**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°6**

**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**

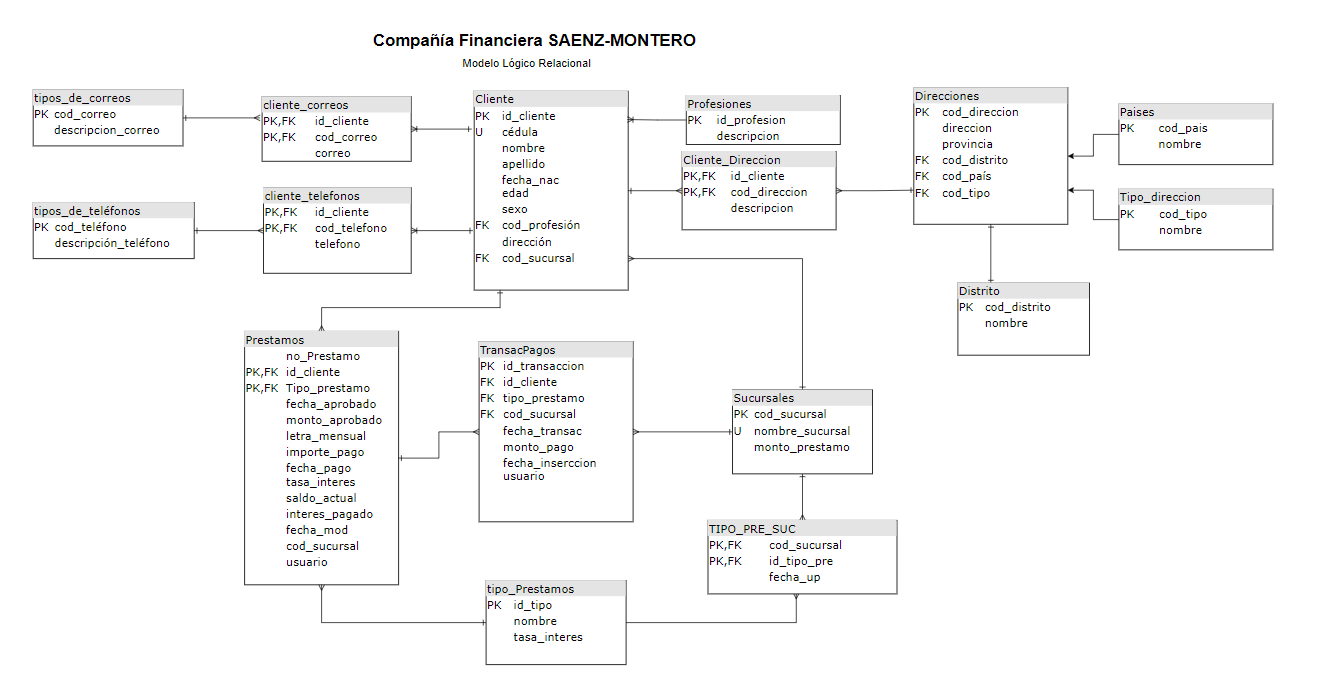
**Caso:**

Tomando en cuenta lo proporcionado por la compañía Financiera SAEZ-MONTERO, S.A del Laboratorio No.2 ‘Una Compañía financiera 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’ I. Realizar las siguientes tareas que afectan el Modelo Físico implementado previamente que permitirá optimizar la sección del proceso de negocio implementado ‘Proceso de Prestamos’ ya que se han aperturado varias sucursales, se han aprobado préstamos y se necesita iniciar con el proceso retorno de la cartera prestada: 1. Agregar el atributo edad a la tabla de clientes proporcionada. 2. Agregar una tabla al modelo físico que almacenes las sucursales de la empresa la financiera con las restricciones correspondientes: Sucursal (cod\_surcursal, nombresucursal, tipoprestamo, montoprestamos) 3. Agregar a la tabla clientes y préstamos el atributo de código de sucursal. 4. Agregar a la tabla del prestamos los siguientes atributos saldoactual, que cuando el préstamo es aprobado es equivalente al monto aprobado, interéspagado que será cargado a partir de los pagos realizados por cliente en proceso de aplicación, fechamodificacion (ddmmyyyy hhmmss), usuario) 5. Agregar una tabla transaccional para recibir los pagos de los clientes con las restricciones correspondiente que contendrá la siguiente atributos:

Transacpagos (cod\_sucursal, id\_transaccion, id\_cliente, tipoprestamo, fechatransaccion, monto del pago, fechainsercion(dd/mm/yyyy, hh:mm:ss, usuario) Todo los campos id que son secuenciales en las tablas de la base de datos deben ser controlados por una secuencia (id\_cliente, numero de prestamos, id\_transaccion) Importante en el Modelo Físico validar la Integridad referencial para los nuevos atributos de ser necesarios y para las nuevas estructuras que formaran parte del Modelo Físico Optimizado. Esto 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 las tablas paramétricas o Tipo de Teléfonos o Tipo de Correos o Profesiones, o Sucursales, o Tipos de prestamos Procedimiento almacenado para la carga o inserción de los clientes con toda la información correspondiente. Este procedimiento debe invocar una Función que calcule la edad de los clientes. Procedimiento almacenado para la carga o inserción de los préstamos aprobados con toda la información correspondiente e igualmente este proceso deberá actualizar la información de préstamo en la tabla de sucursales. Procedimiento almacenado para la carga o inserción de los pagos recibidos de los clientes que se almacena en la tabla transacpagos. Por lo menos uno para cada tipo de préstamos. Procedimiento almacenado que actualice el pago recibo a los préstamos correspondientes. Deberá implementar un cursor que busque los pagos insertados uno a uno y los vaya actualizando en la tabla de préstamos y en la tabla sucursales. Para aplicar el pago este debe rebajarlo del saldo prestamos ( 1000- 20)=980.00. Pero debe tomar en cuenta lo siguiente el préstamo paga interés y este se calcula sobre el saldo del préstamo (saldo del préstamo \* tasa de interés%) el pago interés es mensual (1000 \* 1%) = 50.00.

El cálculo del interés lo realiza una Función que es invocada desde procedimiento. El interés se cobra primero y de quedar alguna porción del monto pagado por el cliente se aplica al saldo del préstamo. La tabla de sucursales solo almacena los monto prestados por la empresa financiera en función de esto, aplicar las actualizaciones. Este laboratorio desarrollo en equipo plazo una semana inicia el 11 de octubre del 2021 Una vez los equipos hayan expuesto el laboratorio el día lunes deberán subirlo a la plataforma Moodle. Sucursal (suc\_codigo varchar2(2) primary key not nul, suc\_descripcion varchar(50)); Tipoprestamos (tip\_codigo varchar(2) primary key not null, tip\_descrip varchar2(50), tasainteres number(15,2)); sucursalTipopres ( suc\_codigo varchar2(2) , tip\_codigo varchar2(2), monto\_prestamos number(15,2), primary key ( suc\_codigo, tip\_codigo)

Modelo



Código

set serveroutput on;

CREATE TABLE tipos\_correos (

cod\_correo NUMBER NOT NULL,

descripcion VARCHAR2(100),

CONSTRAINT correo\_u UNIQUE ( descripcion ),

CONSTRAINT tipos\_correos\_pk PRIMARY KEY ( cod\_correo )

);

CREATE TABLE tipos\_prestamos (

cod\_prestamo NUMBER NOT NULL,

nombre\_prestamo VARCHAR2(100) NOT NULL,

tasa\_interes NUMBER(2, 2) DEFAULT 0,

CONSTRAINT t\_prestam\_u UNIQUE ( nombre\_prestamo ),

CONSTRAINT tipos\_prestamos\_pk PRIMARY KEY ( cod\_prestamo )

);

CREATE TABLE tipos\_telefonos (

cod\_telefono NUMBER NOT NULL,

descripcion VARCHAR2(100),

CONSTRAINT telefonos\_u UNIQUE ( descripcion ),

CONSTRAINT tipos\_telefonos\_pk PRIMARY KEY ( cod\_telefono )

);

CREATE TABLE profesiones (

id\_profesion NUMBER NOT NULL,

descripcion VARCHAR2(100),

CONSTRAINT profesion\_u UNIQUE ( descripcion ),

CONSTRAINT profesion\_pk PRIMARY KEY ( id\_profesion )

);

CREATE TABLE distritos (

cod\_distrito NUMBER NOT NULL,

nombre VARCHAR2(100),

CONSTRAINT distrito\_u UNIQUE ( NOMBRE ),

CONSTRAINT distritos\_pk PRIMARY KEY ( cod\_distrito )

);

CREATE TABLE provincias (

cod\_provincia NUMBER NOT NULL,

nombre VARCHAR2(100),

CONSTRAINT provincia\_u UNIQUE ( nombre ),

CONSTRAINT provincias\_pk PRIMARY KEY ( cod\_provincia )

);

CREATE TABLE provincias\_distritos (

cod\_provincia NUMBER NOT NULL,

cod\_distrito NUMBER NOT NULL,

CONSTRAINT provincias\_distritos\_pk PRIMARY KEY ( cod\_provincia,cod\_distrito ),

CONSTRAINT distritos\_fk FOREIGN KEY ( cod\_distrito )

REFERENCES distritos ( cod\_distrito ),

CONSTRAINT provincias\_fk FOREIGN KEY ( cod\_provincia )

REFERENCES provincias ( cod\_provincia )

);

CREATE TABLE sucursales (

cod\_sucursal NUMBER NOT NULL,

nombre VARCHAR2(50) NOT NULL,

monto\_prestamo NUMBER(15, 2) DEFAULT 0,

CONSTRAINT sucursales\_pk PRIMARY KEY ( cod\_sucursal ),

CONSTRAINT sucursales\_un UNIQUE ( nombre )

);

CREATE TABLE clientes (

id\_cliente NUMBER NOT NULL,

cedula VARCHAR2(10) NOT NULL,

nombre1 VARCHAR2(100) not NULL,

apellido1 VARCHAR2(100) not NULL,

fecha\_nac DATE not NULL,

edad NUMBER(3),

sexo CHAR NOT NULL,

cod\_profesion NUMBER NOT NULL,

direccion VARCHAR2(250) not NULL,

cod\_sucursal NUMBER NOT NULL,

constraint c\_sexo CHECK (sexo in ('F','M')),

CONSTRAINT clientes\_un UNIQUE ( cedula ),

CONSTRAINT clientes\_pk PRIMARY KEY ( id\_cliente ),

CONSTRAINT clientes\_profesion\_fk FOREIGN KEY ( cod\_profesion )

REFERENCES profesiones ( id\_profesion ),

CONSTRAINT clientes\_sucursales\_fk FOREIGN KEY ( cod\_sucursal )

REFERENCES sucursales ( cod\_sucursal )

);

CREATE TABLE clientes\_correos (

id\_cliente NUMBER NOT NULL,

id\_correo NUMBER NOT NULL,

correo VARCHAR2(100),

CONSTRAINT clientes\_correos\_pk PRIMARY KEY ( id\_cliente,id\_correo ),

CONSTRAINT clientes\_fk FOREIGN KEY ( id\_cliente )

REFERENCES clientes ( id\_cliente ),

CONSTRAINT tipos\_correos\_fk FOREIGN KEY ( id\_correo )

REFERENCES tipos\_correos ( cod\_correo )

);

CREATE TABLE clientes\_telefonos (

id\_cliente NUMBER NOT NULL,

id\_telefono NUMBER NOT NULL,

telefono VARCHAR2(10),

CONSTRAINT clientes\_telefonos\_pk PRIMARY KEY ( id\_cliente,id\_telefono ),

CONSTRAINT clientes\_telefonos\_fk FOREIGN KEY ( id\_cliente )

REFERENCES clientes ( id\_cliente ),

CONSTRAINT clientes\_tipos\_telefonos\_fk FOREIGN KEY ( id\_telefono )

REFERENCES tipos\_telefonos ( cod\_telefono )

);

CREATE TABLE tipos\_pre\_sucursal (

cod\_sucursal NUMBER NOT NULL,

cod\_t\_prestam NUMBER NOT NULL,

monto\_presta NUMBER NOT NULL,

fecha\_mod DATE,

CONSTRAINT tipos\_pre\_sucursal\_pk PRIMARY KEY ( cod\_sucursal,cod\_t\_prestam ),

CONSTRAINT tipos\_prestamos\_fk FOREIGN KEY ( cod\_t\_prestam )

REFERENCES tipos\_prestamos ( cod\_prestamo ),

CONSTRAINT tipos\_sucursales\_fk FOREIGN KEY ( cod\_sucursal )

REFERENCES sucursales ( cod\_sucursal )

);

CREATE TABLE prestamos (

no\_prestamo NUMBER NOT NULL,

id\_cliente NUMBER NOT NULL,

cod\_tipo\_prestamo NUMBER NOT NULL,

fecha\_aprobado DATE,

monto\_aprobado NUMBER(15,2) DEFAULT 0,

letra\_mensual NUMBER(15,2) DEFAULT 0,

importe\_pago NUMBER(15,2) DEFAULT 0,

fecha\_pago DATE,

tasa\_interes NUMBER(2, 2) DEFAULT 0,

saldo\_acual NUMBER(15, 2) DEFAULT 0,

interes\_pagado NUMBER(15, 2) DEFAULT 0,

fecha\_mod date,

cod\_sucursal number,

usuario varchar2(50),

CONSTRAINT prestamos\_pk PRIMARY KEY ( id\_cliente,cod\_tipo\_prestamo ),

CONSTRAINT prestamos\_clientes\_fk FOREIGN KEY ( id\_cliente )

REFERENCES clientes ( id\_cliente ),

CONSTRAINT tipos\_presta\_fk FOREIGN KEY ( cod\_tipo\_prestamo )

REFERENCES tipos\_prestamos ( cod\_prestamo )

);

CREATE TABLE transacpagos (

id\_transaccion NUMBER NOT NULL,

id\_cliente NUMBER NOT NULL,

tipo\_prestamo NUMBER NOT NULL,

cod\_sucursal NUMBER NOT NULL,

fecha\_transac DATE,

monto\_pago NUMBER(15, 2) DEFAULT 0,

fecha\_inserccion DATE,

usuario VARCHAR2(45),

status char(2);

*-- PAGADO, EN PROCESO, NO PAGADO*

constraint status\_check CHECK(status in('P','EP', 'NP')),

CONSTRAINT transacpagos\_pk PRIMARY KEY ( id\_transaccion ),

CONSTRAINT transacpagos\_prestamos\_fk FOREIGN KEY ( id\_cliente,tipo\_prestamo )

REFERENCES prestamos ( id\_cliente,cod\_tipo\_prestamo ),

CONSTRAINT transac\_sucursales\_fk FOREIGN KEY ( cod\_sucursal )

REFERENCES sucursales ( cod\_sucursal )

);

*---VISTA DE LOS PRESTAMOS--*

CREATE VIEW VIEW\_PRESTAMOS

AS select pe.NO\_PRESTAMO as "NO. PRESTAMO",c.CEDULA as "CEDULA",

c.NOMBRE1 as "NOMBRE",

c.APELLIDO1 as "APELLIDO",

tp.NOMBRE\_PRESTAMO as "TIPO DEPRESTAMO",

pe.MONTO\_APROBADO as "MONTO APROBADO",

pe.LETRA\_MENSUAL as "LETRA MENSUAL",

p.DESCRIPCION as "PROFESION"

from PROFESIONES p,

PRESTAMOS pe,

TIPOS\_PRESTAMOS tp,

CLIENTES c

where c.id\_cliente = pe.id\_cliente and pe.TIPO\_PRESTAMO = tp.COD\_PRESTAMO and c.COD\_PROFESION = p.ID\_PROFESION

order by c.CEDULA ASC;

*--TIPOS CORREOS*

CREATE or REPLACE PROCEDURE Nuevo\_tipoCorreo(

p\_Correo IN tipos\_correos.descripcion%TYPE)

IS

intSeqVal number(10);

BEGIN

select sec\_cod\_correo.nextval into intSeqVal from dual;

INSERT into TIPOS\_CORREOS (cod\_correo,descripcion)

VALUES (intSeqVal,p\_Correo);

COMMIT;

EXCEPTION

WHEN DUP\_VAL\_ON\_INDEX THEN

DBMS\_OUTPUT.PUT\_LINE('💣 Error: El tipo de correo ya existe.');

END Nuevo\_tipoCorreo;

/

*--TIPOS PRESTAMOS*

CREATE or REPLACE PROCEDURE Nuevo\_tipoPrestamo(

p\_prestam IN tipos\_prestamos.nombre\_prestamo%TYPE,

p\_interes IN TIPOS\_PRESTAMOS.TASA\_INTERES%TYPE)

IS

intSeqVal number(10);

BEGIN

select sec\_cod\_prestamo.nextval into intSeqVal from dual;

INSERT into TIPOS\_PRESTAMOS (cod\_prestamo,nombre\_prestamo,tasa\_interes)

VALUES (intSeqVal,p\_prestam,p\_interes);

COMMIT;

EXCEPTION

WHEN DUP\_VAL\_ON\_INDEX THEN

DBMS\_OUTPUT.PUT\_LINE('💣 Error: El tipo de prestamo ya existe.');

END Nuevo\_tipoPrestamo;

/

*--TIPOS telefonos*

CREATE or REPLACE PROCEDURE Nuevo\_tipotelefonos(

p\_telefonos IN tipos\_telefonos.descripcion%TYPE)

IS

intSeqVal number(10);

BEGIN

select sec\_cod\_telefono.nextval into intSeqVal from dual;

INSERT into TIPOS\_TELEFONOS (cod\_telefono,descripcion)

VALUES (intSeqVal,p\_telefonos);

COMMIT;

EXCEPTION

WHEN DUP\_VAL\_ON\_INDEX THEN

DBMS\_OUTPUT.PUT\_LINE('💣 Error: El tipo de telefono ya existe.');

END Nuevo\_tipotelefonos;

/

*--TIPOS profesion*

CREATE or REPLACE PROCEDURE Nuevo\_tipoprofesion(

p\_profesion IN profesiones.descripcion%TYPE)

IS

intSeqVal number(10);

BEGIN

select sec\_id\_profesion.nextval into intSeqVal from dual;

INSERT into PROFESIONES(id\_profesion,descripcion)

VALUES (intSeqVal,p\_profesion);

COMMIT;

EXCEPTION

WHEN DUP\_VAL\_ON\_INDEX THEN

DBMS\_OUTPUT.PUT\_LINE('💣 Error: La profesion ya existe.');

END Nuevo\_tipoprofesion;

/

*--Distritos*

CREATE or REPLACE PROCEDURE NuevoDistrito(

p\_distrito IN distritos.nombre%TYPE)

IS

intSeqVal number(10);

BEGIN

select sec\_cod\_distrito.nextval into intSeqVal from dual;

INSERT into DISTRITOS(cod\_distrito,nombre)

VALUES (intSeqVal,p\_distrito);

COMMIT;

EXCEPTION

WHEN DUP\_VAL\_ON\_INDEX THEN

DBMS\_OUTPUT.PUT\_LINE('💣 Error: El distrito ya existe.');

END NuevoDistrito;

/

*--Provincias*

CREATE or REPLACE PROCEDURE NuevaProvincia(

p\_provincia IN provincias.nombre%TYPE)

IS

intSeqVal number(10);

BEGIN

select sec\_cod\_provincia.nextval into intSeqVal from dual;

INSERT into PROVINCIAS(cod\_provincia,nombre)

VALUES (intSeqVal,p\_provincia);

COMMIT;

EXCEPTION

WHEN DUP\_VAL\_ON\_INDEX THEN

DBMS\_OUTPUT.PUT\_LINE('💣 Error: La provincia ya existe.');

END NuevaProvincia;

/

*--sucursales*

CREATE or REPLACE PROCEDURE NuevaSucursal(

p\_sucursal IN SUCURSALES.nombre%TYPE)

IS

intSeqVal number(10);

v\_sucursal VARCHAR2(100) := p\_sucursal;

v\_monto number := 0;

BEGIN

select sec\_cod\_sucursal.nextval into intSeqVal from dual;

INSERT into SUCURSALES(COD\_SUCURSAL,nombre,MONTO\_PRESTAMO)

VALUES (

intSeqVal,

v\_sucursal,

v\_monto);

FOR v\_counter IN 1..5 LOOP

INSERT INTO TIPOS\_PRE\_SUCURSAL(

COD\_SUCURSAL,

COD\_T\_PRESTAM,

monto\_prestamo,

fecha\_mod)

VALUES(

intSeqVal,

v\_counter,

v\_monto,

to\_date(sysdate,'DD-MM-YY')

);

COMMIT;

END LOOP;

EXCEPTION

WHEN DUP\_VAL\_ON\_INDEX THEN

DBMS\_OUTPUT.PUT\_LINE('💣 Error: La sucursal ya existe.');

END NuevaSucursal;

/

CREATE OR REPLACE FUNCTION calcularEdadCliente(p\_fecha date)

RETURN NUMBER IS

v\_clienteEdad number(3);

v\_fecha date := p\_fecha;

BEGIN

*-- Necesitamos eso en años*

v\_clienteEdad := (SYSDATE - v\_fecha) / 365;

RETURN v\_clienteEdad;

EXCEPTION

WHEN ZERO\_DIVIDE THEN

DBMS\_OUTPUT.PUT\_LINE('💣 Error: El tipo de correo ya existe.');

END calcularEdadCliente;

/

CREATE OR REPLACE PROCEDURE insertCliente(

p\_cedula IN clientes.cedula%TYPE,

p\_Nombre IN clientes.nombre1%TYPE,

p\_Apellido IN clientes.apellido1%TYPE,

p\_fecha IN clientes.fecha\_nac%TYPE,

p\_sexo IN clientes.SEXO%TYPE,

p\_profesion IN clientes.cod\_profesion%TYPE,

p\_direccion IN clientes.direccion%TYPE,

p\_sucursal IN clientes.cod\_sucursal%TYPE)

IS

intSeqVal number(10);

v\_edad number(3) := calcularEdadCliente(p\_fecha);

BEGIN

select SEC\_ID\_cliente.nextval into intSeqVal from dual;

INSERT into CLIENTES (id\_cliente,

cedula,

nombre1,

apellido1,

fecha\_nac,

edad,

sexo,

cod\_profesion,

direccion,

cod\_sucursal);

VALUES (intSeqVal,

p\_cedula,

p\_nombre,

p\_apellido,

to\_date(p\_fecha,'DD-MON-YY'),

v\_edad,

p\_sexo,

p\_profesion,

p\_direccion,

p\_sucursal);

COMMIT;

EXCEPTION

WHEN DUP\_VAL\_ON\_INDEX THEN

DBMS\_OUTPUT.PUT\_LINE('💣 Error: El cliente ya existe.');

END insertCliente;

/

CREATE OR REPLACE PROCEDURE insertPrestamo(

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

p\_cod\_tipo\_prestamo IN prestamos.cod\_tipo\_prestamo%TYPE,

p\_monto\_aprobado IN prestamos.monto\_aprobado%TYPE,

p\_fecha\_pago IN prestamos.fecha\_pago%TYPE,

p\_cuotas NUMBER;

p\_no\_sucursal number;

)

IS

v\_cod\_prestamo NUMBER := p\_cod\_tipo\_prestamo;

intSeqVal number(10);

v\_fecha date := SYSDATE;

v\_saldo := p\_monto\_aprobado;

v\_moto\_prestamo number := v\_monto\_prestamo+v\_saldo;

v\_interes NUMBER;

v\_importe number := 0;

BEGIN

*--1--*

select sec\_no\_prestamo.nextval into intSeqVal from dual;

SELECT tasa\_interes INTO v\_interes FROM TIPOS\_PRESTAMOS WHERE cod\_prestamo = v\_cod\_prestamo;

*--2*

INSERT INTO PRESTAMOS(

no\_prestamo,

id\_cliente,

cod\_tipo\_prestamo,

fecha\_aprobado,

monto\_aprobado,

letra\_mensual,

importe\_pago,

fecha\_pago,

tasa\_interes,

saldo\_actual,

interes\_pagado,

fecha\_mod,

cod\_sucursal,

usuario);

VALUES (intSeqVal,

p\_no\_prestamo,

p\_id\_cliente,

p\_cod\_tipo\_prestamo,

to\_date(v\_fecha,'DD-MM-YYY HH:MI:SS')

p\_monto\_aprobado,

p\_letra\_mensual,

v\_importe,

to\_date(p\_fecha\_pago, 'DD-MM-YYY'),

v\_interes,

saldo,

p\_interes\_pagado,

to\_date(v\_fecha,'DD-MM-YYY HH:MI:SS'),

p\_no\_sucursal,

user,

);

*--3*

*--ACTUALIZACION DE LA TABLA SUCURSALES: MONTOS*

SELECT monto\_prestamo INTO v\_monto\_prestamo

FROM TIPOS\_PRE\_SUCURSAL

WHERE cod\_sucursal = p\_no\_sucursal and cod\_t\_prestam = p\_cod\_tipo\_prestamo;

*--Actualizacion de la tabla relacion muchos a muchos de TIPO PRESTAMO Y SUCURSAL*

UPDATE SET MONTO\_PRESTAMO=v\_monto\_prestamo+v\_monto\_prestamos FROM TIPOS\_PRE\_SUCURSAL;

COMMIT;

EXCEPTION

WHEN DUP\_VAL\_ON\_INDEX THEN

DBMS\_OUTPUT.PUT\_LINE('💣 Error: El cliente ya existe.');

END insertPrestamo;

/

CREATE OR REPLACE PROCEDURE insertPagos(

p\_id\_transaccion IN transacpagos.id\_transaccion %TYPE,

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

p\_tipo\_prestamo IN transacpagos.tipo\_prestamo %TYPE,

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

p\_fecha\_transac IN transacpagos.fecha\_transac %TYPE,

p\_monto\_pago IN transacpagos.monto\_pago %TYPE

)

IS

intSeqVal number(10);

BEGIN

select sec\_id\_transac.nextval into intSeqVal from dual;

INSERT INTO transacpagos(

id\_transaccion,

id\_cliente,

tipo\_prestamo,

cod\_sucursal,

fecha\_transac,

monto\_pago,

fecha\_inserccion,

usuario

);

VALUES(

intSeqVal,

id\_transaccion,

id\_cliente,

tipo\_prestamo,

cod\_sucursal,

SYSDATE,

monto\_pago,

SYSDATE,

USER);

COMMIT;

EXCEPTION

WHEN DUP\_VAL\_ON\_INDEX THEN

DBMS\_OUTPUT.PUT\_LINE('💣 Error: El cliente ya existe.');

END insertPagos;

/

*-- Función de calcular interés*

CREATE OR REPLACE FUNCTION calcularInteres(

p\_prestamo PRESTAMOS.monto\_aprobado%TYPE,

p\_interes PRESTAMOS.tasa\_interes%TYPE,

)

RETURN NUMBER IS

V\_interes\_calculado NUMBER;

v\_prestamo NUMBER := p\_prestamo;

v\_interes NUMBER := p\_interes;

BEGIN

*-- Interes calculado mediante el préstamo e interes*

V\_interes\_calculado := (v\_prestamo \* v\_interes) ;

RETURN V\_interes\_calculado;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('💣 Error: El préstamo no ha sido encontrado.');

END calcularInteres;

/

*-- Función de disminuir préstamo*

CREATE OR REPLACE FUNCTION disminuirPrestamo(

p\_monto\_mensual IN PRESTAMOS.monto\_aprobado%TYPE,

p\_monto\_interes IN PRESTAMOS.interes\_pagado%TYPE,

p\_monto\_a\_pagar IN PRESTAMOS.importe\_pago%TYPE,

)

RETURN NUMBER IS

V\_monto\_actual NUMBER;

v\_monto\_mensual NUMBER := p\_monto\_mensual;

v\_monto\_a\_pagar NUMBER := p\_monto\_a\_pagar;

v\_monto\_interes NUMBER := p\_monto\_interes;

;

BEGIN

*-- Interes calculado mediante el préstamo e interes*

IF v\_monto\_a\_pagar - (v\_monto\_interes + v\_monto\_mensual) >= 0

V\_monto\_actual = v\_monto\_a\_pagar - (v\_monto\_interes + v\_monto\_mensual)

ELSE

V\_monto\_actual = v\_monto\_a\_pagar - v\_monto\_interes

RETURN V\_monto\_actual

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('💣 Error: El monto insertado no ha sido calculado.');

END disminuirPrestamo;

/

CREATE OR REPLACE PROCEDURE insertUpdate(

p\_monto\_prestamo IN SUCURSALES.monto\_prestamo%TYPE;

p\_saldo\_actual IN PRESTAMOS.saldo\_actual&TYPE;

)

BEGIN

*-- Aqui se implementaría el cursor*

CURSOR Pagos IS

SELECT saldo\_actual, monto\_prestamo

FROM p PRESTAMOS, s SUCURSALES

*-- Aqui iria la logica del update*

END;

/