UNIVERSIDAD TECNOLÓGICA DE PANAMÁ

FACULTAD DE INGENIERÍA EN SISTEMAS COMPUTACIONALES

LIC. ING. EN SISTEMAS DE INFORMACIÓN

LABORATORIO 5

CURSO: SISTEMA BASE DE DATOS 2

FACILITADORA: 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 5

La Compañía Financiera Márquez-Cedeño, 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 clientes y los préstamos e información relacionada a esta sección del proceso a dicha base de datos.

Cuenta con unas relaciones o tablas de bases de datos que han sido identificadas y no están normalizadas que pueden ser utilizada por los Ingenieros de Sistemas de Información para futuras implementación.

La relación Cliente está compuesta por identificación del cliente, cedula, nombre, apellido, sexo, emial (personal, laboral, académico), teléfono (celular, residencia, celular del familiar más cercano, celular del conyugue), profesión (cualquier profesión del cliente), fecha de nacimiento.

La relación Préstamo está compuesta por identificación del cliente, tipo de préstamo (personal, auto, hipoteca, garantizado con ahorros), número de préstamos, fecha de aprobado, monto aprobado, tasa de interés, letra mensual, monto pagado, fecha de pago. Los clientes pueden tener varios tipos de préstamos en la institución financiera.

Modelo lógico

TABLA CLIENTE		
PK	ID_cliente	N
U	Cedula	N
	Nombre	N
	Apellido	N
	Sexo	S
	Fecha_nac	S

TABLA PRESTAMO		
PK	No_prestamo	N
	Tipo_prestamo	N
	Fecha_aprob	N
	Monto_aprob	N
	Interes	N
	Letra_mensual	N
	Monto_pago	N
	Fecha_pago	N

TABLA PRESTAMO_CLIENTE		
FK	ID_cliente	N
FK	No_prestamo	N

Modelo lógico Normalizado

PROFESION		
PK	ID_Profesion	N
	Desc_Prof	N

BANCO_CLIENTES		
PK	ID_Cliente	N
U	Cedula	N
	Nombre	N
	Apellido	N
	Sexo	S
	Fecha_nac	S
FK	ID_Profesion	N

TIPO_TELEFONO		
PK	Codigo_Telf	N
	Desc_Telf	N

TELEFONO_CLIENTE		
FK(PK)	ID_cliente	N
FK(PK)	Codigo_Telf	N
	Telefono	S

TIPO_EMAIL		
PK	Codigo_Email	N
	Desc_Email	N

EMAIL_CLIENTE		
FK(PK)	ID_cliente	N
FK(PK)	Cod_Email	N
	Email	S

PRESTAMO_TIPO		
PK	Cod_Prestamo	N
	Nombre_Prestamo	N
	Tasas_Interes	N

CLIENTES_PRESTAMO		
PK	Cod_Prestamo	N
PK	ID_cliente	N
	Numero_Prestamo	N
	Fecha_aprob	N
	Monto_aprob	N
	Letra_mensual	N
FK	Cod_Prestamo	
FK	ID_cliente	

Scrip de Modelo Físico (En orden de creacion)

- Profesion/Banco_Cliente(Dependiente)

```
SQL> --Creacion de la tabla PROFESION
SQL> Create table Profesion (
  2  ID_Profesion varchar2(10) not null primary key,
  3  Desc_Prof varchar2(50) not null
  4   );

Table created.

SQL> --Creación de tabla Cliente
SQL> Create Table Banco_Clientes (
  2   ID_Cliente varchar2(10) not null primary key,
  3   Cedula varchar2(15) not null Unique,
  4   Nombre varchar2(15) not null,
  5   Apellido varchar2(15) not null,
  6   Sexo varchar2(20) DEFAULT 'Sin especificar',
  7  Fecha_nac date,
  8   ID_Profesion varchar2(10) not null,
  9   Constraint Prof_Cliente foreign key (ID_Profesion) references Profesion (ID_Profesion)
 10      );

Table created.
```

CREATE INDEX IDX_ID_CLIENT ON Banco_Clientes ();

- Tipo_Telefono/TelefonoCliente(Dependiente)

```
SQL> --Creacion de la tabla TIPO_TELEFONO
SQL> Create table Tipo_Telefono (
  2  Cod_Telf varchar2(5) not null primary key,
  3  Desc_Telf varchar2(30) not null
  4   );

Table created.

SQL>
SQL> --Creacion de la tabla TELEFONO_CLIENTE
SQL> Create table Telefono_Cliente (
  2  Telefono varchar2(30) not null,
  3  ID_Cliente,
  4  Constraint identif_cliente foreign key (ID_Cliente) references Banco_Clientes (ID_Cliente),
  5  Codigo_Telf,
  6  Constraint cod_Telef foreign key (Codigo_Telf) references Tipo_Telefono (Cod_Telf),
  7  Constraint Telefonos_Clientes primary key (ID_Cliente, Codigo_Telf)
  8  );

Table created.
```

- Tipo_Email/ Email _Cliente(Dependiente)

```
SQL> --Creacion de la tabla TIPO_EMAIL
SQL> Create table Tipo_Email (
  2  Cod_Email varchar2(10) not null primary key,
  3  Desc_Email varchar2(50) not null
  4   );

Table created.

SQL>
SQL> --Creacion de la tabla EMAIL_CLIENTE
SQL> Create table Email_Cliente (
  2  Email varchar2(30) not null,
  3  ID_Cliente,
  4  Constraint identif_cliente_email foreign key (ID_Cliente) references Banco_Clientes (ID_Cliente),
  5  Codigo_Email,
  6  Constraint cod_Telef_email foreign key (Codigo_Email) references Tipo_Email (Cod_Email),
  7  Constraint Emails_Clientes primary key (ID_Cliente, Codigo_Email)
  8  );

Table created.
```

- Prestamo_Tipo/Clientes_Prestamo

```
SQL> --Creacion de la tabla Prestamo_Tipo
SQL>  Create table Prestamo_Tipo (
  2  Cod_Prestamo varchar2(15) not null primary key,
  3   Nombre_prestamo varchar2(50) not null,
  4   Tasas_Interes number not null
  5  );

Table created.

SQL>
SQL>
SQL>
SQL> --Creacion de la tabla Prestamos
SQL>  Create table Clientes_Prestamo (
  2   Numero_prestamo varchar2(10) not null ,
  3   Fecha_aprob date not null,
  4   Monto_aprob number,
  5   Letra_mensual number not null,
  6  Cod_Prestamo,
  7  Constraint codigo_prestamo foreign key (Cod_Prestamo) references Prestamo_Tipo(Cod_Prestamo),
  8  ID_cliente,
  9  Constraint identif_cliente_prestamo foreign key (ID_Cliente) references Banco_Clientes (ID_Cliente),
 10  Constraint Presta_Cliente primary key (Cod_Prestamo , ID_Cliente)
 11   );

Table created.
```

```
SQL> CREATE INDEX IDX_NUM_PRES ON Clientes_Prestamo (Numero_prestamo);

Index created.
```

UNIVERSIDAD TECNOLÓGICA DE PANAMÁ

FACULTAD DE INGENIERÍA EN SISTEMAS COMPUTACIONALES

LIC. ING. EN SISTEMAS DE INFORMACIÓN

LABORATORIO 6

CURSO: SISTEMA BASE DE DATOS 2

FACILITADORA: 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

Agregar el atributo edad a la tabla de clientes proporcionada:

Alter table Banco_Clientes

add (edad_cl number not null);

```
SQL> Alter table Banco_Clientes
  2  add (edad_cl number not null);

Table altered.
```

Sucursal		
PK	Cod_Sucursal	N
	Nom_Sucursal	N

Suc_Tipo_Prestamo		
FK(PK)	Cod_Sucursal	N
FK(PK)	Cod_Prestamo	N
	Monto_Prestamo	N

```
SQL> Create table Sucursal (
  2  Cod_Sucursal varchar2(5)  primary key not null,
  3  Nom_Sucursal varchar2(25)
  4  );

Table created.
```

Create table Suc_Tipo_Prestamo (

Cod_Sucursal varchar2(5),

Constraint Fk_cod_suc foreign key (Cod_Sucursal) references Sucursal (Cod_Sucursal),

Cod_Prestamo varchar2(15),

Constraint Fk_cod_prestamo foreign key (Cod_Prestamo) references Prestamo_Tipo (Cod_Prestamo),

Monto_Prestamos number(15,2),

Constraint pk_tipo_suc_prest primary key (Cod_Sucursal, Cod_Prestamo)

);

```
SQL> Create table  Suc_Tipo_Prestamo (
  2  Cod_Sucursal varchar2(5),
  3  Constraint Fk_cod_suc foreign key (Cod_Sucursal) references Sucursal (Cod_Sucursal),
  4  Cod_Prestamo varchar2(15),
  5  Constraint Fk_cod_prestamo foreign key (Cod_Prestamo) references Prestamo_Tipo (Cod_Prestamo),
  6  Monto_Prestamos number(15,2),
  7  Constraint pk_tipo_suc_prest primary key (Cod_Sucursal, Cod_Prestamo)
  8  );

Table created.
```

- **MODIFICACIONES A LAS TABLAS**

BANCO_CLIENTES		
PK	ID_Cliente	N

U	Cedula	N
	Nombre	N
	Apellido	N
	Sexo	S
	Fecha_nac	S
FK	ID_Profesion	N
FK	Cod_Sucursal	N

- Atributos agregados a la tabla clientes

Alter table Banco_Clientes add Cod_Sucursal varchar2(5) not null;

```
SQL> Alter table Banco_Clientes add Cod_Sucursal varchar2(5) not null;

Table altered.
```

Alter table Banco_Clientes add constraint FK_clientes_Suc foreign key (Cod_Sucursal) references Sucursal (Cod_Sucursal);

```
SQL> Alter table Banco_Clientes add constraint FK_clientes_Suc foreign key (Cod_Sucursal) references Sucursal (Cod_Sucur
sal);

Table altered.
```

Create sequence seq_IdClient start with 1 increment by 1;

CLIENTES_PRESTAMO		
PK	Cod_Prestamo	N
PK	ID_cliente	N
	Numero_Prestamo	N
	Fecha_aprob	N
	Monto_aprob	N
	Letra_mensual	N
	Saldo_Actual	N
	Interes_Pagado	N
	Fecha_Modif	N
	Usuario	N
FK	Cod_Prestamo	N
FK	ID_cliente	N
FK	Cod_Sucursal	N

- Atributos agregados a la tabla prestamos

Alter table Clientes_Prestamo add Cod_Sucursal varchar2(5) not null;

```
SQL> Alter table Clientes_Prestamo add Cod_Sucursal varchar2(5) not null;

Table altered.
```

Alter table Clientes_Prestamo add constraint FK_clientes_Suc_pres foreign key (Cod_Sucursal) references Sucursal (Cod_Sucursal);

```
SQL> Alter table Clientes_Prestamo add constraint FK_clientes_Suc_pres foreign key (Cod_Sucursal) references Sucursal (C
od_Sucursal);

Table altered.
```

Create sequence seq_NumPres start with 1 increment by 1;

Alter table Clientes_Prestamo add (

Saldo_Actual number not null,

Interes_Pagado number not null,

Fecha_Modif date not null,

Usuario number

);

```
SQL> Alter table Clientes_Prestamo add (
  2  Saldo_Actual number not null,
  3  Interes_Pagado number not null,
  4  Fecha_Modif date not null,
  5  Usuario number
  6  );

Table altered.
```

Alter table Clientes_Prestamo add(monto_pago number);

Alter table Clientes_Prestamo add(edad_cl date);

- Tabla transaccional

Transac_pagos		
PK	ID_Transaccion	N
	Monto_pago	N
	Fecha_transac	N
	Fecha_inserc	N
	Usuario	N
FK	Cod_sucursal	
FK	ID_cliente	

Create table Transac_pagos (

ID_Transaccion number primary key not null,

Cod_Sucursal varchar2(5) not null,

Monto_pago number not null,

Fecha_transac date not null,

Fecha_inserc date not null,

Usuario number not null,

Constraint FK_CodSuc_TransP foreign key (Cod_Sucursal) references Sucursal (Cod_Sucursal),

ID_Cliente number,

Cod_Prestamo varchar2(15),

Constraint FK_IDClient_TPrest_TransP foreign key (ID_Cliente, Cod_Prestamo) references Clientes_Prestamo (ID_Cliente, Cod_Prestamo)

);

Create sequence seq_IDTrans start with 1 increment by 1;

```
SQL> Create table Transac_pagos (
  2  ID_Transaccion varchar2(10) primary key not null,
  3  Cod_Sucursal varchar2(5) not null,
  4  Monto_pago number not null,
  5  Fecha_transac date not null,
  6  Fecha_inserc date not null,
  7  Usuario number not null,
  8  Constraint FK_CodSuc_TransP foreign key (Cod_Sucursal) references Sucursal (Cod_Sucursal),
  9  ID_Cliente varchar2(10),
 10  Cod_Prestamo varchar2(15),
 11  Constraint FK_IDClient_TPrest_TransP foreign key (ID_Cliente, Cod_Prestamo) references Clientes_Prestamo (ID_Client
e, Cod_Prestamo)
 12  );

Table created.
```

- Correcion de los identificadores.

```
SQL> ALTER TABLE Clientes_Prestamo MODIFY Numero_Prestamo number;

Table altered.
```

ALTER TABLE Clientes_Prestamo MODIFY Numero_Prestamo number;

```
SQL> ALTER TABLE Transac_pagos MODIFY ID_Transaccion number;

Table altered.
```

ALTER TABLE Transac_pagos MODIFY ID_Transaccion number;

```
SQL> ALTER TABLE Email_Cliente MODIFY ID_Cliente number;

Table altered.
```

ALTER TABLE Email_Cliente MODIFY ID_Cliente number;

```
SQL> ALTER TABLE Telefono_Cliente MODIFY Codigo_Telf varchar2(5);

Table altered.

SQL> ALTER TABLE Email_Cliente MODIFY Codigo_Email varchar2(10);

Table altered.
```

ALTER TABLE Telefono_Cliente MODIFY Codigo_Telf varchar2(5);

ALTER TABLE Email_Cliente MODIFY Codigo_Email varchar2(10);

CREACION DE PROCEDIMIENTOS

- Tipo telefono

```
Create or Replace Procedure Tipo_tel(
  p_ID_tel Tipo_Telefono.Cod_Telf%TYPE,
  p_desc_tel Tipo_Telefono.Desc_Telf%TYPE
) AS
BEGIN
INSERT INTO Tipo_Telefono (Cod_Telf, Desc_Telf) VALUES (p_ID_tel, p_desc_tel);
Exception
When others then
DBMS_OUTPUT.PUT_LINE( 'REGISTRO NO CREADO');
COMMIT;
END Tipo_tel;
/
------------------------------
BEGIN
 Tipo_tel ( 'R01', 'Residencial');
Tipo_tel ('P01', 'Personal');
Tipo_tel ( 'C01', 'Otro contacto');
END;
/
```

```
SQL> Create or Replace Procedure Tipo_tel(
  2    p_ID_tel Tipo_Telefono.Cod_Telf%TYPE,
  3    p_desc_tel Tipo_Telefono.Desc_Telf%TYPE
  4  ) AS
  5  BEGIN
  6  INSERT INTO Tipo_Telefono (Cod_Telf, Desc_Telf) VALUES (p_ID_tel, p_desc_tel);
  7  COMMIT;
  8  END Tipo_tel;
  9  /

Procedure created.

SQL> BEGIN
  2   Tipo_tel ( 'R01', 'Residencial');
  3  Tipo_tel ( 'P01', 'Personal');
  4  Tipo_tel ( 'C01', 'Otro contacto');
  5  END;
  6  /

PL/SQL procedure successfully completed.
```

```
SQL> BEGIN
  2   Tipo_tel ( 'R01', 'Residencial');
  3  Tipo_tel ( 'P01', 'Personal');
  4  Tipo_tel ( 'C01', 'Otro contacto');
  5  END;
  6  /

PL/SQL procedure successfully completed.
```

```
SQL> Create view vista_tipo_tel as
  2  Select Cod_Telf, Desc_Telf
  3  From Tipo_Telefono;

View created.

SQL> select * from vista_tipo_tel;

COD_T DESC_TELF
----- ------------------------------
C01   Otro contacto
R01   Residencial
P01   Personal
```

Create view vista_tipo_tel as

select Cod_Telf,Desc_telf

From Tipo_Telefono;

select * from vista_tipo_tel;

- **Tipos correos**

```
CREATE OR REPLACE PROCEDURE P_Tipos_Correos(
p_Cod_Correo Tipo_Email.Cod_Email%TYPE,
p_Desc_Correo Tipo_Email.Desc_Email%TYPE
) AS
BEGIN
INSERT INTO Tipo_Email  VALUES (p_Cod_Correo, p_Desc_Correo);
Exception
When others then
DBMS_OUTPUT.PUT_LINE( 'REGISTRO NO CREADO');
COMMIT;
END P_Tipos_Correos;
/

BEGIN
P_Tipos_Correos ( ‘PMail’, ‘Personal’);
P_Tipos_Correos ( ‘IMail’, ‘Institucional’);
P_Tipos_Correos ( ‘OMail’, ‘Otro’);
END;
/
```

- **Profesiones**

```
CREATE OR REPLACE PROCEDURE P_Profesiones(
    p_IDProfesion  Profesion.ID_Profesion%TYPE,
    p_DescProf Profesion.Desc_Prof%TYPE)
AS
BEGIN
    INSERT INTO Profesion VALUES (p_IDProfesion , p_DescProf);
Exception
When others then
DBMS_OUTPUT.PUT_LINE( 'REGISTRO NO CREADO');
COMMIT;
    END P_Profesiones;
    /

BEGIN
        P_Profesiones ( ‘ING’, ‘INGENIERO’);
        P_Profesiones ( ‘ABG’, ‘ABOGADO’);
        P_Profesiones ( ‘DOC’, ‘DOCTOR’);
        P_Profesiones ( ‘POL’, ‘POLICIA’);
        P_Profesiones ( ‘YTB’, ‘YOUTUBER’);
        P_Profesiones ( ‘OTS’, ‘OTROS’);
END;
/
```

- Sucursal

```
CREATE OR REPLACE PROCEDURE P_Sucursal(
    p_Cod_Sucursal  Sucursal.Cod_Sucursal%TYPE,
    p_Nom_Sucursal Sucursal.Nom_Sucursal%TYPE)
AS
BEGIN
    INSERT INTO Sucursal VALUES (p_Cod_Sucursal, p_Nom_Sucursal);
Exception
When others then
DBMS_OUTPUT.PUT_LINE( 'REGISTRO NO CREADO');
COMMIT;
    END P_Sucursal;
    /



BEGIN
        P_Sucursal (‘VL’, ‘VILLA LUCRE’);
        P_Sucursal (‘CL’, ‘CAMPO LIMBERG’);
        P_Sucursal (‘LP’, ‘LOS PUEBLOS’);
END;
/
```

- **Insercion clientes**
- Funcion

```
CREATE OR REPLACE FUNCTION Calcular_Edad(
        p_Fecha_nac Banco_Clientes.Fecha_nac%TYPE
)RETURN NUMBER
AS
Edad_cl number(3);
BEGIN
Edad_cl:= FLOOR(MONTHS_BETWEEN(SYSDATE, p_Fecha_nac))/12;
RETURN Edad_cl;
END Calcular_Edad;
/

PRUEBA
BEGIN DBMS_OUTPUT.PUT_LINE (‘AGE   ' ||
Calcular_Edad(to_date('&c','DD/MM/YYYY')));
end;

/
```

- Procedimiento

```
CREATE OR REPLACE PROCEDURE P_clientes(
        p_ID_Clientes Banco_Clientes.ID_Cliente%TYPE,
        p_Cedula Banco_Clientes.Cedula%TYPE,
        p_Nombre Banco_Clientes.Nombre%TYPE,
        p_Apellido Banco_Clientes.Apellido%TYPE,
        p_Sexo Banco_Clientes.Sexo%TYPE,
        p_Fecha_nac Banco_Clientes.Fecha_nac%TYPE,
        p_ID_Profesion Banco_Clientes.ID_Profesion%TYPE,
        p_edad_cl Banco_Clientes.edad_cl%TYPE,
        p_Cod_Sucursal Banco_Clientes.Cod_Sucursal%TYPE
)
AS
BEGIN
INSERT INTO Banco_Clientes (ID_Cliente, Cedula, Nombre, Apellido, Sexo, Fecha_nac,
ID_Profesion, Cod_Sucursal, edad_cl ) VALUES (
        p_ID_Clientes, p_Cedula,
        p_Nombre, p_Apellido,
        p_Sexo , p_Fecha_nac ,
        p_ID_Profesion,
        p_Cod_Sucursal, p_edad_cl );
Exception
When others then
DBMS_OUTPUT.PUT_LINE( 'REGISTRO NO CREADO');
COMMIT;


        END P_clientes;
        /
```

- Llamada

```
DECLARE
        p_ID_Clientes Banco_Clientes.ID_Cliente%TYPE;
        p_Cedula Banco_Clientes.Cedula%TYPE;
        p_Nombre Banco_Clientes. Nombre%TYPE;
        p_Apellido Banco_Clientes. Apellido%TYPE;
        p_Sexo Banco_Clientes.Sexo%TYPE;
        p_Fecha_nac Banco_Clientes.Fecha_nac%TYPE;
        p_ID_Profesion Banco_Clientes.ID_Profesion%TYPE;
        p_edad_cl Banco_Clientes.edad_cl%TYPE;
        p_Cod_Sucursal Banco_Clientes.Cod_Sucursal%TYPE;
BEGIN
        p_ID_Clientes := seq_IdClient.nextval;
        p_Cedula := '&p_Cedula';
        p_Nombre := '&p_Nombre';
        p_Apellido := '&p_Apellido';
        p_Sexo := '&p_Sexo';
        p_Fecha_nac := (to_date('&p_Fecha_nac','DD/MM/YYYY'));
        p_ID_Profesion:= '&p_ID_Profesion';
        p_Cod_Sucursal := '&p_Cod_Sucursal';
p_edad_cl := Calcular_Edad(p_Fecha_nac);
P_clientes (p_ID_Clientes, p_Cedula, p_Nombre, p_Apellido, p_Sexo, p_Fecha_nac,
p_ID_Profesion, p_edad_cl, p_Cod_Sucursal );

END;
/
```

Llenar para el lab7

- Prestamos aprobados

```
CREATE OR REPLACE PROCEDURE P_Prestamo_aprobado(
        p_nump Clientes_Prestamo.Numero_prestamo%TYPE,
        p_fecha_ap Clientes_Prestamo.Fecha_aprob%TYPE,
        p_monto_ap Clientes_Prestamo.Monto_aprob%TYPE,
        p_letra Clientes_Prestamo.Letra_mensual%TYPE,
        p_IDp Clientes_Prestamo.Cod_Prestamo%TYPE,
        p_IDc Clientes_Prestamo.ID_cliente%TYPE,
        p_suc Clientes_Prestamo.Cod_Sucursal%TYPE,
        p_saldo Clientes_Prestamo.Saldo_Actual%TYPE,
        p_interes Clientes_Prestamo.Interes_Pagado%TYPE,
        p_fecha_mod Clientes_Prestamo.Fecha_Modif%TYPE,
        p_usuario Clientes_Prestamo.Usuario%TYPE,
        p_monto_pago Clientes_Prestamo.monto_pago%TYPE
)AS
BEGIN
INSERT INTO Clientes_Prestamo (Numero_prestamo, Fecha_aprob, Monto_aprob, Letra_mensual,
Cod_Prestamo, ID_cliente, Cod_Sucursal, Saldo_Actual, Interes_Pagado, Fecha_Modif, Usuario,
monto_pago)
```

```
VALUES (p_nump, p_fecha_ap, p_monto_ap, p_letra, p_IDp, p_IDc, p_suc, p_saldo, p_interes,
p_fecha_mod, p_usuario, p_monto_pago);


UPDATE Suc_Tipo_Prestamo
        SET
        Monto_Prestamos = Monto_Prestamos + p_monto_ap
        WHERE Cod_Prestamo =  p_IDp AND Cod_Sucursal = p_suc;
IF SQL%ROWCOUNT = 0 THEN
        INSERT INTO Suc_Tipo_Prestamo (Cod_Sucursal, Cod_Prestamo, Monto_Prestamos)
        VALUES (p_suc, p_IDp, p_monto_ap);
END IF;
EXCEPTION
WHEN OTHERS THEN
DBMS_OUTPUT.PUT_LINE( 'REGISTRO NO CREADO');
COMMIT;
END P_Prestamo_aprobado;
/
```

INSERCION DE DATOS CON LLAMADA DEL PROCEDIMIENTO

```
BEGIN
P_Prestamo_aprobado (seq_NumPres.nextval, SYSDATE, 30000, 300, 'AT', 1, 'VL', 30000,
9.25,SYSDATE, 2132, 0);

P_Prestamo_aprobado (seq_NumPres.nextval, SYSDATE, 60000, 450, 'ED', 2, 'VL', 60000,
2.25,SYSDATE, 2133, 0);

P_Prestamo_aprobado (seq_NumPres.nextval, SYSDATE, 310000, 600, 'CC', 3, 'LP', 310000,
15.6,SYSDATE, 2131, 0);

P_Prestamo_aprobado (seq_NumPres.nextval, SYSDATE, 65000, 500, 'PR', 4, 'CL', 65000,
13.1,SYSDATE, 2132, 0);

END;
/
```

Create view vista_Prestamo_aprob as

Select Numero_Prestamo, Fecha_aprob, Monto_aprob, Letra_mensual, Cod_Prestamo, ID_cliente, Cod_Sucursal, Saldo_Actual, Interes_Pagado, Fecha_Modif, Usuario

From Clientes_Prestamo;

Select * from vista_Prestamo_aprob;

- Tipo Prestamo

```
ALTER TABLE Prestamo_tipo MODIFY Tasas_Interes number (5,2);
```

```
SQL> ALTER TABLE Prestamo_tipo MODIFY Tasas_Interes number (5,2);

Table altered.
```

```
CREATE OR REPLACE PROCEDURE Tipos_Prestamos (
p_cod_prestamo Prestamo_tipo.cod_prestamo%TYPE,
p_nombre Prestamo_tipo.nombre_prestamo%TYPE,
p_tasa Prestamo_tipo.Tasas_Interes %TYPE
)
AS
BEGIN
INSERT INTO Prestamo_tipo (cod_prestamo, nombre_prestamo, Tasas_Interes)
                Values (p_cod_prestamo, p_nombre, p_tasa );
Exception
When others then
DBMS_OUTPUT.PUT_LINE( 'REGISTRO NO CREADO');
COMMIT;
END Tipos_prestamos;
/
```

```
SQL> CREATE OR REPLACE PROCEDURE Tipos_Prestamos (
  2  p_cod_prestamo Prestamo_tipo.cod_prestamo%TYPE,
  3  p_nombre Prestamo_tipo.nombre_prestamo%TYPE,
  4  p_tasa Prestamo_tipo.Tasas_Interes %TYPE
  5  )
  6  AS
  7  BEGIN
  8  INSERT INTO Prestamo_tipo (cod_prestamo, nombre_prestamo, Tasas_Interes)
  9                       Values (p_cod_prestamo, p_nombre, p_tasa );
 10  COMMIT;
 11  END Tipos_prestamos;
 12  /

Procedure created.
```

BEGIN

Tipos_Prestamos (‘HP’ , ‘HIPOTECARIO’, 6.50);

Tipos_Prestamos (‘PR’ , ‘PERSONAL’, 13.1);

Tipos_Prestamos (‘ED’ , ‘EDUCATIVO’, 2.25);

Tipos_Prestamos (‘CC’ , ‘CONSTRUCCION’, 15.6);

Tipos_Prestamos (‘AT’ , ‘AUTOMOVIL’, 9.25);

END;

/

```
CREATE OR REPLACE PROCEDURE Trans_Pag(
p_ID_Transaccion Transac_pagos.ID_Transaccion%TYPE,
p_Cod_Sucursal  Transac_pagos.Cod_Sucursal%TYPE,
p_Monto_pago Transac_pagos.Monto_pago%TYPE,
p_Fecha_transac Transac_pagos.Fecha_transac%TYPE,
p_Fecha_inserc Transac_pagos.Fecha_inserc%TYPE,
p_Usuario Transac_pagos.Usuario%TYPE,
p_ID_Cliente Transac_pagos.ID_Cliente%TYPE,
p_Cod_Prestamo Transac_pagos.Cod_Prestamo%TYPE
) AS
BEGIN
INSERT INTO Transac_pagos VALUES (
p_ID_Transaccion,
p_Cod_Sucursal,
p_Monto_pago,
p_Fecha_transac,
p_Fecha_inserc,
p_Usuario,
p_ID_Cliente ,
p_Cod_Prestamo);
Exception
When others then
DBMS_OUTPUT.PUT_LINE( 'REGISTRO NO CREADO');
COMMIT;
End Trans_Pag;
/
```

```
BEGIN
Trans_Pag (seq_IDTrans.nextval, 'VL', 300, SYSDATE, SYSDATE, 2321, 2, 'ED'
);
Trans_Pag (seq_IDTrans.nextval, 'LP', 600, SYSDATE, SYSDATE, 2321, 3, 'CC'
);
Trans_Pag (seq_IDTrans.nextval, 'CL', 500, SYSDATE, SYSDATE, 2422, 4, 'PR'
);
Trans_Pag (seq_IDTrans.nextval, 'VL', 300, SYSDATE, SYSDATE, 3321, 1, 'AT'
);
END;
/
```

- Procesamiento Pagos

Calcular interés

```
CREATE OR REPLACE FUNCTION f_Interes(
f_tasa_int  Prestamo_Tipo.Tasas_Interes%TYPE,
f_saldo_actual  Clientes_Prestamo.Saldo_Actual%TYPE
)RETURN NUMBER
AS
interes_aplicado number(10, 3);
BEGIN
interes_aplicado:= (f_saldo_actual * (f_tasa_int / 100));
RETURN interes_aplicado;
END f_Interes;
/
```

Actualizar pagos

```
CREATE OR REPLACE PROCEDURE actualizar_pagos
AS
CURSOR act_pago IS
SELECT monto_pago, Cod_Sucursal, Cod_Prestamo, ID_Cliente, ID_transaccion
FROM Transac_pagos;
v_pago Clientes_Prestamo.interes_pagado%TYPE;
v_tasa Prestamo_Tipo.Tasas_interes%TYPE;
v_saldo Transac_pagos.monto_pago%TYPE;
v_monto Clientes_Prestamo.monto_aprob%TYPE;
v_saldo_act Clientes_Prestamo.Saldo_Actual%TYPE;
v_apago Clientes_Prestamo.interes_pagado%TYPE;
BEGIN
FOR i IN act_pago LOOP
SELECT Tasas_interes into v_tasa from Prestamo_Tipo
where Cod_Prestamo = i.Cod_Prestamo;
```

```
SELECT Saldo_Actual into v_saldo_act from Clientes_Prestamo
where Cod_Prestamo =i.Cod_Prestamo and ID_Cliente =i. ID_Cliente;
v_apago := f_Interes (v_saldo_act, v_tasa);
v_saldo:= v_apago - i.monto_pago;
IF (v_saldo > 0) THEN
UPDATE Clientes_Prestamo
SET
monto_pago = v_saldo,
Interes_Pagado = v_apago,
Saldo_Actual = Saldo_Actual – v_saldo,
Fecha_Modif = sysdate,
Usuario = user
WHERE ID_Cliente =i.ID_Cliente and Cod_Prestamo = i.Cod_Prestamo;
UPDATE Suc_Tipo_Prestamo
SET
Monto_Prestamos = Monto_Prestamos - v_saldo
where Cod_Prestamo = i.Cod_Prestamo and Cod_Sucursal = i.Cod_Sucursal;
ELSE
UPDATE Clientes_Prestamo
SET
monto_pago = v_saldo,
Interes_Pagado = v_apago,
Fecha_Modif = sysdate,
Usuario = user
WHERE
ID_Cliente =i.ID_Cliente and Cod_Prestamo = i.Cod_Prestamo;
END IF;
END LOOP;
EXCEPTION
```

```
WHEN NO_DATA_FOUND THEN
DBMS_OUTPUT.PUT_LINE ('No se encontro datos');
COMMIT;
END actualizar_pagos;
/
```

UNIVERSIDAD TECNOLÓGICA DE PANAMÁ

FACULTAD DE INGENIERÍA EN SISTEMAS COMPUTACIONALES

LIC. ING. EN SISTEMAS DE INFORMACIÓN

LABORATORIO 7

CURSO: SISTEMA BASE DE DATOS 2

FACILITADORA: 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

--CREACIÓN DE LA TABLA TIPO_AHORRO

```
Create table tipo_ahorro(
cod_ahorro varchar2(5) primary key not null,
descripcion varchar2(50),
tasa_interes number
);
```

--MODIFICACIONES A LA TABLA SUCURSAL******este se repite

```
ALTER TABLE Sucursal  ADD (
Cod_Prestamo varchar2(15),
Constraint Fk_otro_cod_prestamos foreign key (Cod_Prestamo) references Prestamo_Tipo (Cod_Prestamo),
Monto_aprob number,
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) primary key not null,
        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)
);


--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,
        tipo_transac number,
        fecha_transac date,
        monto_dr number,
        fecha_inser date,
        usuario number,
        Cod_Sucursal varchar2(5),
Constraint Fk_otromas_cod_suc foreign key (Cod_Sucursal) references Sucursal (Cod_Sucursal),
        ID_Cliente number,
Constraint Fk_otromas_id_cliente foreign key (ID_Cliente) references Banco_Clientes (ID_Cliente),
        cod_ahorro varchar2(5),
Constraint Fk_otromas_cod_ahorro foreign key (cod_ahorro) references tipo_ahorro (cod_ahorro)
);
--------------------------------------------------------------------------------
ALTER TABLE Transac_dr MODIFY tipo_transac VARCHAR(10);

--------------------------------------------------------------------------------
--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, --secuencia
tipo_transac number,
saldo_cta_anterior number,
monto_transac number,
saldo_final number,
usuario number,
fecha date,
ID_Cliente number,
Constraint Fk_id_clientes foreign key (ID_Cliente) references Banco_Clientes (ID_Cliente),
cod_ahorro varchar2(5),
Constraint Fk_cod_ahorros foreign key (cod_ahorro) references tipo_ahorro (cod_ahorro)
);

-----------------------------------------------------------------------------------------
ALTER TABLE Auditoria MODIFY tipo_transac VARCHAR(10);

ALTER TABLE Transac_dr MODIFY ID_Transaccion primary key;
ALTER TABLE Auditoria ADD reg_auditoria number;
```

```
CREATE SEQUENCE sqn_reg_auditoria START WITH 1 INCREMENT BY 1;


---Creacion de la tabla temp_table

Create table temp_table(
ID_Transaccion number, --secuencia
tipo_transac number,
saldo_cta_anterior number,
monto_transac number,
saldo_final number,
usuario number,
fecha date,
ID_Cliente number,
Constraint Fk_id_clientessss foreign key (ID_Cliente) references Banco_Clientes
(ID_Cliente),
cod_ahorro varchar2(5),
Constraint Fk_cod_ahorrossss foreign key (cod_ahorro) references tipo_ahorro
(cod_ahorro)
);
ALTER TABLE temp_table ADD reg_auditoria number;
```

PROCEDIMIENTOS

--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_ahorro);
EXCEPTION
WHEN OTHERS THEN
DBMS_OUTPUT.PUT_LINE( 'REGISTRO NO CREADO');
COMMIT;
END p_Ahorro_Aprobado;
/
```

Disparadores

```
CREATE OR REPLACE TRIGGER insert_auditoria
BEFORE INSERT ON Auditoria
DECLARE
v_reg_auditoria Auditoria.reg_auditoria%TYPE;
v_ID_Transaccion Auditoria.ID_Transaccion%TYPE;
v_tipo_transac Auditoria.tipo_transac%TYPE;
v_saldo_cta_anterior Auditoria.saldo_cta_anterior%TYPE;
v_monto_transac Auditoria.monto_transac%TYPE;
v_saldo_final Auditoria.saldo_final%TYPE;
v_usuario Auditoria.usuario%TYPE;
v_fecha Auditoria.fecha%TYPE;
v_ID_Cliente Auditoria.ID_Cliente%TYPE;
v_cod_ahorro Auditoria.cod_ahorro%TYPE;
BEGIN

INSERT INTO temp_table (reg_auditoria,ID_Transaccion, tipo_transac, saldo_cta_anterior,
monto_transac, saldo_final, usuario, fecha, ID_Cliente, cod_ahorro)
VALUES (v_reg_auditoria ,v_ID_Transaccion,v_tipo_transac, v_saldo_cta_anterior,
v_monto_transac, v_saldo_final, v_usuario, v_fecha, v_ID_Cliente, v_cod_ahorro);
END  insert_auditoria;
/
```

```
--PROCEDIMIENTO PARA ALMACENAMIENTO DE LA TABLA TRANSAC

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
)
AS
BEGIN
INSERT INTO Transac_dr (ID_Transaccion, tipo_transac , fecha_transac , monto_dr ,
fecha_inser , usuario, 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_Cod_Sucursal , p_ID_Cliente ,p_cod_ahorro
);
EXCEPTION
WHEN OTHERS THEN
DBMS_OUTPUT.PUT_LINE( 'REGISTRO NO CREADO');
COMMIT;
END p_Transacciones;
/
```

----------****---------

```
CREATE OR REPLACE PROCEDURE transac_de_cuenta  (
v_sal_ahorro Ahorros.sal_ahorro%TYPE,
v_sal_interes Ahorros.sal_ahorro%TYPE,
v_monto_dr Transac_dr.monto_dr%TYPE,
v_ID_Cliente Ahorros.ID_Cliente%TYPE,
v_tipo_transac Transac_dr.tipo_transac%TYPE,
v_cod_ahorro Transac_dr.cod_ahorro%TYPE,
v_interes Transac_dr.interes%TYPE)AS
CURSOR  Recorrido IS
SELECT t.tipo_transac,a.sal_ahorro FROM Banco_Clientes b  join  Ahorros a
        on b.ID_Cliente= a.ID_Cliente
        join Transac_dr t on t.cod_ahorro=a.cod_ahorro;
BEGIN
OPEN Recorrido;
LOOP
FECTH Recorrido INTO  v_tipo_transac, v_cod_ahorro;
EXIT WHEN Recorrido%NOTFOUND ;
IF( v_cod_ahorro= '01' AND v_cod_ahorro= '03' AND v_tipo_transac= 'DEPOSITO') THEN
UPDATE Ahorros
                SET
                interes = f_calcular_interes(v_interes, v_monto_dr),
                sal_ahorro = v_sal_ahorro,
                sal_interes = v_interes
                WHERE ID_Cliente = v_ID_Cliente;
ELSE IF (v_cod_ahorro= '02' AND v_tipo_transac= 'DEPOSITO')THEN
UPDATE Ahorros
                SET
```

```
                    sal_ahorro= v_sal_ahorro +v_monto_dr
                    WHERE ID_Cliente = v_ID_Cliente;
ELSE IF (v_cod_ahorro= ‘02’ AND v_tipo_transac= ‘RETIRO’ AND v_sal_ahorro > v_monto_dr)THEN
          IF v_sal_ahorro < v_monto_dr THEN
          DBMS_OUTPUT.PUT_LINE('SALDO INSUFICIENTE');
          ELSE IF v_sal_ahorro > v_monto_dr THEN
          UPDATE Ahorros
                              SET
                              sal_ahorro= v_sal_ahorro -v_monto_dr
                              WHERE ID_Cliente = v_ID_Cliente;
          END IF;
ELSE
          DBMS_OUTPUT.PUT_LINE('NO SE PUDO REALIZAR LA TRANSACCION');
END IF;
FECTH Recorrido INTO  v_tipo_transac, v_cod_ahorro;
END LOOP;
CLOSE Recorrido;
END transac_de_cuenta;
/
```

2 errores NO es mucho pero es honesto;

```
-------------------------------------------------------------------------------------------------------------

CREATE OR REPLACE TRIGGER act_sucursal

AFTER INSERT ON Transac_dr FOR EACH ROW

DECLARE

BEGIN

        UPDATE Sucursal

        SET monto_ahorrado = monto_ahorrado + :new.monto_dr;

END;

/

-------------------------------------------------------------------------------------------------------------


CREATE OR REPLACE FUNCTION f_calcular_interes(

        f_interes Ahorros.interes%TYPE,

        f_saldo Ahorros.sal_ahorro%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;
/
-------------------------------------------------------------
```

```
-----------------------------------------------------
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;
 /
-----------------------------------------------
```

INSERCIONES

```
BEGIN
p_Tipo_Ahorro(01, 'AHORRO DE NAVIDAD', 6);
p_Tipo_Ahorro(02, 'AHORRO CORRIENTE', 4);
p_Tipo_Ahorro(03, 'AHORRO ESCOLAR', 6);
END;
/

BEGIN
p_Ahorro_Aprobado (01, seq_num_cta.nextval, sysdate, 6, 10, 0, 0, sysdate, sysdate, 1231,
sysdate, 2, 'VL');
p_Ahorro_Aprobado (02, seq_num_cta.nextval, sysdate, 4, 25, 0, 0, sysdate, sysdate, 1131,
sysdate, 3, 'LP');
p_Ahorro_Aprobado (03, seq_num_cta.nextval, sysdate, 6, 5, 0, 0, sysdate, sysdate, 2231, sysdate,
4, 'CL');
END;
/


BEGIN
p_Transacciones (seq_id_trans.nextval, 'DEPOSITO', sysdate, 25,sysdate, 1243, 'VL',3, '01' );
p_Transacciones (seq_id_trans.nextval, 'DEPOSITO', sysdate, 10,sysdate, 6578, 'LP',2,'02' );
p_Transacciones (seq_id_trans.nextval, 'DEPOSITO', sysdate, 5,sysdate, 9807, 'CL', 1,'03' );
END;
/
```

LOS QUIERO SON LOS MEJORES!! <3