



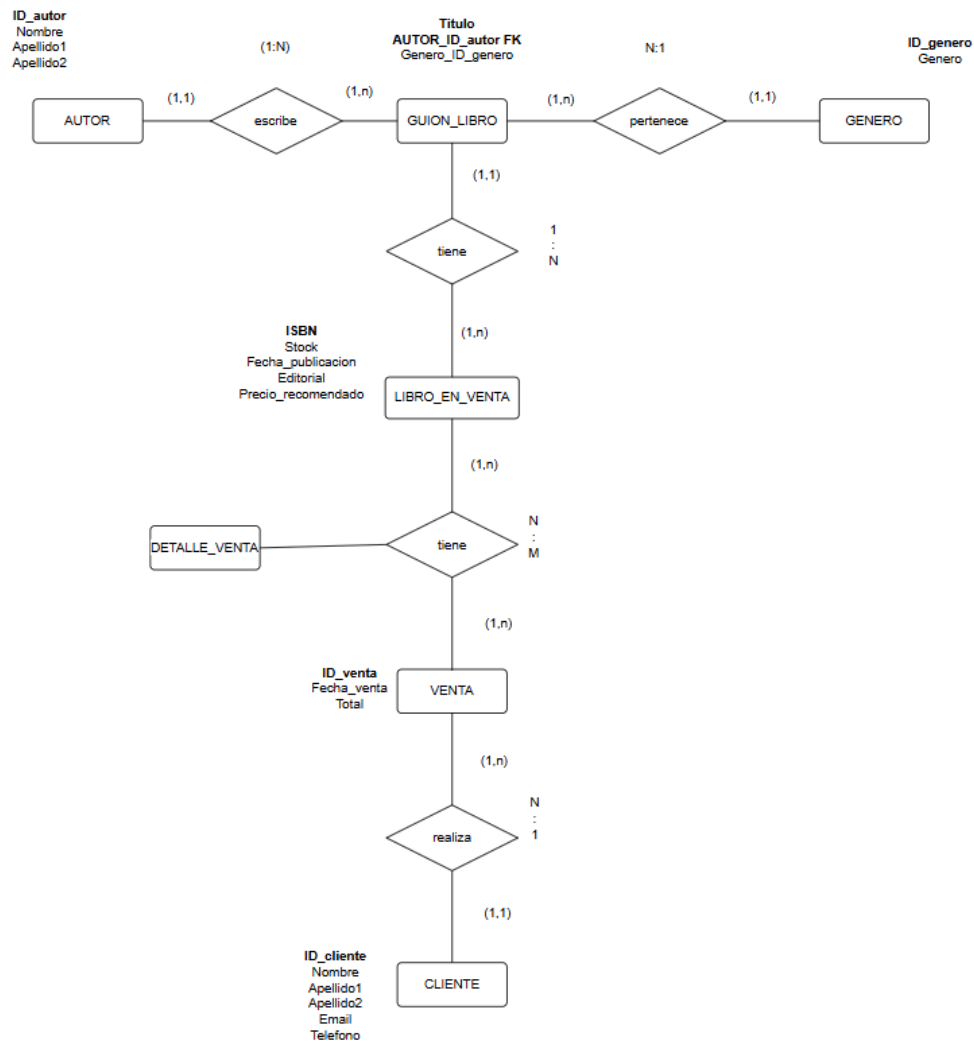
ÍNDICE

Introducción	2
Modelo Entidad Relación	2
Modelo Relacional	3
Carga masiva	4
Consultas	4
Una consulta de una tabla con where	4
Una consulta de más de una tabla	5
Una consulta con agrupación	5
Una consulta con sub consultas	6
Una que combine varias anteriores	7
Vistas	8
Vista consulta de más de una tabla	8
Vista consulta con agrupación	8
Funciones	9
Función de cálculo	9
Función de conteo	10
Procedimientos	11
Procedimiento de inserción	11
Procedimiento de consulta	12
Procedimiento de actualización	13
Triggers	14
Trigger de validación	14
Trigger de auditoría	15
GitHub	16
AWS	16
Conclusión	16

Introducción

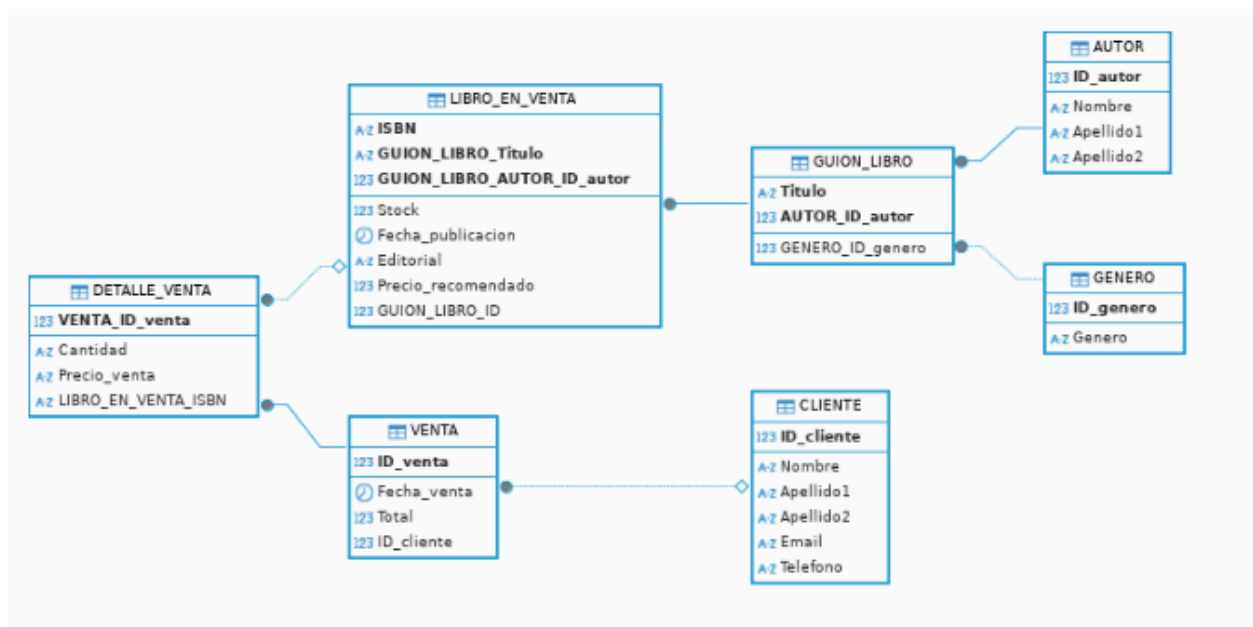
El proyecto consiste en el diseño de una base de datos para una pequeña librería de barrio. El sistema ayudará a organizar libros, autores, géneros, ventas y clientes de forma sencilla y eficiente. Esto permitirá mejorar el control de inventario y las ventas, facilitando las operaciones diarias y ofreciendo un mejor servicio a la comunidad.

Modelo Entidad Relación



He diseñado el diagrama de esta forma porque refleja cómo se relacionan las entidades en el sistema de gestión de libros. **AUTOR** se conecta con **GUION_LIBRO** mediante la relación **1:N escribe**, ya que un autor puede escribir varios libros, pero un libro solo puede tener un autor. **GÉNERO** está relacionado con **GUION_LIBRO** mediante la relación **N:1 pertenece**, ya que un género puede tener muchos libros y un libro solo tiene un género, **GUION_LIBRO** está relacionado con **LIBRO_EN_VENTA** mediante **1:N tiene**, ya que un libro tiene un guion y un guion puede tener más de un libro, **LIBRO_EN_VENTA** está relacionado con **VENTA** mediante **N:M tiene**, ya que un libro puede tener muchas ventas y una venta puede tener muchos libros, por último, **VENTA** está relacionado con **CLIENTE** mediante **N:1** ya que un cliente puede realizar muchas compras y una venta solo puede ser de un cliente.

Modelo Relacional



Carga masiva

Para la realización de la carga masiva utilicé mockaroo, fui poniendo el tipo de datos que deseaba por cada tabla y así fui llenando todas las tablas.

	ID cliente	A-Z Nombre	A-Z Apellido1	A-Z Apellido2	A-Z Email	A-Z Telefono
472	472	Burke	Puig	Youd	bbargeryd3@soup.io	+46 833 778 6196
473	473	Waneta	Huertas	Cluff	wseakesd4@zimbio.com	+7 526 917 9914
474	474	Obed	Villaverde	Beese	ocrockettd5@biglobe.ne.jp	+230 216 799 6453
475	475	Sly	Montaña	Tidbury	sagotttd6@parallels.com	+48 953 840 2563
476	476	Otto	Diez	McChesney	okirked7@gravatar.com	+62 815 509 6267
477	477	Lana	Pavón	Scarman	lbothend8@qq.com	+591 727 114 0727
478	478	Bertie	Peral	Bennike	bsargentd9@g.co	+237 638 924 3756
479	479	Tawnya	Montoya	Shoulders	tfeenanda@chicagotribune.com	+358 651 522 4510
480	480	Didi	Casares	Whellams	dsemperdb@4shared.com	+7 794 165 4295
481	481	Fax	Gonzalo	Foucar	fscullydc@hud.gov	+63 313 646 9300
482	482	Karilynn	Narváez	Waymont	kleivesleydd@php.net	+1 203 632 5584
483	483	Reube	Fiol	Crampsy	rcrootede@examiner.com	+48 198 219 1857
484	484	Grethel	Morell	Dunley	gstarmoredf@nature.com	+242 671 860 8931
485	485	Florinda	Lozano	Scandred	fetheridgedg@ask.com	+33 452 528 8418
486	486	Stefano	Viana	Tybalt	skeeridh@statcounter.com	+62 992 109 2928
487	487	Averyl	Montoya	Billingsly	airdaledi@alexa.com	+86 654 468 7651
488	488	Maure	Cortes	Voaden	mcannawaydj@csmonitor.com	+256 659 283 7408
489	489	Madelena	Bonet	Hucker	medlingtondk@alibaba.com	+86 948 935 4317
490	490	Kelsey	Navarro	Zarb	ksevillel@bizjournals.com	+62 223 718 0831
491	491	Torre	Gámez	Tankus	tmlsteadm@cisco.com	+62 424 554 5134
492	492	Tabb	Uribe	Johantges	tswinyarddn@springer.com	+66 163 654 3674
493	493	Kandy	Frutos	Allum	kanthilldo@businesswire.com	+48 270 477 4181
494	494	Fredi	Losa	Welbrock	fmanthroppedp@icq.com	+46 524 577 4562
495	495	Jodie	Esteban	Nicklen	jsilmondq@geocities.com	+86 240 177 3014
496	496	Bunni	Corbacho	Moorfield	bmackellardr@nature.com	+46 350 739 6396
497	497	Ethelin	Castro	Lambirth	emcsparands@nps.gov	+86 712 740 0204

Consultas

Una consulta de una tabla con where

- Esta consulta busca a todos los clientes cuyo nombre sea Lana.

SELECT *

FROM CLIENTE **c**

WHERE c.Nombre = 'Lana';

CLIENTE 1 X						
SELECT * FROM CLIENTE WHERE Nombre <small>Enter a SQL expression to filter results (use Ctrl+Space)</small>						
	ID cliente	A-Z Nombre	A-Z Apellido1	A-Z Apellido2	A-Z Email	A-Z Telefono
1	477	Lana	Pavón	Scarman	lbothend8@qq.com	+591 727 114 0727

Una consulta de más de una tabla

- Esta consulta une las tablas ventas y cliente y luego filtra las ventas con un importe mayor al 100.

```
SELECT v.ID_venta , c.Nombre , c.Apellido1 , c.Apellido2 , v.Total
from VENTA v inner join CLIENTE c
on v.ID_cliente = c.ID_cliente
where v.Total > 100;
```

SQL Query: `SELECT v.ID_venta , c.Nombre , c.Apellido1 , c.Apellido2 , v.Total`

	ID venta	Nombre	Apellido1	Apellido2	Total
1	1	Danyelle	Pedraza	Sondon	203
2	2	Evelina	Morán	Tabour	110
3	3	Gilberta	Alcolea	Durnan	238
4	4	Ethelin	Castro	Lambirth	250
5	6	Jodie	Esteban	Nicklen	134
6	7	Fredi	Losa	Welbrock	179
7	9	Tabb	Uribe	Johantges	231
8	11	Kelsey	Navarro	Zarb	236
9	12	Madelena	Bonet	Hucker	217
10	13	Maure	Cortes	Voaden	204

Una consulta con agrupación

- Esta consulta cuenta cuántos libros hay de cada género en la tabla guion_libro, sólo muestra los géneros que tienen más de 5 libros.

```
SELECT g.Genero , COUNT(*) as Total_Libros
FROM GUION_LIBRO gl inner join GENERO g
on gl.GENERO_ID_genero = g.ID_genero
group by g.Genero
HAVING COUNT(*) > 5;
```

SQL Query: `SELECT g.Genero , COUNT(*) as Total_Libros`

	Genero	Total Libros
1	Ficción	94
2	No Ficción	94
3	Fantasía	103
4	Ciencia Ficción	109
5	Historia	100

Una consulta con sub consultas

- Esta consulta encuentra los nombres y apellidos de los clientes que han realizado al menos una venta con un total superior al promedio de todas las ventas.

```
SELECT c.Nombre , CONCAT_WS(' ', c.Apellido1, c.Apellido2) as Apellidos
FROM CLIENTE c
where c.ID_cliente IN (
```

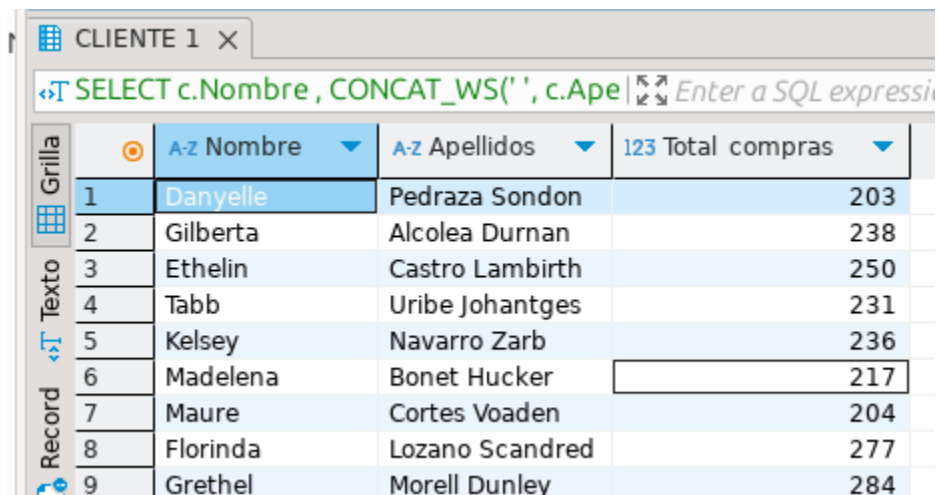
```
SELECT ID_cliente
FROM VENTA v
WHERE v.Total > (
SELECT AVG(v.Total)
FROM VENTA v));
```

CLIENTE 1 X		
SELECT c.Nombre , CONCAT_WS(' ', c.Ape		
Grilla	A-z Nombre	A-z Apellidos
1	Danyelle	Pedraza Sondon
2	Gilberta	Alcolea Durnan
3	Ethelin	Castro Lambirth
4	Fredi	Losa Welbrock
5	Tabb	Uribe Johantges
6	Kelsey	Navarro Zarb
7	Madelena	Bonet Hucker
8	Maure	Cortes Voaden
9	Florinda	Lozano Scandred
10	Grethel	Morell Dunley
11	Karilun	Manríez Woumont

Una que combine varias anteriores

- Esta consulta muestra los clientes que han gastado más de 200 en total desde el 1 de enero de 2024.

```
SELECT c.Nombre , CONCAT_WS(' ' , c.Apellido1, c.Apellido2) as Apellidos, SUM(v.Total) as
Total_compras
FROM CLIENTE c inner join VENTA v
on c.ID_cliente = v.ID_cliente
where v.Fecha_venta >= '2024-01-01'
GROUP BY c.Nombre , Apellidos
HAVING SUM(v.Total) >200;
```



	A-z Nombre	A-z Apellidos	123 Total compras
1	Danyelle	Pedraza Sondon	203
2	Gilberta	Alcolea Durnan	238
3	Ethelin	Castro Lambirth	250
4	Tabb	Uribe Johantges	231
5	Kelsey	Navarro Zarb	236
6	Madelena	Bonet Hucker	217
7	Maure	Cortes Voaden	204
8	Florinda	Lozano Scandred	277
9	Grethel	Morell Dunley	284

Vistas

Vista consulta de más de una tabla

-- Clientes que han realizado compras mayores a 100

create view vista_clientes_compras **as**

select v.id_venta, c.nombre, c.apellido1, c.apellido2, v.total

from VENTA v

inner join CLIENTE c **on** v.id_cliente = c.id_cliente

where v.total > 100;

ID venta	Nombre	Apellido1	Apellido2	Total
1	Danyelle	Pedraza	Sondon	203
2	Evelina	Morán	Tabour	110
3	Gilberta	Alcolea	Durnan	238
4	Ethelin	Castro	Lambirth	250
5	Jodie	Esteban	Nicklen	134
6	Fredi	Losa	Welbrock	179
7	Tabb	Uribe	Johantges	231
8	Kelsey	Navarro	Zarb	236
9	Madelena	Bonet	Hucker	217
10	Maure	Cortes	Voaden	204

Vista consulta con agrupación

-- Total de libros por género con más de 5 libros

create view vista_libros_por_genero **as**

select g.genero, **count**(*) **as** total_libros

from GUION_LIBRO gl

inner join GENERO g **on** gl.GENERO_ID_genero = g.ID_genero

group by g.genero

having **count**(*) > 5;

Genero	Total_Libros
Ficción	94
No Ficción	94
Fantasía	103
Ciencia Ficción	109
Historia	100

Funciones

Función de cálculo

-- Obtener el total gastado por un cliente

delimiter \$\$

create function totalgastadoporcliente(cliente_id **int**) **returns decimal**(10,2) **deterministic**

begin

declare total **decimal**(10,2);

select sum(v.total) **into** total **from** VENTA v **where** v.id_cliente = cliente_id;

return ifnull(total, 0);

end \$\$

delimiter ;

The screenshot shows a SQL IDE interface. At the top, there's a script editor with the following SQL code:

```
use libreria;  
select totalgastadoporcliente(1);
```

Below the script editor, there's a results pane titled "Resultados 1". It shows a table with one row and one column. The column header is "123 totalgastadoporcliente(1)" and the value in the row is "279".

123 totalgastadoporcliente(1)
279

Función de conteo

-- Obtener la cantidad de ventas realizadas en un mes específico

delimiter \$\$

create function ventasenmes(año **int**, mes **int**) **returns int deterministic**

begin

declare total **int**;

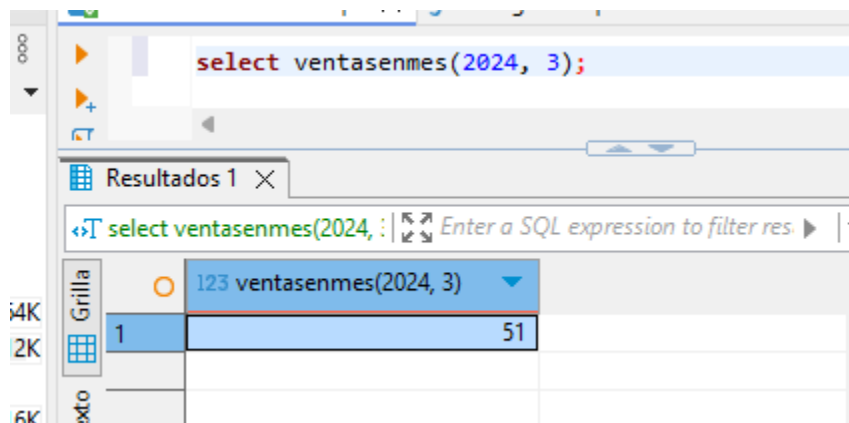
select count(*) **into** total **from** VENTA v **where year**(v.fecha_venta) = año **and**

month(v.fecha_venta) = mes;

return total;

end \$\$

delimiter ;



Results 1

	123 ventasenmes(2024, 3)
1	51

Procedimientos

Procedimiento de inserción

-- Insertar un nuevo cliente

delimiter \$\$

create procedure insertarcliente(**in** id_cliente **int**, **in** nombre **varchar**(50), **in** apellido1 **varchar**(50),
in apellido2 **varchar**(50), **in** email **varchar**(100), **in** telefono **varchar**(20))

begin

insert into CLIENTE(ID_cliente, nombre, apellido1, apellido2, email, telefono)

values (id_cliente, nombre, apellido1, apellido2, email, telefono);

end \$\$

delimiter ;

The screenshot shows a database IDE with a query editor at the top containing the command: `call insertarcliente(1001, 'Juan', 'Pérez', 'Gómez', 'juan.perez@email.com', '6557768125');`. Below the editor, a statistics window titled "Estadísticas 1" displays the following data:

Name	Value
Updated Rows	1
Execute time	0.197s
Start time	Tue Apr 01 20:38:43 CEST 2025
Finish time	Tue Apr 01 20:38:43 CEST 2025
Query	call insertarcliente(1001, 'Juan', 'Pérez', 'Gómez', 'juan.perez@email.com', '6557768125')

At the bottom, a table grid shows the result of the procedure. The first row is highlighted in blue and contains the values: 1000, 1,000, wnitaker, and Pérez.

Record	1000	1,000	wnitaker	Pérez
1001	1,001		Juan	

Procedimiento de consulta

-- Consultar total gastado por un cliente

delimiter \$\$

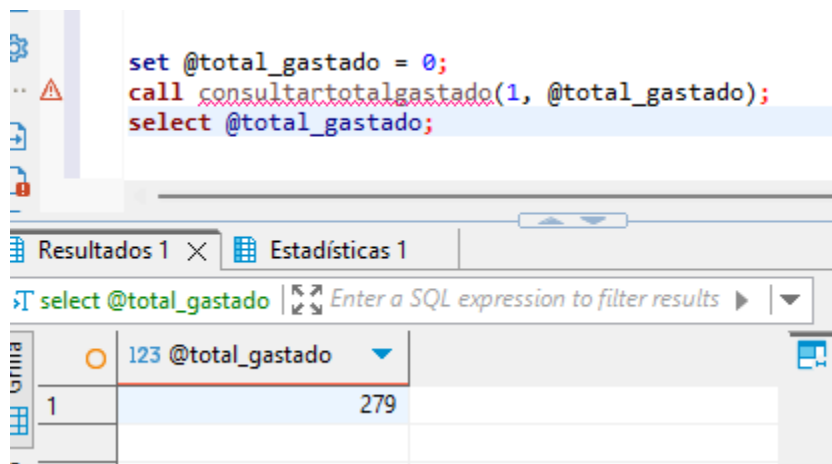
create procedure consultartotalgastado(**in** cliente_id **int**, **out** total **decimal**(10,2))

begin

set total = totalgastadoporcliente(cliente_id);

end \$\$

delimiter ;



Procedimiento de actualización

-- Actualizar el email de un cliente dado su id

delimiter \$\$

create procedure actualizaremailcliente(**in** cliente_id **int**, **in** nuevo_email **varchar**(100))

begin

update CLIENTE **set** email = nuevo_email **where** id_cliente = cliente_id;

end \$\$

delimiter ;

The screenshot shows a SQL query being executed in a database IDE. The query is:

```
call actualizaremailcliente(1, 'carrasco@email.com');
```

Below the query editor, a window titled "Estadísticas 1" displays the execution statistics:

Name	Value
Updated Rows	1
Execute time	0.203s
Start time	Tue Apr 01 20:23:15 CEST 2025
Finish time	Tue Apr 01 20:23:15 CEST 2025
Query	call actualizaremailcliente(1, 'carrasco@email.com')

	Apellido1	A-Z Apellido2	A-Z Email	A-Z Te
1	co	Gozzard	carrasco@email.co	+86 5

Triggers

Trigger de validación

-- Evitar ventas con total negativo

delimiter \$\$

create trigger verificartotalventa **before insert on** VENTA

for each row

begin

if new.total < 0 **then**

signal sqlstate '45000' **set** message_text = 'el total de la venta no puede ser negativo';

end if;

end \$\$

delimiter ;

-- PRUEBA

insert into VENTA (ID_venta, Fecha_venta, Total, ID_cliente) **values** (1, '2024-03-05', -50, 1);

-- PRUEBA

```
insert into VENTA (ID_venta, Fecha_venta, Total, ID_cliente) values (1, '2024-03-05', -50, 1);
```

Execution Error



Error occurred during SQL query execution

Reason:

SQL Error [1644] [45000]: el total de la venta no puede ser negativo

Trigger de auditoría

-- Registrar historial de cambios en stock de libros

```
create table HISTORIAL_STOCK (
    id_historico INT auto increment primary key,
    isbn varchar(20),
    stock_anterior int,
    stock_nuevo int,
    fecha_cambio datetime
);
```

delimiter \$\$

create trigger historialstock **after update on** LIBRO_EN_VENTA

for each row

begin

```
    insert into HISTORIAL_STOCK (isbn, stock_anterior, stock_nuevo, fecha_cambio)
    values (old.isbn, old.stock, new.stock, now());
```

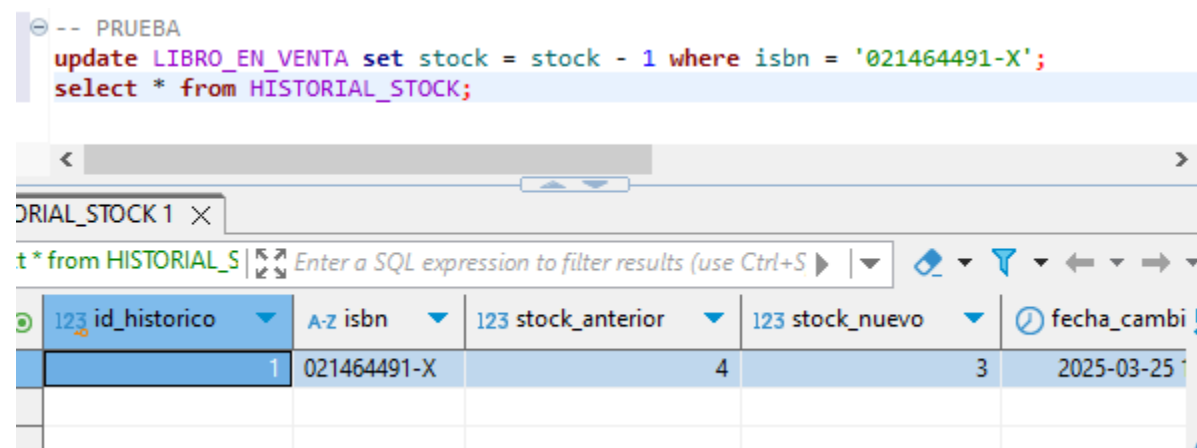
end \$\$

delimiter ;

-- PRUEBA

update LIBRO_EN_VENTA **set** stock = stock - 1 **where** isbn = '021464491-X';

select * from HISTORIAL_STOCK;



```
-- PRUEBA
update LIBRO_EN_VENTA set stock = stock - 1 where isbn = '021464491-X';
select * from HISTORIAL_STOCK;
```

123 id_historico	A-Z isbn	123 stock_anterior	123 stock_nuevo	fecha_cambi
1	021464491-X	4	3	2025-03-25



GitHub

Enlace a mi [GitHub](#)

AWS

18.205.133.168

Conclusión

Este proyecto me ha ayudado a mejorar mis conocimientos sobre bases de datos, especialmente en el diseño y la realización de consultas. He aprendido a estructurar la información de manera más eficiente y a utilizar herramientas como Mockaroo para generar datos.

Unos de los aspectos más difíciles fue asegurarme de que las relaciones entre las tablas fueran correctas y que las consultas funcionaran bien. Aun así, creo que el resultado ha sido bueno.

En el futuro, podría mejorar el rendimiento de las consultas y añadir más funciones al proyecto, como reportes más detallados o medidas de seguridad adicionales. En general, ha sido un trabajo útil para seguir aprendiendo.