Se tiene el siguiente sistema de información sobre una biblioteca:

* Un autor crea varias obras (al menos una) y una obra es creada por varios autores (al menos uno). La información de los autores es id\_autor, nombre, país.
* En cuanto a las obras, pueden ser de 3 tipos: CD, libro o película. La información común de cada una obra es: id\_obra, nombre, año. La información de cada CD es número de canciones. La información de cada libro es número de páginas. La información de cada película es duración.
* Una obra tiene asociadas varias copias, siendo la existencia de una copia totalmente dependiente de la existencia de la obra a la que pertenece (es decir, una copia no puede existir si no existe la obra a la que están asociada). La información de una copia es número de copia, si está deteriorada, comentarios.
* Un socio pide prestadas varias copias (opcionalmente ninguna). Una copia es prestada a varios socios (opcionalmente ninguno). La información del socio es: nombre, apellidos, cod\_socio, teléfono, dirección. Nota: el cod\_socio es un identificador único para los socios. Se almacena la información siguiente sobre los préstamos realizados: fecha\_prestamo, fecha\_devolución\_prestamo, fecha\_tope\_prestamo.

1. (2 puntos) Realizar el diagrama entidad-relación, indicando entidades, relaciones, atributos, clave primaria, cardinalidad de las relaciones.
2. (2 puntos) Realizar el diagrama o modelo relacional, es decir, escribir los esquemas relacionales, indicando bien en cada esquema el nombre de la tabla, atributos e indicar que atributo(s) forma(n) la clave primaria.
3. Dado el siguiente esquema relacional de un banco: cada sucursal tiene al menos un cliente (y un cliente solo pertenece a una sucursal); y un cliente tiene al menos una cuenta (y una cuenta pertenece a un cliente solamente):

sucursal (numero\_sucursal, ciudad)

cliente (numero\_cliente, nombre, apellido, teléfono, numero\_sucursal)

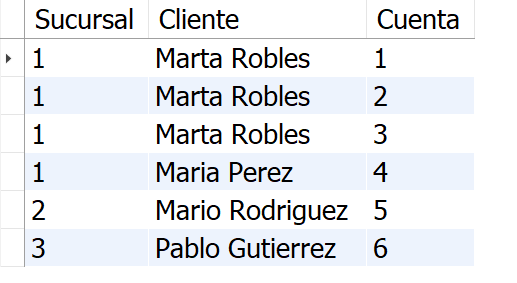
cuenta (numero\_cuenta, saldo, fecha\_apertura, numero\_cliente)

Respecto a los tipos de datos, todos son varchar, menos saldo que es decimal, y fecha, que es de tipo datetime.

Codifica las siguientes sentencias en MySQL para realizar lo siguiente:

1. (0,75 puntos) Crear las tres tablas, con la primary key (clave primaria) y las foreign key (clave forénea). Las sentencias de creación de tablas deben estar completas.
2. (1 puntos) Mostrar los números de las sucursales junto a los nombres y apellidos de los clientes de cada cuenta, y los números de cuenta de cada cliente. Debe mostrarlo por orden ascendente de nombre de sucursal. Además, en la cabecera (o título) de cada atributo debe mostrar Sucursal (para número\_sucursal), Cliente (para nombre apellido del cliente), Cuenta (para número\_cuenta).

Un ejemplo de la salida es el siguiente:



1. (0,5 puntos) Crear una vista que muestre el atributo ciudad de las sucursales y la cantidad de clientes que tiene cada sucursal.
2. (0,75 ) Crear una vista que muestre los números de cliente de la sucursal 1 (numero\_ sucursal es 1), y la suma de saldo de todas las cuentas de los clientes de esa sucursal.
3. (0,25 puntos) Borra el índice primario de la tabla sucursal.
4. (1,25 puntos) Crear un procedimiento almacenado que realice lo siguiente:

- Parametros de entrada: n\_cliente (de tipo varchar(10)), limite (de tipo decimal(13,2))

-Parametro de salida: tipo (de tipo varchar(10)

-Crear una variable llamada resultado de tipo int

-Guardar en la variable resultado, la cantidad de cuentas cuyo numero\_cliente coincida con n\_cliente, pero que cuyo saldo NO supere limite.

-Si resultado es mayor que 1, guardar en la variable tipo el valor ‘ALTO’; si resultado es igual a 1, guardar en la variable tipo el valor ‘MEDIO’, si resultado es igual a 0, guardar en la variable tipo el valor ‘BAJO’.

g. (0,5 puntos) Invocar a este procedimiento del apartado g. con el n\_cliente 10100, limite de 1000.50. Mostrar el valor del parámetro de salida llamado tipo.

h. (0,25 puntos) Crear una tabla llamada movimientos que contenga los siguientes atributos: id\_cuenta varchar(10), cantidad decimal(13,2), fecha datetime). Esta tabla va a contener las actualizaciones que se realicen sobre la tabla cuenta.

i. (0,75 puntos) Crear un disparador que realice lo siguiente:

Después de cada actualización sobre la tabla cuenta, si el saldo nuevo es mayor del anterior, debe insertar una fila nueva en la tabla movimientos, con el numero\_cuenta que se ha visto actualizada, la diferencia entre los saldos (saldo nuevo menos saldo anterior), y el día y hora actuales.

***Ejercicios resueltos***

Ejercicio 1:

Diagrama

Descripción generada automáticamente

Ejercicio 2:

autores (id autor, nombre, país)

crear (id obra, id\_autor)

obra (id obra, nombre, año)

cd (id obra, n\_canciones)

libro (id obra, n\_paginas)

película (id obra, duración)

copias (id obra, n\_copia, deteriorada, comentarios)

socios (cod\_socio, nombre, apellidos, dirección, teléfono)

realizar\_prestamo (id obra, n\_copia, cod\_socio, fecha\_presta, fecha\_devo, fecha\_tope)

Ejercicio 3:

CREATE TABLE sucursal

(numero\_sucursal varchar(10) PRIMARY KEY,

ciudad varchar(30));

CREATE TABLE cliente (numero\_cliente varchar(10) primary KEY,

nombre varchar(30), apellido varchar(30),

teléfono varchar(10), numero\_sucursal varchar(10),

FOREIGN KEY (numero\_sucursal)

REFERENCES sucursal(numero\_sucursal));

CREATE TABLE cuenta (numero\_cuenta varchar(10) primary KEY,

saldo decimal (13,2), fecha datetime, numero\_cliente varchar(10),

FOREIGN KEY (numero\_cliente)

REFERENCES cliente(numero\_cliente));

select numero\_sucursal as 'Sucursal',

concat (nombre, ' ', apellido) as 'Cliente',

numero\_cuenta as 'Cuenta'

from sucursal

inner join cliente using (numero\_sucursal)

inner join cuenta using (numero\_cliente)

order by numero\_sucursal;

create view vista4 as select ciudad, count(\*) from

sucursal inner join cliente using (numero\_sucursal)

group by ciudad;

create view vista5 as select numero\_cliente, sum(saldo) from

cliente inner join cuenta using (numero\_cliente)

where numero\_sucursal = '1'

group by numero\_cliente;

DROP INDEX `PRIMARY` ON sucursal;

CREATE DEFINER=`root`@`localhost` PROCEDURE `Examen\_final`(IN n\_cliente varchar(10),

IN limite decimal(13,2),

OUT tipo varchar(10))

BEGIN

DECLARE resultado int;

SELECT COUNT(numero\_cuenta)

INTO resultado

FROM cuenta

WHERE numero\_cliente = n\_cliente

and saldo <= limite;

IF resultado > 1 THEN

SET TIPO = 'ALTO';

ELSEIF resultado = 1 THEN

SET TIPO = 'MEDIO';

ELSE

SET TIPO = 'BAJO';

END IF;

END

CALL Examen\_final ('10100’, 1000.50, @t);

select @t;

create table movimientos

(id\_cuenta varchar(10),

cantidad decimal(13,2),

fecha datetime);

DELIMITER $$

CREATE TRIGGER after\_cuentas\_update

AFTER UPDATE

ON cuenta FOR EACH ROW

BEGIN

IF new.saldo > old.saldo THEN

INSERT INTO movimientos

VALUES(old.numero\_cuenta, new.saldo-old.saldo, now());

END IF;

END$$

DELIMITER ;