorro de Navidad', 0.06);  EXECUTE Nuevo\_tipoAhorro('Ahorro de Corriente', 0.04);  EXECUTE Nuevo\_tipoAhorro('Ahorro escolar', 0.06);    -- Insertando valores en tipos ah sucursales  BEGIN  INSERT INTO TIPO\_AH\_SUC VALUES(1,1,0,SYSDATE);  INSERT INTO TIPO\_AH\_SUC VALUES(1,2,0,SYSDATE);  INSERT INTO TIPO\_AH\_SUC VALUES(1,3,0,SYSDATE);  INSERT INTO TIPO\_AH\_SUC VALUES(2,1,0,SYSDATE);  INSERT INTO TIPO\_AH\_SUC VALUES(2,2,0,SYSDATE);  INSERT INTO TIPO\_AH\_SUC VALUES(2,3,0,SYSDATE);  INSERT INTO TIPO\_AH\_SUC VALUES(3,1,0,SYSDATE);  INSERT INTO TIPO\_AH\_SUC VALUES(3,2,0,SYSDATE);  INSERT INTO TIPO\_AH\_SUC VALUES(3,3,0,SYSDATE);  INSERT INTO TIPO\_AH\_SUC VALUES(4,1,0,SYSDATE);  INSERT INTO TIPO\_AH\_SUC VALUES(4,2,0,SYSDATE);  INSERT INTO TIPO\_AH\_SUC VALUES(4,3,0,SYSDATE);  END;  /    -- -- INVOCACION PROCEDIMIENTO DE APERTURA DE AHORROS  /\*--PARAMETROS AHORROS  1-id\_cliente number  2-tipo\_ahorro number  3-cod\_sucursal number  4-deposito MENSUAL number  5-fecha deposito=dia,  6-fecha retiro=dia \*/  EXECUTE insertAhorro(1,1,1,10,15,10);  EXECUTE insertAhorro(2,1,2,10,15,10);  EXECUTE insertAhorro(3,2,1,100,15,10);  EXECUTE insertAhorro(4,3,2,200,15,10);  EXECUTE insertAhorro(5,1,2,300,15,10);    -- INVOCACION PROCEDIMIENTO DE INSERCCION DE DEPOSITOS O RETIROS  /\*--PARAMETROS TRANSAC  1-id\_cliente number  2-NO CUENTA number SEC 100 EN 100  3-TIPO AHORRO number  4-SUCURSAL number  5-TIPO TRANSAC (1=DEPO, 2=RETI),  6-MONTO NUMBER \*/  EXECUTE insertTransaDeporeti(1,100,1,1,1,20);  EXECUTE insertTransaDeporeti(2,200,1,1,1,139);  EXECUTE insertTransaDeporeti(3,300,2,1,1,100);  EXECUTE insertTransaDeporeti(3,300,2,1,2,40);  EXECUTE insertTransaDeporeti(4,400,3,2,1,50);  EXECUTE insertTransaDeporeti(5,500,1,2,1,80);  EXECUTE insertTransaDeporeti(5,500,1,2,1,120);  EXECUTE insertTransaDeporeti(5,500,1,2,1,150);  EXECUTE insertTransaDeporeti(1,100,1,1,1,100);    -- INVOCACION PROCEDIMIENTO DE ACTUALIZACION DE DEPOSITOS Y RETIROS  EXECUTE actualizarAhorros;  -- INVOCACION PROCEDIMIENTO DE ACTUALIZACION DE LAS CUENTAS TIPO AHORRO CORRIENTE  EXECUTE calcularInteresDeCorriente; |
| --- |

**Vista de tablas**

Tabla de ahorros

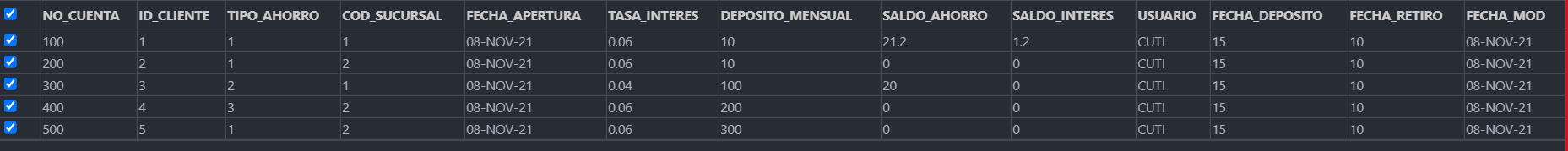


Tabla de transadeporeti

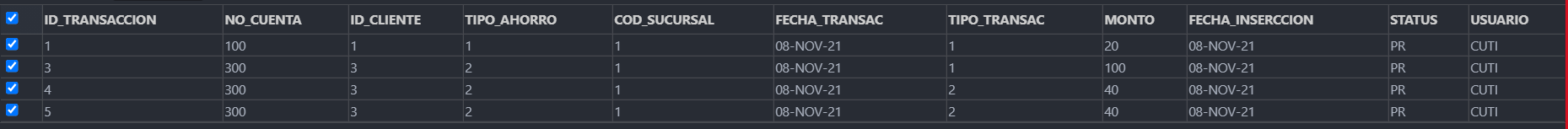


Tabla auditoria



Tabla Sucursales

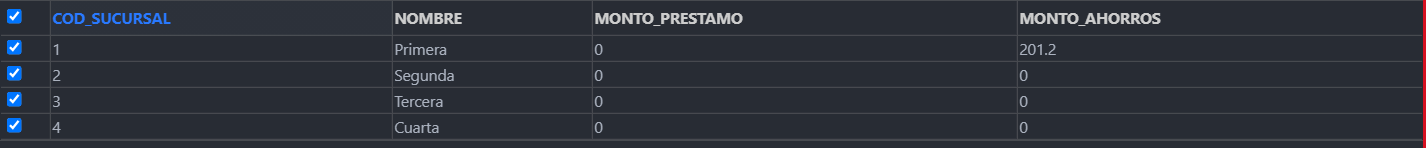


Tabla Tipo\_AH\_SUC



Tabla Clientes

