

# UNIVERSIDAD PRIVADA

## “FRANZ TAMAYO”

Ingeniería de Sistemas



### BASE DE DATOS DE UNA BIBLIOTECA

#### AUTORES:

Andrés Vladimir Quiroga Huariste	SIS9208422
Marco Antonio Calle Vaquiata	SIS9929317
Ilia Araceli Sarzo Laura	SIS14125434
Melanie Ingrid Villca Copa	SIS10078756
Dafnet Layda Mamani Laura	SIS1283197

#### DOCENTE:

Lic. William Roddy Barra Paredes

El Alto – Bolivia 2022

## INDICE

CAPITULO I.....	5
Introducción .....	5
1.2 Problema General .....	6
1.3 Objetivos .....	6
1.3.1 Objetivos Generales .....	6
1.3.3 Objetivos Específicos .....	6
CAPITULO II .....	8
Marco Teórico .....	8
2.1 Base de Datos Relacional.....	8
2.2 MariaDB.....	8
2.3 SQL .....	8
2.4 DDL (lenguaje de definición de base de datos) .....	9
2.5 DML (lenguaje de manipulación de bases de datos).....	10
2.6 DCL (lenguaje de control de bases de datos).....	11
2.7 DQL (lenguaje de consulta de base de datos) .....	11
2.8 FUNCIONES EN MARIADB .....	12
2.8.1 Características que debe de tener una función .....	12
2.8.2 ¿Cómo crear, modificar y cómo eliminar una función?.....	12
2.9 Función CONCAT .....	13

2.10 Función SUBSTRING .....	13
2.9 Función STRCMP .....	14
2.10 Función CHAR_LENGTH y LOCATE.....	14
2.11 Parametros de entrada y salida .....	15
2.12Trigger en MariaDB .....	16
2.13 Vistas en MariaDB .....	16
CAPITULO III .....	17
Marco Aplicative .....	17
3.1 Analisis Y Diseño De La Base De Datos .....	17
3.2 Diseño de la base de Datos.....	18
3.2.1 Diseño E-R .....	18
3.2.3 Modelo Lógico .....	29
3.2.3 Modelo Lógico-Vistas.....	30
3.3 Usabilidad .....	30
Consultas-Resultado.....	30
FUNCIONES .....	32
VISTAS.....	33
TRIGGERS .....	34
3.4 Video De La Funcionalidad Del Sistema.....	35
Link: .....	35

CAPITULO IV .....	36
4.1 Conclusiones .....	36

## **CAPITULO I**

### **Introducción**

Una base de datos proporciona a los usuarios el acceso a datos, que pueden visualizar, ingresar o actualizar, en concordancia con los derechos de acceso que se les hayan otorgado. En este trabajo realizado aplicaremos todos los temas que aprendimos durante el semestre, siendo así de nuestra elección la creación de la base de datos para una biblioteca. Utilizando MariaDB en DataGrip, mostraremos el diagrama de entidad-relación, el modelo lógico de la DB, las respectivas tablas con sus relaciones y atributos, su modelo jerárquico, los diferentes comandos.

## **1.2 Problema General**

El problema a la que se enfrenta la biblioteca es el manejo de la gestión de los recursos cuando se solicita algún tipo de información, prestar y recibir los libros, etc, ya sea para el usuario o el administrador de la biblioteca ocasiona pérdida de tiempo y al buscar de manera manual la información solicitada.

Debido a que no cuenta con un sistema automatizado que ayude a tener acceso a la información oportuna, esto produce resultados negativos al factor social, económico y tecnológico. Es por aquello que la biblioteca necesita tener dicha información disponible de manera rápida y precisa, automatizando el proceso de búsqueda y agilitando al bibliotecario la información deseada

## **1.3 Objetivos**

### **1.3.1 Objetivos Generales**

Implementar un sistema de información bibliotecario que permita gestionar los procesos administrativos de la Biblioteca

### **1.3.3 Objetivos Específicos**

Desarrollar una base datos en DataGrip para almacenar datos actuales e historicos de la biblioteca.

Organizar y proporcionar informacion de manera sencilla.

Mediante este sistema de control de la base de datos, tendra la biblioteca una administracion organizada y clara ante el uso de datos del cliente como tambien el de los libros

Tener una flexibilidad para poder modificar o ingresar datos que son cambiantes de forma continua, esto nos permite poder mantenernos al día con la información y de una manera simple.

Otro de los objetivos fundamentales de una base de datos es mantener la calidad e integridad de los datos bajo cualquier circunstancia

## CAPITULO II

### Marco Teorico

Figura 1



Fuente: google

### 2.1 Base de Datos Relacional

Una base de datos relacional es básicamente un conjunto de tablas (relaciones bidimensionales), similares a las tablas de una hoja de cálculo, formadas por filas (registros) y columnas (campos).

### 2.2 MariaDB

MariaDB es un sustituto de MySQL, con licencia GPL, en donde se incorporan todas las mejoras con más funcionalidades y un máximo rendimiento que permite modificar, almacenar y extraer información para servicios SQL sólidos y escalables. Fue desarrollado por Michael Widenius, fundador de MySQL y la comunidad de desarrolladores de software libre.

### 2.3 SQL

SQL es un lenguaje de base de datos estándar, un lenguaje descriptivo no procedimental orientado a conjuntos.

Es potente, eficiente, fácil de aprender y mantener. El lenguaje SQL se divide en las siguientes cuatro



categorías:

DDL, DML, DCL, DQL.

## 2.4 DDL (lenguaje de definición de base de datos)

Se utiliza para crear varios objetos en la base de datos: tablas, vistas, índices, sinónimos, clústeres, etc., comandos de uso común como crear, soltar, alterar, etc.

- **crear:** se utiliza para crear bases de datos, tablas, índices, vistas, etc.

### Sintaxis común:

# Crear base de datos:

```
CREATE DATABASE [IF NOT EXISTS] db_name ;
```

# Crear tabla:

```
CREATE TABLE [IF NOT EXISTS] tbl_name
```

```
(create_definition,...)
```

```
[table_options] ;
```

- **drop:** se utiliza para eliminar bases de datos, tablas, índices, vistas, etc.

### Sintaxis común:

# Eliminar base de datos

```
DROP DATABASE [IF EXISTS] db_name ;
```

# Eliminar tabla

```
DROP TABLE [IF EXISTS] tbl_name [, tbl_name] .. ;
```

- **alter:** se usa para modificar la base de datos, la tabla o los campos de la tabla, índices, vistas y otra información.

### Sintaxis común:

# Modificar la información de la base de datos

```
ALTER DATABASE [db_name] [DEFAULT] CHARACTER SET [=] charset_name ;
```

# Modificar la información de la tabla:

```
ALTER TABLE tbl_name [alter_specification [, alter_specification] ...];
```

#alter\_specification puede tener el siguiente contenido:

alter\_specification :

```
    ADD [COLUMN] col_name column_definition
        [FIRST | AFTER col_name ]
|  ADD [CONSTRAINT] PRIMARY KEY
|  ALTER [COLUMN] col_name {SET DEFAULT value | DROP DEFAULT}
|  DROP [COLUMN] col_name
|  DROP PRIMARY KEY
|  CHANGE [COLUMN] old_col_name new_col_name column_definition
        [FIRST|AFTER col_name]
|  MODIFY [COLUMN] col_name column_definition
        [FIRST | AFTER col_name];
```

## 2.5 DML (lenguaje de manipulación de bases de datos)

DML es un lenguaje que puede realizar operaciones en datos de tabla en los datos. Las instrucciones de manipulación de datos incluyen: actualizar, insertar y eliminar.

- insertar: inserta datos en la tabla de datos especificada, que puede ser uno o más datos,

sintaxis:

```
INSERT [INTO] tbl_name [(col_name,...)]
    {VALUES | VALUE} (value1),(value2),...;
```

- actualización: se utiliza para modificar los datos de la tabla. gramática:

```
UPDATE table_name
    SET col_name1={expr1|DEFAULT} [, col_name2={expr2|DEFAULT}] ...
```

[WHERE where\_condition];

- borrar: borra los datos de la tabla. gramática:

DELETE [col\_name] FROM tbl\_name

[WHERE where\_condition];

**Nota:** Cuando actualice y elimine los datos en la tabla, debe usar la cláusula where, de lo contrario, el valor predeterminado es operar todas las filas. Si especifica col\_name en la declaración de eliminación, eliminará los datos del campo.

## 2.6 DCL (lenguaje de control de bases de datos)

DCL se utiliza para otorgar o revocar ciertos privilegios para acceder a la base de datos y controlar el tiempo y el efecto de las transacciones de manipulación de la base de datos y monitorear la base de datos. El procesamiento de transacciones es principalmente retrotracción y confirmación. Lo principal es el control de permisos. La sintaxis es la siguiente:

GRANT privileges ON database. table TO 'username'@'host' IDENTIFIED BY('password');

**Nota:** la base de datos y la tabla usan \* para indicar todo, y el host usa% para indicar cualquier parámetro.

## 2.7 DQL (lenguaje de consulta de base de datos)

La estructura básica del lenguaje de consulta de datos DQL se compone de cláusula SELECT, cláusula FROM, WHERE

Cláusula . gramática:

SELECT

[ALL | DISTINCT ] select\_expr1 [, select\_expr2 ...]

[FROM table\_references

[WHERE where\_condition]  
[GROUP BY {col\_name | expr | position}  
[ASC | DESC] [HAVING where\_condition]]  
[ORDER BY {col\_name | expr | position}  
[ASC | DESC], ...]  
[LIMIT {[offset,] row\_count | row\_count OFFSET offset}]

## 2.8 FUNCIONES EN MARIADB

Una función es una rutina creada para tomar unos parámetros, procesarlos y retornar en una salida.

### 2.8.1 Características que debe de tener una función

- Solamente pueden tener parámetros de entrada IN y no parámetros de salida OUT o INOUT
- Deben retornar en un valor con algún tipo de dato definido
- Pueden usarse en el contexto de una sentencia SQL
- Solo retornan un valor individual, no un conjunto de registros

### 2.8.2 ¿Cómo crear, modificar y cómo eliminar una función?

Para crear una función debemos de usar la sentencia CREATE FUNCTION. La sintaxis para crear una función es casi idéntica a la de crear un procedimiento, veamos:

```
CREATE FUNCTION nombre_función (parametro1, parametro2)
RETURNS tipoDato
[atributos de la rutina]
<bloque de instrucciones>
```

Para modificar una función usamos el comando ALTER FUNCTION. Con esta sentencia podemos cambiar los atributos de la función, pero no podremos cambiar el cuerpo. Veamos la sintaxis:

```
ALTER FUNCTION nombre_funcion  
[SQL SECURITY {DEFINER|INVOKER}]  
[COMMENT descripción ]
```

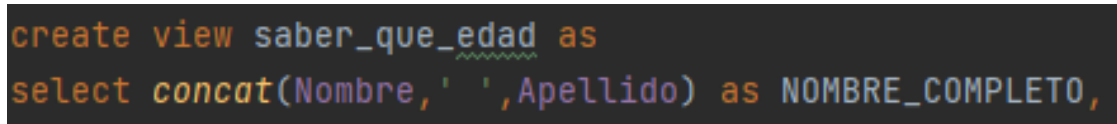
Para eliminar una función usamos el comando DROP FUNCTION. Simplemente especificamos el nombre de la función y esta se borrará de la base de datos. Su sintaxis esta definida de la siguiente forma:

```
DROP FUNCTION nombre_funcion
```

## 2.9 Función CONCAT

CONCAT es una función de cadena para combinar o unir dos o más cadenas y devolverlas como un solo valor. El nombre CONCAT proviene del verbo concatenación, que significa unir 2 o más entidades juntas.

**Figura 2**



```
create view saber_que_edad as  
select concat(Nombre, ' ', Apellido) as NOMBRE_COMPLETO,
```

**Fuente: propia**

## 2.10 Función SUBSTRING

La función de subcadena se utiliza para extraer una subcadena o una parte de la cadena contra la cadena de entrada. Como sugiere el nombre, la función Substring opera en una cadena de entrada y devuelve una subcadena más pequeña contra las opciones especificadas.

**Figura 3**

```
set new.usuario=concat(substring(new.nombre,1,2),  
set new.password = concat(substring(new.nombre,1,
```

**Fuente: propia**

## **2.9 Función STRCMP**

La función STRCMP() se usa para comparar dos strings. Si ambas strings son iguales, devuelve 0, si el primer argumento es más pequeño que el segundo según el orden definido, devuelve -1 y devuelve 1 cuando el segundo es más pequeño que el primero.

**Figura 4**

```
Select STRCMP('Geeks', 'Geeks') As 'Cmp_Value'
```

**Fuente: google**

## **2.10 Función CHAR\_LENGTH y LOCATE**

La función CHAR\_LENGTH() en MySQL se usa para encontrar la longitud de una string dada (en caracteres). Cuenta el número de caracteres e ignora si los caracteres son de un solo byte o de varios bytes.

La función LOCATE() en MySQL se usa para encontrar la ubicación de una substring en una string. Devolverá la ubicación de la primera aparición de la substring en la string. Si la substring no está presente en la string, devolverá 0

**Figura 5**

```
SELECT CHAR_LENGTH("SQL Tutorial in geeksforgeeks") AS  
LengthOfString
```

**Fuente:** google

### **¿Cuál es la diferencia entre las funciones de agregación y funciones creados por el DBA?**

Las funciones de agregación en SQL nos permiten efectuar operaciones sobre un conjunto de resultados, pero devolviendo un único valor agregado para todos ellos. Es decir, nos permiten obtener medias, máximos, etc... sobre un conjunto de valores. Una función definida por el usuario (UDF) es un modo de extender MariaDB con una nueva función que funciona como una función nativa de MariaDB tal como ABS() o CONCAT() . function\_name es el nombre que debe usarse en comandos SQL para invocar la función.

#### **2.11 Parametros de entrada y salida**

**IN:** Es el tipo de parámetro que se usa por defecto. La aplicación o código que invoque al procedimiento tendrá que pasar un argumento para este parámetro. El procedimiento trabajará con una copia de su valor, teniendo el parámetro su valor original al terminar la ejecución del procedimiento.

**OUT:** El valor de este parámetro puede ser cambiado en el procedimiento, y además su valor modificado será enviado de vuelta al código o programa que invoca el procedimiento.

**INOUT:** Es una mezcla de los dos conceptos anteriores. La aplicación o código que invoca al procedimiento puede pasarle un valor a éste, devolviendo el valor modificado al terminar la ejecución. En caso de resultarte confuso, echa un ojo al ejemplo que verás más adelante.

## 2.12 Trigger en MariaDB

El trigger es un objeto de la base de datos que está asociado con una tabla. Se activará cuando una acción definida se ejecute en la tabla. El trigger puede usarse para ejecutar una de las siguientes sentencias MySQL en la tabla: INSERT, UPDATE y DELETE. Se puede invocar antes o después del evento.

**Figura 6**

```
create or replace trigger genera
before insert on bibliotecario
for each row
begin
```

**Fuente: propia**

## 2.13 Vistas en MariaDB

Las vistas en MariaDB o MySQL son tablas virtuales que no almacenan ningún dato, sino que es el resultado de la consulta de varias tablas o una según se allá hecho la consulta.

**Figura 7**

```
create view saber_que_edad as
select concat(Nombre, ' ', Apellido) as NOMBRE_COMPLETO,
       case
         when edad >=18 then 'mayor de edad'
         when edad <18 then 'menor de edad'
       end as EDAD_DE_LOS_ESTUDIANTES,
       Fecha,
       Id_estudiante
from estudiantes;
```

**Fuente: propia**



## CAPITULO III

### Marco Aplicativo

#### 3.1 Analisis Y Diseño De La Base De Datos

##### a. Contexto de la Base de Datos

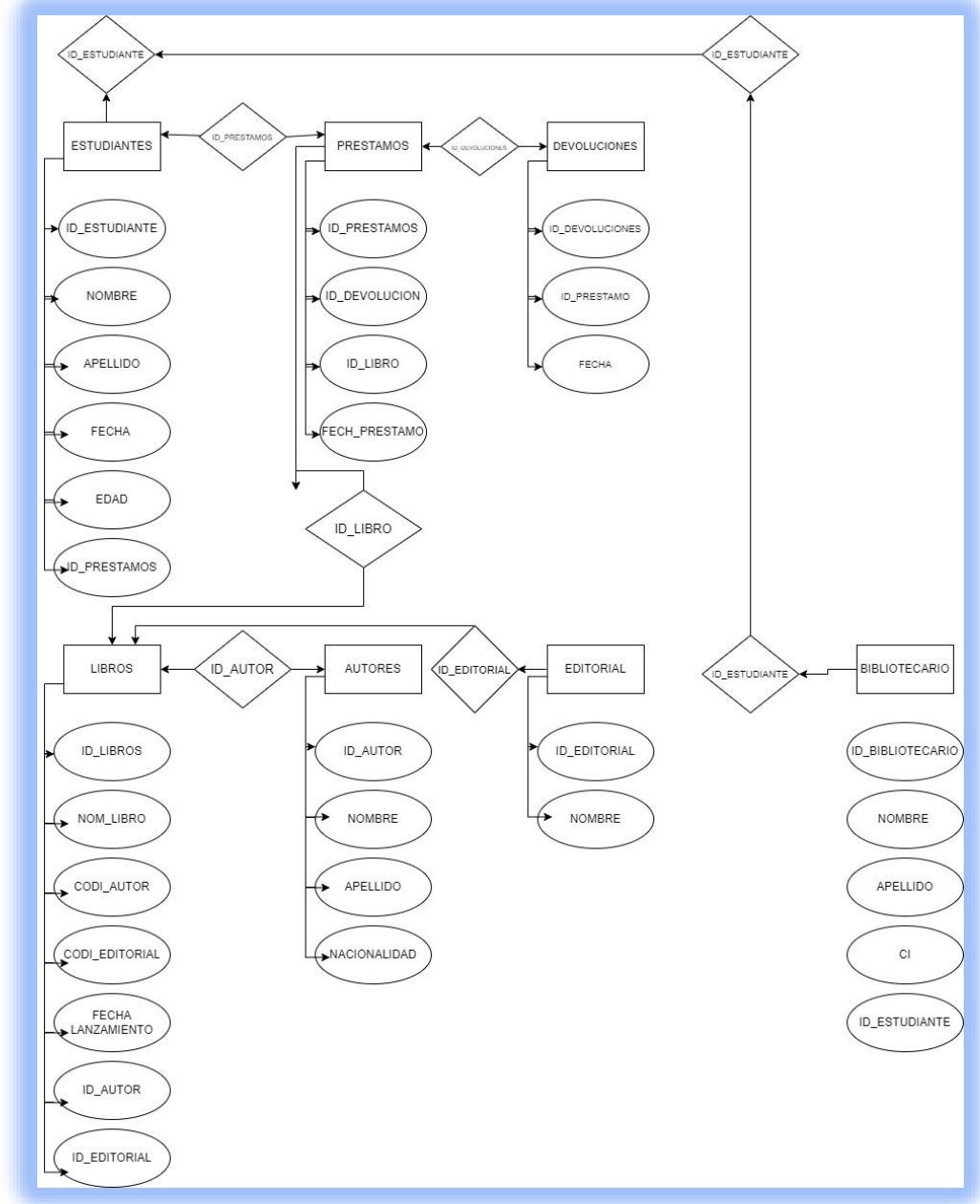
Dada la situación de mejorar la Biblioteca, identificamos que el nombre adecuado para la base de datos deberá ser DB\_Biblioteca.

##### b. Entidades/tablas de sistema

<b>Bibliotecario</b>	Almacena el registro del bibliotecario
<b>Estudiantes</b>	Almacena los datos de los estudiantes
<b>Prestamos</b>	Almacena los prestamos que hizo un estudiante
<b>Devoluciones</b>	Almacena las devoluciones que hizo el estudiante
<b>Libros</b>	Almacene los datos de los libros que hay en la Biblioteca
<b>Autores</b>	Almacena los datos de los autores que hicieron el libro
<b>Editorial</b>	Almacena las editoriales de los libros

## 3.2 Diseño de la base de Datos

### 3.2.1 Diseño E-R



Link: <https://drive.google.com/file/d/16prQ8oxajJ91GttkZ8DXvTLJeSpXugsu/view?usp=sha>

ring

### 3.2.2 Archivo.sql

```
CREATE DATABASE DB_Biblioteca;
USE DB_Biblioteca;
DROP DATABASE DB_Biblioteca;
CREATE TABLE Estudiantes
(
    Id_estudiante INTEGER AUTO_INCREMENT PRIMARY KEY NOT NULL,
    Nombre VARCHAR(20),
    Apellido VARCHAR(20),
    Fecha DATE,
    edad integer,
    Id_prestamo INTEGER,
    FOREIGN KEY (Id_prestamo) REFERENCES Prestamos (Id_prestamo)
);

CREATE TABLE Prestamos
(
    Id_prestamo INTEGER AUTO_INCREMENT PRIMARY KEY NOT NULL,
    Id_devolucion INTEGER,
    Id_libro INTEGER,
    fec_prestamo DATE,
    FOREIGN KEY (Id_devolucion) REFERENCES Devoluciones (Id_devolucion),
    FOREIGN KEY (Id_libro) REFERENCES Libros (Id_libro)
);

CREATE TABLE Devoluciones
(
    Id_devolucion INTEGER AUTO_INCREMENT PRIMARY KEY NOT NULL,
    Id_prestamo INTEGER,
    Fecha DATE
);

CREATE TABLE Libros
(
    Id_libro INTEGER AUTO_INCREMENT PRIMARY KEY NOT NULL,
    NombreLibro VARCHAR(25),
   Codigo_autor VARCHAR(50),
   Codigo_editorial VARCHAR(50),
    Fecha_lanzamiento DATE,
    Id_autor INTEGER,
    Id_editorial INTEGER,
    FOREIGN KEY (Id_autor) REFERENCES Autores (Id_autor),
    FOREIGN KEY (Id_editorial) REFERENCES Editorial (Id_editorial)
);
```

```

CREATE TABLE Autores
(
    Id_autor      INTEGER AUTO_INCREMENT PRIMARY KEY NOT NULL,
    Nombre        VARCHAR(20),
    Apellido      VARCHAR(50),
    Nacionalidad  VARCHAR(20)
);

CREATE TABLE Editorial
(
    Id_editorial  INTEGER AUTO_INCREMENT PRIMARY KEY NOT NULL,
    Nombre        VARCHAR(30)
);

CREATE TABLE Bibliotecario
(
    id_bibliotecario INTEGER AUTO_INCREMENT PRIMARY KEY NOT NULL,
    nombre           VARCHAR(100),
    apellido          VARCHAR(100),
    ci                VARCHAR(150),
    Id_estudiante     INTEGER,
    FOREIGN KEY (Id_estudiante) REFERENCES Estudiantes (Id_estudiante)
);

ALTER TABLE Bibliotecario ADD COLUMN HORARIO VARCHAR(50);

INSERT INTO Editorial(Id_editorial, Nombre) VALUES
(1, 'VISOR'),
(2, 'RENACIMIENTO'),
(3, 'MIRAHADAS'),
(4, 'NATURA'),
(5, 'IMPEDIMENTA');

INSERT INTO Autores(Id_autor, Nombre, Apellido, Nacionalidad) VALUES
(1, 'JOSE', 'LUIS BORGER', 'ROMA'),
(2, 'MARIO', 'VARGAS LIOSA', 'INGLATERRA'),
(3, 'GABRIEL', 'GARZIA MENDOZA', 'ESPAÑA'),
(4, 'ISABEL', 'ILLANDEL', 'MEXICO'),
(5, 'ALFONSINA', 'STOMI', 'EEUU');

INSERT INTO Libros(Id_libro, NombreLibro, Codigo_autor, Codigo_editorial,
Fecha_lanzamiento, Id_autor, Id_editorial) VALUES
(1, 'VIAJE AL FIN DE LA NOCHE', '123VFN', '22233344', '1932-04-11', 1, 1);
INSERT INTO Libros(Id_libro, NombreLibro, Codigo_autor, Codigo_editorial,
Fecha_lanzamiento, Id_autor, Id_editorial) VALUES
(2, 'Don Quijote de la Mancha', '123DQM', '11144455', '1605-07-23', 2, 2),
(3, 'Los cuentos de Canterbury', '123LCC', '33355577', '1971-10-04', 3, 3),
(4, 'LAS MIL Y UNA NOCHES', '123LMN', '12345678', '1543-11-27', 4, 4),
(5, 'DECAMERON', '123DCN', '987654321', '1999-11-13', 5, 5),
(6, 'FICCIONES', '123FCC', '999888777', '1881-10-15', 5, 5),
(7, 'EL EXTRANJERO', '123EXT', '555666777', '1761-06-06', 3, 3),
(8, 'GRANDES ESPERANZAS', '123ESP', '000111222', '2000-01-21', 4, 4);

```

```

INSERT INTO Devoluciones(Id_devolucion, Id_prestamo, Fecha) VALUES
(1,1,'2022-12-11'),
(2,2,'2022-12-07'),
(3,3,'2022-12-05'),
(4,4,'2022-11-22'),
(5,5,'2022-11-18');

INSERT INTO Prestamos(Id_prestamo, Id_devolucion, Id_libro, fec_prestamo) VALUES
(1,1,1,'2022-09-09'),
(2,2,5,'2022-09-07'),
(3,3,3,'2022-09-05'),
(4,4,6,'2022-08-20'),
(5,5,8,'2022-08-18');

INSERT INTO Estudiantes(Id_estudiante, Nombre, Apellido,edad, Fecha, Id_prestamo)
VALUES
(1,'MARCO ANTONIO','CALLE VAQUIATA',20,'2002-11-13',1),
(2,'IRIS','MISHEL VELASCO',19,'2002-11-11',2),
(3,'ILIA','SARZO',15,'2006-07-08',3),
(4,'JHON','TORREZ',22,'2003-03-21',4),
(5,'EXTERMINADOR','JUICIO FINAL',25,'1999-06-06',5);
INSERT INTO Estudiantes(Id_estudiante, Nombre, Apellido,edad, Fecha, Id_prestamo)
VALUES
(6,'ROSARIO','KANTUTA',13,'2011-10-21',4);

INSERT INTO bibliotecario(ID_BIBLIOTECARIO, NOMBRE, APELLIDO, CI,HORARIO,
ID_ESTUDIANTE)VALUES
(1,'JHON','TRAVOLTA','123456LP','8AM-14PM',1),
(2,'JIMENA','LAURA','654321CBB','14PM-8PM',2);
INSERT INTO bibliotecario(ID_BIBLIOTECARIO, NOMBRE, APELLIDO, CI,HORARIO,
ID_ESTUDIANTE)VALUES
(4,'ROCKY','SILVESTRE','12326CBB','8AM-14PM',6);

```

## Consultas

```

use db_biblioteca;
#1. Consultas SQL que maneja JOINS = 5 Consultas

#mostraremos los nombres y apellidos de los estudiantes que se prestaron hasta la
fecha de hoy

select Nombre,Apellido,edad, fec_prestamo
from prestamos
inner join estudiantes e on prestamos.Id_prestamo = e.Id_prestamo
where fec_prestamo<'2022-12-07';

#mostraremos todos los datos de los libros que tienen el editorial='NATURA'

select*
from editorial

```

```
inner join libros l on editorial.Id_editorial = l.Id_editorial
where Nombre='NATURA';
```

#Mostraremos los nombres de los autores mas la fecha de prestamo que hicieron los estudiantes mayores a 20 años

```
select (autores.Nombre) as
AUTORES, NombreLibro, fec_prestamo, e2.Nombre, e2.Apellido, e2.edad
from autores
inner join libros l on autores.Id_autor = l.Id_autor
inner join editorial e on l.Id_editorial = e.Id_editorial
inner join prestamos p on l.Id_libro = p.Id_libro
inner join devoluciones d on p.Id_devolucion = d.Id_devolucion
inner join estudiantes e2 on p.Id_prestamo = e2.Id_prestamo
where edad >20;
```

#Mostrame todos los datos del bibliotecario que presto a estudiantes menores a 15 años

#adicionalmente mostrame a que estudiantes presto(solo nombres y edad)

```
select b.nombre, b.apellido, b.ci, b.HORARIO, estudiantes.Nombre, edad
from estudiantes
inner join bibliotecario b on estudiantes.Id_estudiante = b.Id_estudiante
where edad <15;
```

#mostrame los nombres de los estudiantes y que libro mas la fecha en que se les presto y en la que devolvieron

```
select e.Nombre, NombreLibro, fec_prestamo, devoluciones.Fecha
from devoluciones
inner join prestamos p on devoluciones.Id_devolucion = p.Id_devolucion
inner join estudiantes e on p.Id_prestamo = e.Id_prestamo
inner join libros l on p.Id_libro = l.Id_libro
```

## FUNCIONES

```
USE db_biblioteca;
```

#Crearemos una funcion donde devuelva todos los datos del estudiante de mayor edad

```
create or replace function edad_mayor()
returns integer
begin
    return (
        select max(edad)
        from estudiantes
    );
end;
```

```
select*
from estudiantes
```

```
where edad=edad_mayor();
```

```
#Crearemos una funcion donde devuelva todos los datos del estudiante de menor edad
```

```
create or replace function edad_menor()
returns integer
begin
    return (
        select min(edad)
        from estudiantes
        );
end;
```

```
select*
from estudiantes
where edad=edad_menor();
```

```
#Crearemos un funcion que devuelva todos lo datos del estudiante segun su nombre y fecha de nacimiento
#la funcion debe de recibir dos parametros
```

```
create or replace function buscar_estu(nombres varchar(20), fecha date)
returns text
begin
    return (
        select Id_estudiante
        from estudiantes e
        where e.Nombre=nombres and e.Fecha=fecha
        );
end;
```

```
select id_estudiante, nombre, apellido, fecha, id_prestamo, edad
from estudiantes
where id_estudiante=buscar_estu('JHON', '2003-03-21');
```

```
#Crearemos una funcion que busque su fecha de lanzamiento segun el nombre y su autor
```

```
create or replace function buscar_libro(nomlibro VARCHAR(50), autor VARCHAR(50))
returns VARCHAR(50)
begin
    return (
        select Fecha_lanzamiento
        from autores
        inner join libros l on autores.Id_autor = l.Id_autor
        where NombreLibro=nomlibro and Nombre=autor
        );
end;
```

```
SELECT buscar_libro('Don Quijote de la Mancha', 'MARIO');
```

```
#crear un funcion que permita concatenar nombres y apellidos de la tabla
estudiantes
```

```
CREATE or replace function concatenamos_nombres_apellidos(nombres VARCHAR(20),
apellidos varchar(20))
RETURNS VARCHAR(100)
begin
    declare resultado VARCHAR(100) DEFAULT '';
    set resultado = concat('Nombres: ', nombres, ' Apellidos: ', apellidos);
    return resultado;
end;
SELECT concatenamos_nombres_apellidos('MARCO', 'CALLE') as datos;
select concatenamos_nombres_apellidos(Nombre,Apellido) as datos_de_estudiantes
from estudiantes
```

```
1 #Crear un Funcion con la condicionante when, then
```

```
set @usuario='admin';
```

```
create or replace function ejercicio_5()
returns text
begin
    declare respuesta text default '';
    if @usuario = 'admin'
        then
            set respuesta='Usuario Admin';
        else
            set respuesta='Usuario invitado';
    end if;
    return respuesta;
end;
```

```
select ejercicio_5()
```

```
#crear una serie del 1 al 10
```

```
create or replace function ejercicio_6(limite int)
returns text
begin
    declare x int default 1;
    declare serie text default '';

    while x<=limite do
        set serie=concat(serie,x,', ');
        set x=x+1;
    end while;
    return serie;
end;
```



```
select ejercicio_6(10)
```

## VISTAS

```
use db_biblioteca;
```

```
#utilizamos la primera consulta para la creacion de una vista
```

```
create view prestar_fecha_actual as
select Nombre,Apellido,edad, fec_prestamo
from prestamos
inner join estudiantes e on prestamos.Id_prestamo = e.Id_prestamo
where fec_prestamo<'2022-12-07';
```

```
select*
from prestar_fecha_actual;
```

```
#utilizamos la segunda consulta para la creacion de una vista
```

```
create view busca_editorial as
select editorial.nombre,
       id_libro,
       nombrelibro,
       codigo_autor,
       codigo_editorial,
       fecha_lanzamiento,
       id_autor
from editorial
inner join libros l on editorial.Id_editorial = l.Id_editorial
where Nombre='NATURA';
```

```
select*
from busca_editorial;
```

```
#utilizamos la tercera consulta para la creacion de una vista
```

```
create or replace view busca_mayores_20 as
select (autores.Nombre) as
AUTORES,NombreLibro,fec_prestamo,e2.Nombre,e2.Apellido,e2.edad
from autores
inner join libros l on autores.Id_autor = l.Id_autor
inner join editorial e on l.Id_editorial = e.Id_editorial
inner join prestamos p on l.Id_libro = p.Id_libro
inner join devoluciones d on p.Id_devolucion = d.Id_devolucion
inner join estudiantes e2 on p.Id_prestamo = e2.Id_prestamo
where edad >15;
```

```
select*
from busca_mayores_20;
```

```
#crearemos una vista donde tendremos que concatenar el nombre y apellido y tiene
que decir full_name
# donde la edad sera years
# donde los que no devolvieron el libro hasta la fecha actual se dira 'DEUDOR'
```

```
CREATE OR REPLACE VIEW BUSCAR_DEUDOR AS
select CONCAT(Nombre, '-',Apellido) as full_name,
      edad as years,
      case
        when d.Fecha>'2022-12-05' then 'DEUDOR'
      end as fecha_en_la_que_devolvio
from libros
inner join prestamos p on libros.Id_libro = p.Id_libro
inner join estudiantes e on p.Id_prestamo = e.Id_prestamo
inner join devoluciones d on p.Id_devolucion = d.Id_devolucion;
```

```
SELECT*
FROM BUSCAR_DEUDOR;
```

```
#Creamos una vista que te muestre todos los datos de los estudiantes y que si
#son mayores o igual a 18 años ponerlos 'mayor de edad' y si son menores a 18
años poner 'menor de edad'
```

```
create view saber_que_edad as
select concat(Nombre, '-',Apellido) as NOMBRE_COMPLETO,
      case
        when edad >=18 then 'mayor de edad'
        when edad <18 then 'menor de edad'
      end as EDAD_DE_LOS_ESTUDIANTES,
      Fecha,
      Id_estudiante
from estudiantes;
```

```
select EDAD_DE_LOS_ESTUDIANTES
from saber_que_edad;
```

```
select*
from saber_que_edad;
```

## TRIGGERS

```
use db_biblioteca;
```

```
#EN ESTA PARTE HAREMOS USO DE LOS TRIGERRS
```

```
#HAREMOS DOS ADUTORIAS
```

```
CREATE TABLE COPIA_ESTUDIANTES
(
  Nombre VARCHAR(20),
```

```

Apellido VARCHAR(20),
Fecha DATE,
edad integer,
Id_prestamo INTEGER,
FOREIGN KEY (Id_prestamo) REFERENCES Prestamos (Id_prestamo)
);

CREATE TRIGGER GUARDA_DATOS_DEPUES
    BEFORE INSERT ON estudiantes
    FOR EACH ROW
BEGIN
    INSERT INTO COPIA_ESTUDIANTES (Nombre, Apellido, Fecha, edad, Id_prestamo)
    VALUES (NEW.Nombre, NEW.Apellido, NEW.Fecha, NEW.edad, NEW.Id_prestamo);

end;

INSERT INTO estudiantes (Id_estudiante, Nombre, Apellido, Fecha, edad,
Id_prestamo) VALUES
(9, 'JOEL', 'LOPEZ', '2002-07-13', 24, 2);

SELECT*
FROM COPIA_ESTUDIANTES;

CREATE TABLE GUARDA_LIBROS
(
    NombreLibro VARCHAR(25),
   Codigo_autor VARCHAR(50),
Codigo_editorial VARCHAR(50),
Fecha_lanzamiento DATE,
Id_autor INTEGER,
Id_editorial INTEGER,
FOREIGN KEY (Id_autor) REFERENCES Autores (Id_autor),
FOREIGN KEY (Id_editorial) REFERENCES Editorial (Id_editorial)
);

CREATE TRIGGER LIBROS_GUARDADOS_DATOS
    AFTER insert ON libros
    FOR EACH ROW
BEGIN
    INSERT INTO GUARDA_LIBROS (NombreLibro, Codigo_autor, Codigo_editorial,
Fecha_lanzamiento, Id_autor, Id_editorial)
    VALUES (NEW.NombreLibro, NEW.Codigo_autor, NEW.Codigo_editorial,
NEW.Fecha_lanzamiento, new.Id_autor, new.Id_editorial);

end;

insert into libros (id_libro, nombrelibro, codigo_autor, codigo_editorial,
fecha_lanzamiento, id_autor, id_editorial)
values (9, 'LA TIERRA PERDIDA', '1234HGF123', 'D123D1', '1781-09-22', 3, 3);

SELECT*

```

```

FROM guarda_libros;

#HAREMOS UN TRIGGERS DE VERIFICACION

alter table bibliotecario add column password varchar(50);
alter table bibliotecario add column usuario varchar(50);

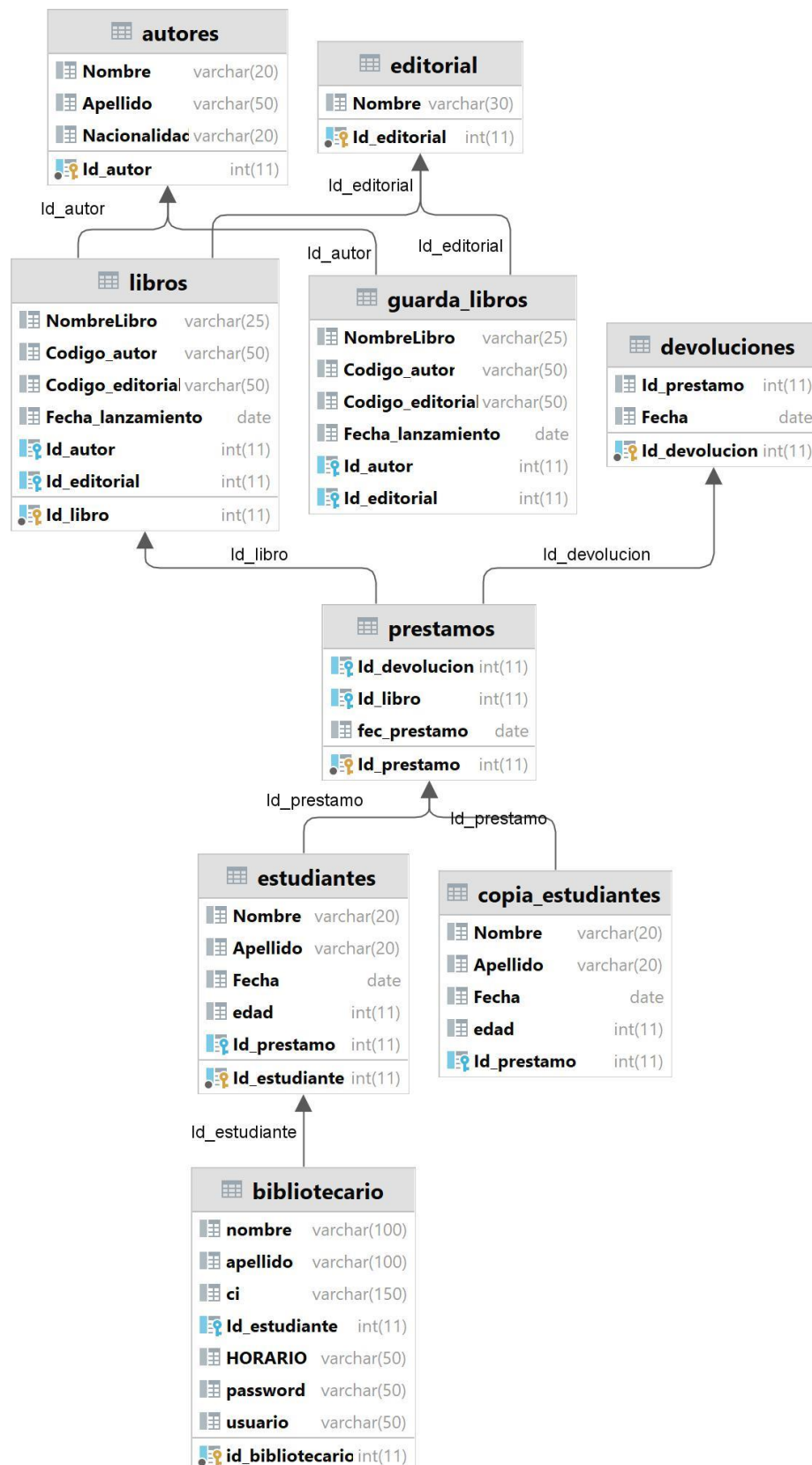
create or replace trigger genera_password_usuario
before insert on bibliotecario
for each row                                     #FOR EACH=PARA CADA FILA
before = despues
begin
    #substring de cada dos palabras iniciales
    set
new.usuario=concat(substring(new.nombre,1,2), substring(new.apellido,1,2));
    set new.password =
concat(substring(new.nombre,1,2), substring(new.apellido,1,2), substring(new.ci,1,2
));
end;

insert into bibliotecario(id_bibliotecario, nombre, apellido, ci, Id_estudiante,
HORARIO)
values (3, 'AURON', 'TORREZ', '8U482LP', 9, '8am-14pm');

select*
from bibliotecario;

```

### 3.2.3 Modelo Lógico



### 3.2.3 Modelo Lógico-Vistas



### 3.3 Usabilidad

#### Consultas-Resultado

*#mostraremos los nombres y apellidos de los estudiantes que se prestaron hasta la fecha de hoy*

	Nombre	Apellido	edad	fec_prestamo
1	MARCO ANTONIO	CALLE VAQUIATA	20	2022-09-09
2	IRIS	MISHEL VELASCO	19	2022-09-07
3	ILIA	SARZO	15	2022-09-05
4	JHON	TORREZ	22	2022-08-20
5	EXTERMINADOR	JUICIO FINAL	25	2022-08-18
6	ROSARIO	KANTUTA	13	2022-08-20
7	JOEL	LOPEZ	24	2022-09-07

#mostraremos todos los datos de los libros que tienen el editorial='NATURA'

	nombre	nombrelibro	codigo_autor	codigo_editorial	fecha_lanzamiento
1	NATURA	LAS MIL Y UNA NOCHES	123LMN	12345678	1543-11-27
2	NATURA	GRANDES ESPERANZAS	123ESP	000111222	2000-01-21

#Mostraremos los nombres de los autores mas la fecha de prestamo que hicieron los estudiantes mayores a 20 años

	AUTORES	NombreLibro	fec_prestamo	Nombre	Apellido	edad
1	ALFONSINA	FICCIONES	2022-08-20	JHON	TORREZ	22
2	ISABEL	GRANDES ESPERANZAS	2022-08-18	EXTERMINADOR	JUICIO FINAL	25
3	ALFONSINA	DECAMERON	2022-09-07	JOEL	LOPEZ	24
4	GABRIEL	Los cuentos de Canterbury	2022-09-05	EVER	ROJAS	28

#Mostrame todos los datos del bibliotecario que presto a estudiantes menores a 15 años

#adicionalmente mostrame a que estudiantes presto(solo nombres y edad)

	b.nombre	apellido	ci	HORARIO	estudiantes.Nombre	edad
1	ROCKY	SILVESTRE	12326CBB	8AM-14PM	ROSARIO	13

#mostrame los nombres de los estudiantes y que libro mas la fecha en que se les presto y en la que devolvieron

	Nombre	NombreLibro	fec_prestamo	Fecha
1	MARCO ANTONIO	VIAJE AL FIN DE LA NOCHE	2022-09-09	2022-12-11
2	IRIS	DECAMERON	2022-09-07	2022-12-07
3	ILIA	Los cuentos de Canterbury	2022-09-05	2022-12-05
4	JHON	FICCIONES	2022-08-20	2022-11-22
5	EXTERMINADOR	GRANDES ESPERANZAS	2022-08-18	2022-11-18
6	ROSARIO	FICCIONES	2022-08-20	2022-11-22
7	JOEL	DECAMERON	2022-09-07	2022-12-07

## FUNCIONES

*#Crearemos una funcion donde devuelva todos los datos del estudiante de mayor edad*

	Id_estudiante	Nombre	Apellido	Fecha	edad	Id_prestamo
1	5	EXTERMINADOR	JUICIO FINAL	1999-06-06	25	5

*#Crearemos una funcion donde devuelva todos los datos del estudiante de menor edad*

	Id_estudiante	Nombre	Apellido	Fecha	edad	Id_prestamo
1	6	ROSARIO	KANTUTA	2011-10-21	13	4

*#Crearemos un funcion que devuelva todos lo datos del estudiante segun su nombre y fecha de nacimiento*

*#la funcion debe de recibir dos parametros*

	id_estudiante	nombre	apellido	fecha	id_prestamo	edad
1	4	JHON	TORREZ	2003-03-21	4	22

*#Crearemos una funcion que busque su fecha de lanzamiento segun el nombre y su autor*

	`buscar_libro('Don Quijote de la Mancha','MARIO')`
1	1605-07-23

*#crear un funcion que permita concatenar nombres y apellidos de la tabla estudiantes*

	datos_de_estudiantes
1	Nombres: MARCO ANTONIO Apellidos: CALLE VAQUIATA
2	Nombres: IRIS Apellidos: MISHEL VELASCO
3	Nombres: ILIA Apellidos: SARZO
4	Nombres: JHON Apellidos: TORREZ
5	Nombres: EXTERMINADOR Apellidos: JUICIO FINAL
6	Nombres: ROSARIO Apellidos: KANTUTA
7	Nombres: JOEL Apellidos: LOPEZ



#Crear un Funcion con la condicionante when, then

	`ejercicio_5()`
1	Usuario Admin

#crear una serie del 1 al 10

	`ejercicio_6(10)`
1	1,2,3,4,5,6,7,8,9,10,

## VISTAS

#utilizamos la primera consulta para la creacion de una vista

	Nombre	Apellido	edad	fec_prestamo
1	MARCO ANTONIO	CALLE VAQUIATA	20	2022-09-09
2	IRIS	MISHEL VELASCO	19	2022-09-07
3	ILIA	SARZO	15	2022-09-05
4	JHON	TORREZ	22	2022-08-20
5	EXTERMINADOR	JUICIO FINAL	25	2022-08-18
6	ROSARIO	KANTUTA	13	2022-08-20
7	JOEL	LOPEZ	24	2022-09-07

#utilizamos la segunda consulta para la creacion de una vista

	nombre	id_libro	nombrelibro	codigo_autor	codigo_editorial	fecha_lanz
1	NATURA	4	LAS MIL Y UNA NOCHES	123LMN	12345678	1543-11-27
2	NATURA	8	GRANDES ESPERANZAS	123ESP	000111222	2000-01-21

#utilizamos la tercera consulta para la creacion de una vista

	AUTORES	NombreLibro	fec_prestamo	Nombre	Apellido	edad
1	JOSE	VIAJE AL FIN DE LA NOCHE	2022-09-09	MARCO ANTONIO	CALLE VAQUIATA	20
2	ALFONSINA	DECAMERON	2022-09-07	IRIS	MISHEL VELASCO	19
3	ALFONSINA	FICCIONES	2022-08-20	JHON	TORREZ	22
4	ISABEL	GRANDES ESPERANZAS	2022-08-18	EXTERMINADOR	JUICIO FINAL	25
5	ALFONSINA	DECAMERON	2022-09-07	JOEL	LOPEZ	24

```
#crearemos una vista donde tendremos que concatenar el nombre y apellido y tiene
que decir full_name
# donde la edad sera years
# donde los que no devolvieron el libro hasta la fecha actual se dira 'DEUDOR'
```

	full_name	years	fecha_en_la_que_devolvio
1	MARCO ANTONIO-CALLE VAQUIATA	20	DEUDOR
2	IRIS-MISHEL VELASCO	19	DEUDOR
3	ILIA-SARZO	15	<null>
4	JHON-TORREZ	22	<null>
5	EXTERMINADOR-JUICIO FINAL	25	<null>
6	ROSARIO-KANTUTA	13	<null>
7	JOEL-LOPEZ	24	DEUDOR

```
#Creamos una vista que te muestre todos los datos de los estudiantes y que si
#son mayores o igual a 18 años ponerlos 'mayor de edad' y si son menores a 18
años poner 'menor de edad'
```

	NOMBRE_COMPLETO	EDAD_DE_LOS_ESTUDIANTES	Fecha
1	MARCO ANTONIO CALLE VAQUIATA	mayor de edad	2002-11-13
2	IRIS MISHEL VELASCO	mayor de edad	2002-11-11
3	ILIA SARZO	menor de edad	2006-07-08
4	JHON TORREZ	mayor de edad	2003-03-21
5	EXTERMINADOR JUICIO FINAL	mayor de edad	1999-06-06
6	ROSARIO KANTUTA	menor de edad	2011-10-21
7	JOEL LOPEZ	mayor de edad	2002-07-13

## TRIGGERS

```
#EN ESTA PARTE HAREMOS USO DE LOS TRIGERRS
```

```
#HAREMOS DOS ADUTORIAS
```

	Nombre	Apellido	Fecha	edad	Id_prestamo
1	JOEL	LOPEZ	2002-07-13	24	2

	NombreLibro	Codigo_autor	Codigo_editorial	Fecha_lanzamiento	Id_autor
1	LA TIERRA PERDIDA	1234HGF123	D123D1	1781-09-22	3

#HAREMOS UN TRIGGERS DE VERIFICACION

	👤 id_bibliotecario ▾	👤 nombre ▾	👤 apellido ▾	👤 ci ▾	👤 Id_estudiante ▾	👤 HORARIO ▾	👤 password ▾
1	1	JHON	TRAVOLTA	123456LP	1	8AM-14PM	<null>
2	2	JIMENA	LAURA	654321CBB	2	14PM-8PM	<null>
3	3	AURON	TORREZ	8U482LP	9	8am-14pm	AUT08U
4	4	ROCKY	SILVESTRE	12326CBB	6	8AM-14PM	<null>

### 3.4 Video De La Funcionalidad Del Sistema

Link:[https://www.facebook.com/100082796172894/videos/1660123344390958/?notif\\_id=1670296462344456&notif\\_t=video\\_processed&ref=notif](https://www.facebook.com/100082796172894/videos/1660123344390958/?notif_id=1670296462344456&notif_t=video_processed&ref=notif)

## **CAPITULO IV**

### **4.1 Conclusiones**

Luego de haber concluido este trabajo de bases de datos sobre la biblioteca db\_biblioteca fueron muchos los esfuerzos y conocimientos adquiridos durante el semestre y que fueron plasmados en este proyecto.

Algunos de los aspectos aprendidos y que de gran peso es la base de datos su definición, requerimiento, ventajas y características donde podemos decir que la base de datos:

Es una colección de datos o información usados para facilitar la informacion al usuario y tambien tener una mejor organización.

Con la implementación de esta base de datos a la biblioteca a tener un mejor control en sus procesos.

- Ayudaremos a tener una catalogación correcta.

- Ayudará al bibliotecario a tener datos de una manera más rápida con información real.