**vistas**

use biblioteca\_dap;

CREATE VIEW

libros\_disponibles AS

SELECT titulo FROM libro;

CREATE VIEW

cantidad\_de\_libros AS

SELECT libro.titulo, inventario.cantidad\_disponible FROM libro, inventario;

CREATE VIEW

autores\_disponibles AS

SELECT autor FROM libro;

CREATE VIEW

libro\_y\_autor AS

SELECT titulo, autor FROM libro;

CREATE VIEW

generos\_disponibles AS

SELECT genero FROM libro;

CREATE VIEW

editoriales\_disponibles AS

SELECT editorial FROM libro;

CREATE VIEW

libros\_info AS

SELECT libro.ID\_libro, libro.titulo, libro.autor, libro.genero, libro.editorial, libro.fecha\_publicacion, inventario.cantidad\_disponible

FROM libro, inventario;

CREATE VIEW

roles\_info AS

SELECT \* FROM rol;

CREATE VIEW

estados\_SP\_info AS

SELECT \* FROM estado;

CREATE VIEW

inventario\_info AS

SELECT \* FROM inventario;

CREATE VIEW

usuarios\_info AS

SELECT \* FROM usuario;

CREATE VIEW

usuario\_y\_prestamos AS

SELECT usuario.ID\_usuario, usuario.nombre\_usuario, usuario.apellido\_usuario, usuario.documento, solicitud\_prestamo.ID\_solicitud\_prestamo

FROM usuario inner join solicitud\_prestamo on usuario.ID\_usuario=solicitud\_prestamo.ID\_solicitud\_prestamo

inner join libro on solicitud\_prestamo.ID\_libro= libro.ID\_libro;

CREATE VIEW

sanciones\_info AS

SELECT \* FROM sanciones;

CREATE VIEW

prestamo\_info AS

SELECT \* FROM solicitud\_prestamo;

CREATE VIEW

libros\_no\_disponibles AS

SELECT libro.ID\_libro, libro.titulo, libro.autor, inventario.cantidad\_bruta

FROM libro, inventario

where inventario.cantidad\_bruta=0;

CREATE VIEW

libros\_disponibles\_ordenado\_cantidad AS

SELECT libro.ID\_libro, libro.titulo, libro.autor, inventario.cantidad\_bruta

FROM libro, inventario order by inventario.cantidad\_bruta desc;

create view administradores

as select usuario.ID\_usuario, usuario.nombre\_usuario, usuario.apellido\_usuario, usuario.rol from usuario where rol=1;

create view directivos

as select usuario.ID\_usuario, usuario.nombre\_usuario, usuario.apellido\_usuario, usuario.rol from usuario where rol=2;

create view estudiantes

as select usuario.ID\_usuario, usuario.nombre\_usuario, usuario.apellido\_usuario, usuario.rol from usuario where rol=3;

**consultas multitabla**

use biblioteca\_dap;

SELECT libro.titulo, inventario.cantidad\_disponible, inventario.cantidad\_bruta FROM libro, inventario;

SELECT libro.ID\_libro, libro.titulo, libro.autor, libro.genero, libro.editorial, libro.fecha\_publicacion, inventario.cantidad\_bruta

FROM libro, inventario;

SELECT usuario.ID\_usuario, usuario.nombre\_usuario, usuario.apellido\_usuario, usuario.documento, solicitud\_prestamo.ID\_solicitud\_prestamo

FROM usuario inner join solicitud\_prestamo on usuario.ID\_usuario=solicitud\_prestamo.ID\_solicitud\_prestamo

inner join libro on solicitud\_prestamo.ID\_libro= libro.ID\_libro;

SELECT libro.ID\_libro, libro.titulo, libro.autor, inventario.cantidad\_disponible

FROM libro, inventario

where inventario.cantidad\_disponible=0;

SELECT libro.ID\_libro, libro.titulo, libro.autor, inventario.cantidad\_bruta

FROM libro, inventario order by inventario.cantidad\_bruta desc;

SELECT libro.ID\_libro, libro.titulo, libro.autor, inventario.cantidad\_bruta

FROM libro, inventario order by inventario.cantidad\_bruta asc;

SELECT libro.ID\_libro, libro.titulo, libro.autor, inventario.ID\_libro, inventario.cantidad\_disponible, inventario.estado from libro, inventario;

**TRIGGERS**

create table libro\_aux

(

ID\_libro\_aux int not null auto\_increment primary key,

titulo varchar(50)

)

drop trigger t\_actualizar\_libro;

delimiter $$

create trigger t\_actualizar\_libro

after insert on libro for each row

begin

update libro\_aux set titulo= new.titulo;

end$$

delimiter ;

create table usuario\_aux

(

ID\_usuario\_aux int not null auto\_increment primary key,

nombre varchar(50)

)

drop trigger t\_actualizar\_usuario;

delimiter $$

create trigger t\_actualizar\_usuario

after insert on usuario for each row

begin

update usuario\_aux set nombre= new.nombre\_usuario;

end$$

delimiter ;

delimiter $$

CREATE TRIGGER t\_insertar\_usuario

BEFORE INSERT ON usuario

FOR EACH ROW

BEGIN

INSERT INTO usuario\_aux

(ID\_usuario\_aux, nombre)

VALUES(NEW.ID\_usuario, NEW.nombre\_usuario);

END $$

delimiter ;

delimiter $$

CREATE TRIGGER t\_insertar\_libro

BEFORE INSERT ON libro

FOR EACH ROW

BEGIN

INSERT INTO libro\_aux

(ID\_libro\_aux, titulo)

VALUES(NEW.ID\_libro, NEW.titulo);

END $$

delimiter ;

**Proc alm**

use biblioteca\_dap;

drop procedure sp\_actualizar\_usuario;

delimiter $$

create procedure sp\_actualizar\_usuario

(

in parid int,

in nombre varchar(50),

in apellido varchar (50),

in contrasena varchar(50),

in documento varchar (30),

in fecha\_nacimiento date,

in correo varchar (50),

in telefono varchar (30),

in rol varchar(10)

)

begin

UPDATE Usuario SET

ID\_usuario=parid,

nombre\_usuario=nombre,

apellido\_usuario=apellido,

contrasena=contrasena,

documento=documento,

fecha\_nacimiento=fecha\_nacimiento,

correo=correo,

telefono=telefono,

rol=rol

where ID\_usuario=parid;

end$$

delimiter ;

call sp\_actualizar\_usuario (7,'Jimin','Park','teamominyoongi','123456333','1995-10-13','jiminp','1234577','3');

drop procedure sp\_actualizar\_libro;

delimiter $$

create procedure sp\_actualizar\_libro

(

in pair int,

in titulo varchar(50),

in autor varchar (50),

in genero varchar(50),

in editorial varchar (30),

in fecha\_public date

)

begin

UPDATE libro SET

ID\_libro=pair,

titulo=titulo,

autor=autor,

genero=genero,

editorial=editorial,

fecha\_publicacion=fecha\_public

where ID\_libro=pair;

end$$

delimiter ;

call sp\_actualizar\_libro (2,'Cien Anos De Soledad','Gabriel Garcia Marquez','A','Planeta','1995-10-13');

DROP PROCEDURE sp\_eliminar\_usuario;

DELIMITER $$

CREATE PROCEDURE sp\_eliminar\_usuario

(

parid int

)

BEGIN

delete from usuario where ID\_usuario=parid;

END $$

DELIMITER ;

call sp\_eliminar\_usuario (6);

DROP PROCEDURE sp\_eliminar\_libro;

DELIMITER $$

CREATE PROCEDURE sp\_eliminar\_libro

(

parid int

)

BEGIN

delete from libro where ID\_libro=parid;

END $$

DELIMITER ;

call sp\_eliminar\_libro (2);

drop procedure sp\_salida\_libro;

delimiter $$

create procedure sp\_salida\_libro

(

in ID\_libro int,

inout cantidad\_disponible varchar (50)

)

begin

select cantidad\_disponible into cantidad\_disponible

from inventario

where cantidad\_disponible=cantidad\_disponible;

end $$

delimiter ;

set @existencia=0;

call sp\_salida\_libro (1,@existencia);

select @existencia;

drop procedure sp\_salida\_usuario;

delimiter $$

create procedure sp\_salida\_usuario

(

in ID int,

out contrasena varchar (50)

)

begin

select contrasena into contrasena

from usuario

where contrasena=contrasena;

end $$

delimiter ;

set @contrasena=0;

call sp\_salida\_usuario(2,@contrasena);

select @contrasena;