

# PROYECTO BASE DE DATOS

## EL REFUGIO DE LAS LETRAS

13/12/2024

## Ander Burzaco Sierra

IES Alixar

1º DAW



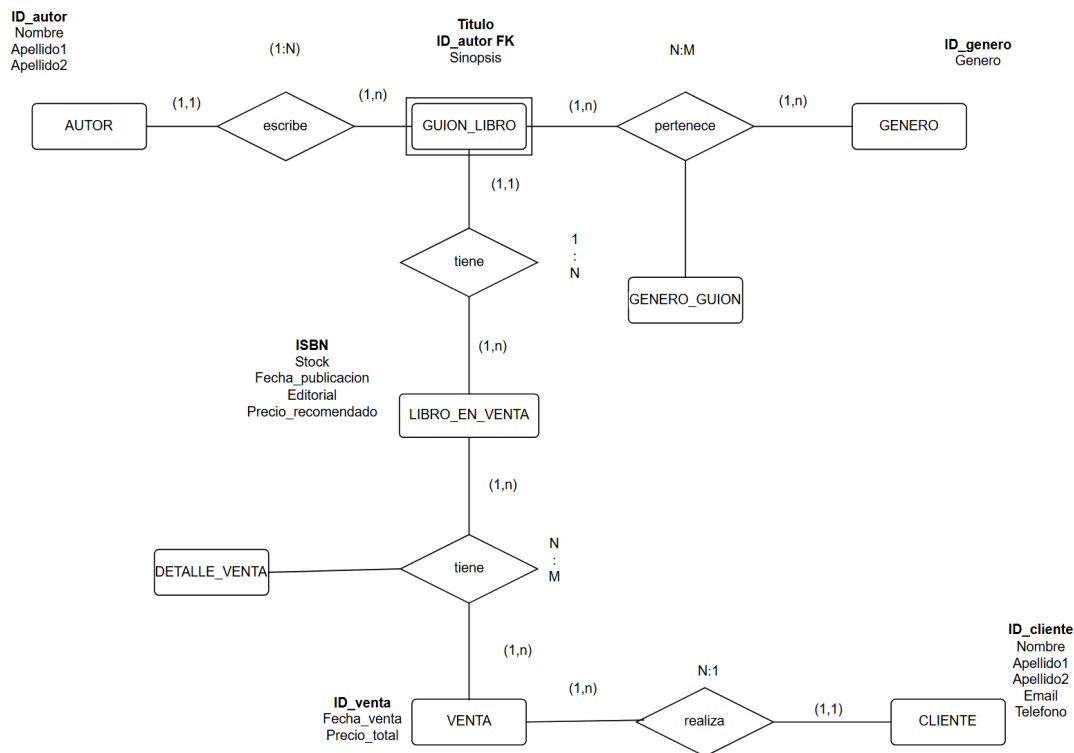
# ÍNDICE

<b>Introducción</b>	<b>2</b>
<b>Modelo Entidad Relación</b>	<b>2</b>
<b>Modelo Relación</b>	<b>3</b>
<b>Carga masiva</b>	<b>3</b>
<b>Consultas</b>	<b>4</b>
Consulta de control de stock agotado	4
Consulta de ranking de ventas por libro	4
Consulta de análisis de ingresos por fecha	5
Consulta de ingresos por autor	6
Consulta de clientes con más compras	7
<b>Vistas</b>	<b>8</b>
Vista de ranking de libros vendidos	8
Vista de autores con más ingresos	9
<b>Funciones</b>	<b>10</b>
Función de cálculo de stock actual	10
Función de obtención de nombre completo de cliente	11
<b>Procedimientos</b>	<b>12</b>
Procedimiento de inserción de libro	12
Procedimiento de actualización de stock	14
Procedimiento de consulta de detalles de venta	15
<b>Triggers</b>	<b>16</b>
Trigger de cálculo automático del subtotal	16
Trigger de validación de stock disponible	18
<b>GitHub</b>	<b>21</b>
<b>AWS</b>	<b>21</b>
<b>Conclusión</b>	<b>21</b>

## Introducción

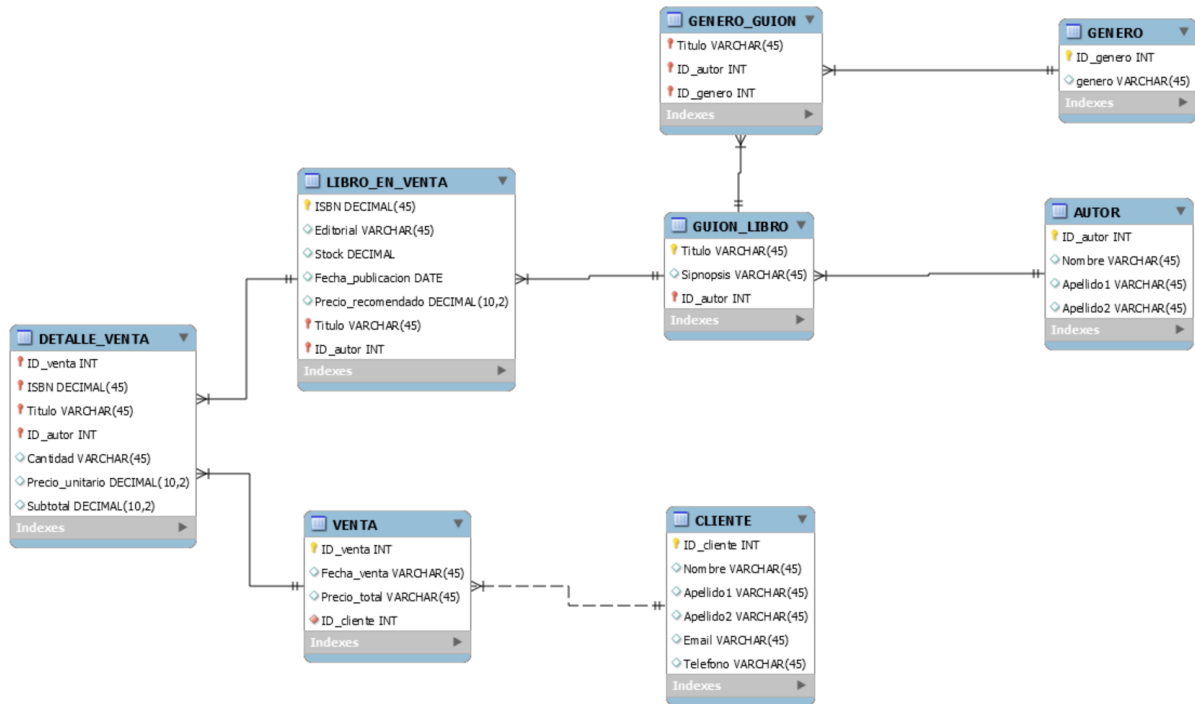
El proyecto consiste en el diseño de una base de datos para una pequeña librería de barrio. El sistema ayudará a organizar libros, autores, géneros, ventas y clientes de forma sencilla y eficiente. Esto permitirá mejorar el control de inventario y las ventas, facilitando las operaciones diarias y ofreciendo 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:M pertenece**, ya que un género puede tener muchos libros y un libro puede tener muchos géneros, **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 Relación



## 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	gstarsmoredf@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

## Consultas

### Consulta de control de stock agotado

-- Esta consulta nos muestra los libros que se han agotado

```
select /.ISBN, /.Titulo, /.Stock
from LIBRO_EN_VENTA / inner join DETALLE_VENTA dv
on /.ISBN = dv.ISBN
group by /.ISBN, /.Titulo, /.Stock
having /.Stock = 0;
```

-- Esta consulta nos muestra los libros que se han agotado

```
select L.ISBN, L.Titulo, L.Stock
from LIBRO_EN_VENTA L inner join DETALLE_VENTA dv
on L.ISBN = dv.ISBN
group by L.ISBN, L.Titulo, L.Stock
having L.Stock = 0;
```

LIBRO\_EN\_VENTA 1 ×

select L.ISBN, L.Titulo, L.Stock fr *Enter a SQL expression to filter results (use Ctrl+Space)*

	ISBN	Titulo	Stock
1	021464491-X	Singing Detective, The	0

### Consulta de ranking de ventas por libro

-- Esta consulta nos muestra los libros que más se han vendido

```
select dv.ISBN, lev.Titulo, sum(dv.cantidad) as total_vendido
from DETALLE_VENTA dv inner join LIBRO_EN_VENTA lev
on dv.ISBN = lev.ISBN
group by dv.ISBN, lev.Titulo
order by total_vendido desc
limit 5;
```

DETALLE\_VENTA 1 ×

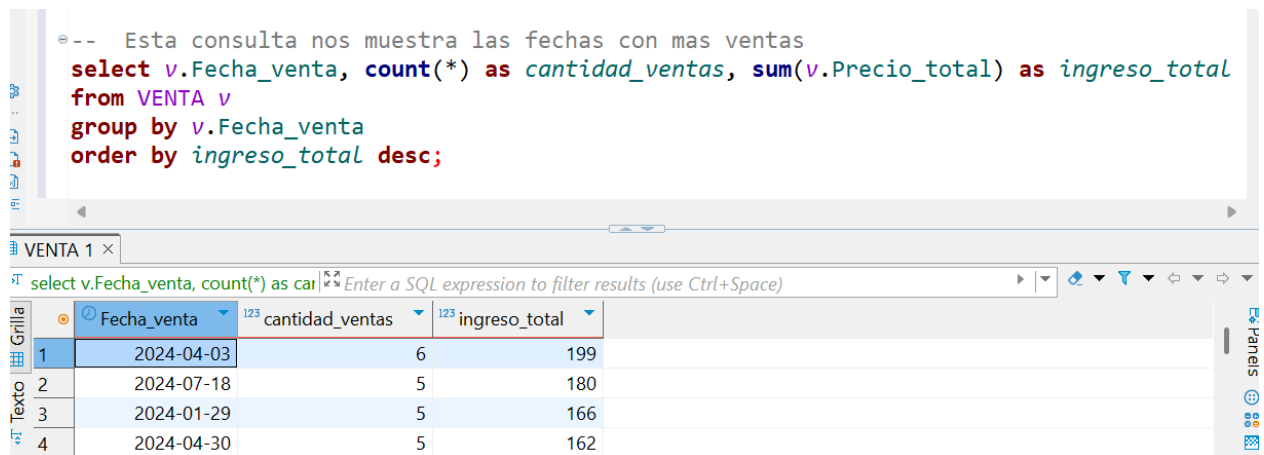
select dv.ISBN, dv.Titulo, sum(*Enter a SQL expression to filter results (use Ctrl+Space)*)

	ISBN	Titulo	total_vendido
1	349895053	Mission London	49
2	456481736-1	Some Body	6
3	034042694-2	Take, The	6

## Consulta de análisis de ingresos por fecha

-- Esta consulta nos muestra las fechas con más ventas

```
select v.Fecha_venta, count(*) as cantidad_ventas, sum(v.Precio_total) as
ingreso_total
from VENTA v
group by v.Fecha_venta
order by ingreso_total desc;
```



The screenshot shows a SQL IDE interface. The top pane contains a SQL query with a comment: "-- Esta consulta nos muestra las fechas con mas ventas". The query is: `select v.Fecha_venta, count(*) as cantidad_ventas, sum(v.Precio_total) as ingreso_total from VENTA v group by v.Fecha_venta order by ingreso_total desc;`. The bottom pane shows the results of the query in a table grid. The table has four columns: an index, Fecha\_venta, cantidad\_ventas, and ingreso\_total. There are four rows of data. The interface includes a sidebar on the left with icons for Explorer, Text, and Grid, and a toolbar on the right with icons for various database operations.

	Fecha_venta	cantidad_ventas	ingreso_total
1	2024-04-03	6	199
2	2024-07-18	5	180
3	2024-01-29	5	166
4	2024-04-30	5	162

## Consulta de ingresos por autor

-- Esta consulta muestra los autores cuyos libros han generado más ingreso por ventas

```
select a.ID_autor, CONCAT_WS(' ', a.Nombre, a.Apellido1, a.Apellido2) as autor,
sum(dv.cantidad * dv.Precio_unitario) as ingresos_totales
from AUTOR a inner join LIBRO_EN_VENTA l
on a.ID_autor = l.id_autor
inner join DETALLE_VENTA dv
on l.ISBN = dv.ISBN
group by a.ID_autor, autor
order by ingresos_totales desc;
```

-- Esta consulta muestra los autores cuyos libros han generado más ingreso por ventas

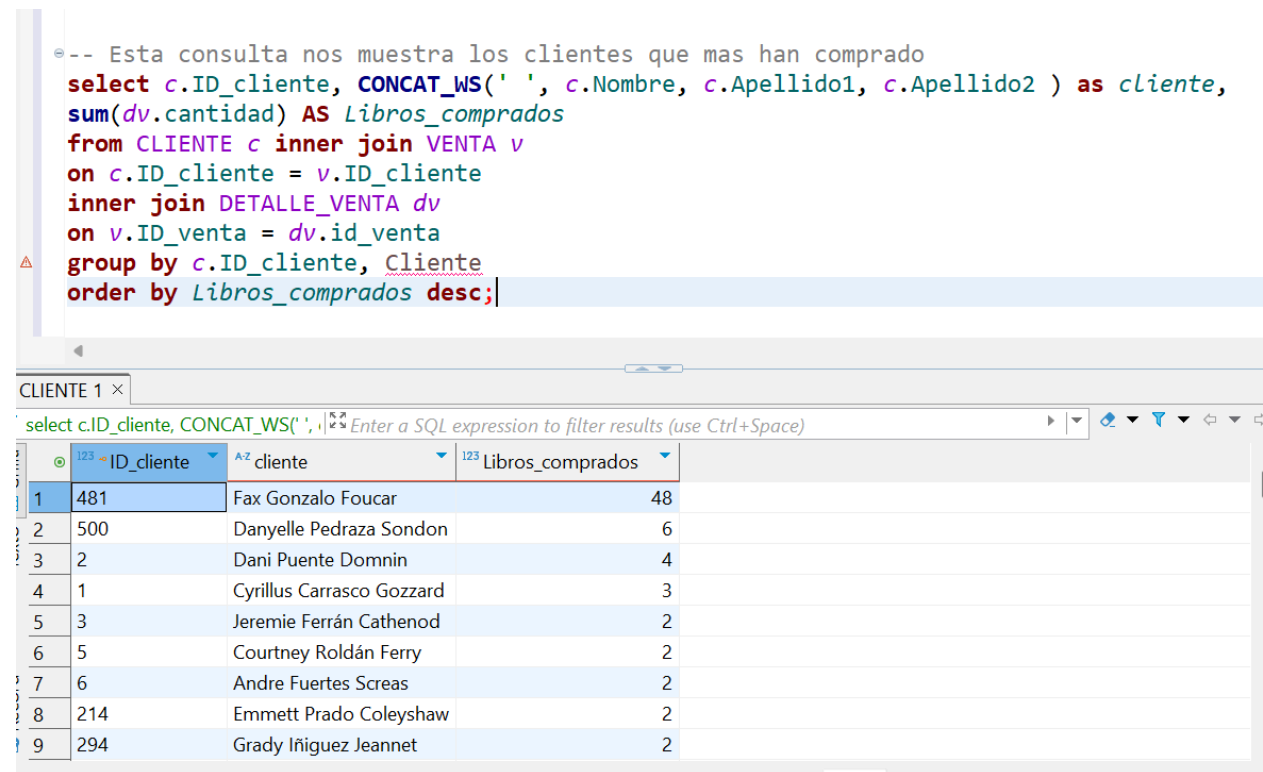
```
select a.ID_autor, CONCAT_WS(' ', a.Nombre, a.Apellido1, a.Apellido2) as autor,
sum(dv.cantidad * dv.Precio_unitario) as ingresos_totales
from AUTOR a inner join LIBRO_EN_VENTA l
on a.ID_autor = l.id_autor
inner join DETALLE_VENTA dv
on l.ISBN = dv.ISBN
group by a.ID_autor, autor
order by ingresos_totales desc;
```

	ID_autor	autor	ingresos_totales
1	344	Alessandro Pepis Izaguirre	1,029
2	420	Barbra Goudman Ferrández	180
3	431	Dareen Gyurko Noriega	156
4	479	Yehudi Shoulders Montoya	150
5	152	Welby Mattevi Falcón	144
6	459	Stefania Dodman Peralec	128

## Consulta de clientes con más compras

-- Esta consulta nos muestra los clientes que más han comprado

```
select c.ID_cliente, CONCAT_WS(' ', c.Nombre, c.Apellido1, c.Apellido2 ) as cliente,  
sum(dv.cantidad) AS Libros_comprados  
from CLIENTE c inner join VENTA v  
on c.ID_cliente = v.ID_cliente  
inner join DETALLE_VENTA dv  
on v.ID_venta = dv.id_venta  
group by c.ID_cliente, Cliente  
order by Libros_comprados desc;
```



The screenshot shows a SQL IDE with the query from the previous block. Below the query editor, a results window titled 'CLIENTE 1' displays the output of the query. The results are shown in a table with columns: ID\_cliente, cliente, and Libros\_comprados. The table is sorted by Libros\_comprados in descending order.

	ID_cliente	cliente	Libros_comprados
1	481	Fax Gonzalo Foucar	48
2	500	Danyelle Pedraza Sondon	6
3	2	Dani Puente Dominin	4
4	1	Cyrillus Carrasco Gozzard	3
5	3	Jeremie Ferrán Cathenod	2
6	5	Courtney Roldán Ferry	2
7	6	Andre Fuertes Screas	2
8	214	Emmett Prado Coleyshaw	2
9	294	Grady Iñiguez Jeannet	2



## Vistas

### Vista de ranking de libros vendidos

-- Esta vista nos muestra los libros que más se han vendido

**create view** libros\_mas\_vendidos **as**

**select** dv.ISBN, lev.Titulo, **sum**(dv.cantidad) **as** total\_vendido

**from** DETALLE\_VENTA dv **inner join** LIBRO\_EN\_VENTA lev

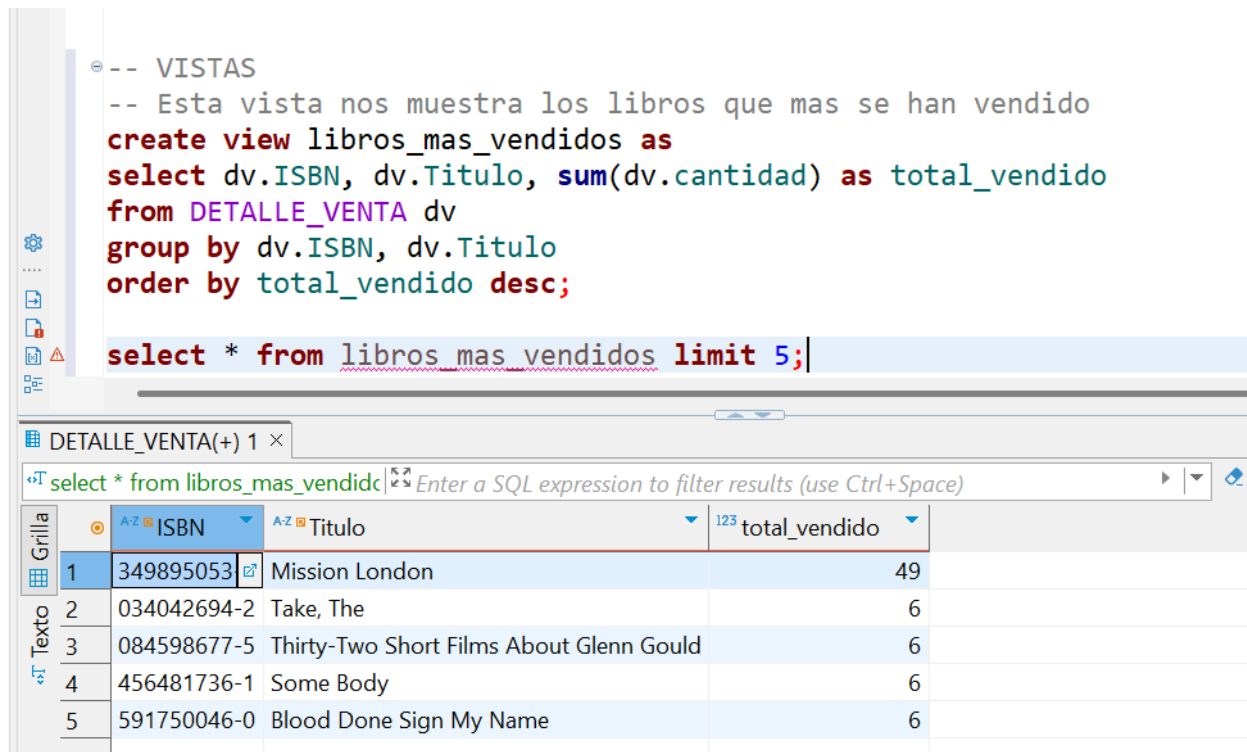
**on** dv.ISBN = lev.ISBN

**group by** dv.ISBN, lev.Titulo

**order by** total\_vendido **desc**;

-- Ejecutar

**select** \* **from** libros\_mas\_vendidos **limit** 5;



```
-- VISTAS
-- Esta vista nos muestra los libros que mas se han vendido
create view libros_mas_vendidos as
select dv.ISBN, dv.Titulo, sum(dv.cantidad) as total_vendido
from DETALLE_VENTA dv
group by dv.ISBN, dv.Titulo
order by total_vendido desc;

select * from libros_mas_vendidos limit 5;
```

DETALLE\_VENTA(+) 1 ×

select \* from libros\_mas\_vendidos Enter a SQL expression to filter results (use Ctrl+Space)

	ISBN	Titulo	total_vendido
1	349895053	Mission London	49
2	034042694-2	Take, The	6
3	084598677-5	Thirty-Two Short Films About Glenn Gould	6
4	456481736-1	Some Body	6
5	591750046-0	Blood Done Sign My Name	6

## Vista de autores con más ingresos

-- Esta vista muestra los autores cuyos libros han generado más ingreso por ventas

**create view** autores\_top\_ingresos **as**

**select** a.ID\_autor, **CONCAT\_WS**(' ', a.Nombre, a.Apellido1, a.Apellido2) **AS** autor,

**sum**(dv.cantidad \* dv.Precio\_unitario) **AS** ingresos\_totales

**from** AUTOR a **inner join** LIBRO\_EN\_VENTA l

**on** a.ID\_autor = l.id\_autor

**inner join** DETALLE\_VENTA dv

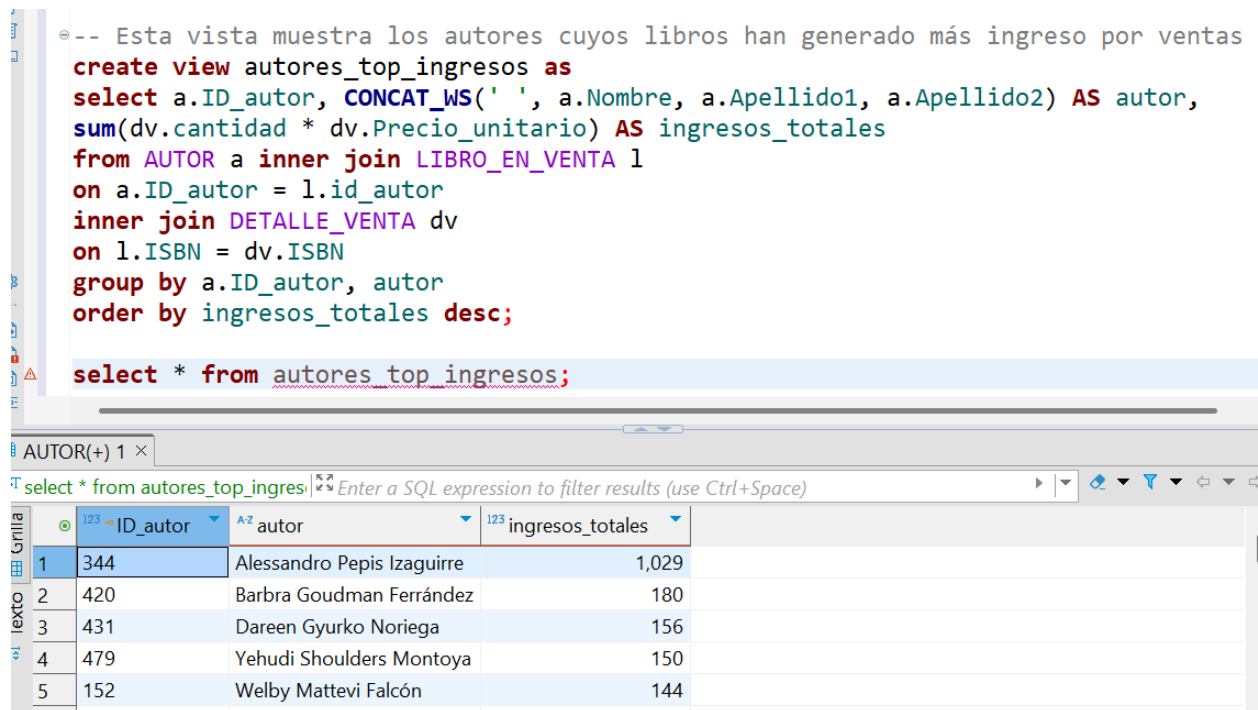
**on** l.ISBN = dv.ISBN

**group by** a.ID\_autor, autor

**order by** ingresos\_totales **desc**;

-- Ejecutar

**select \* from** autores\_top\_ingresos;



```
-- Esta vista muestra los autores cuyos libros han generado más ingreso por ventas
create view autores_top_ingresos as
select a.ID_autor, CONCAT_WS(' ', a.Nombre, a.Apellido1, a.Apellido2) AS autor,
sum(dv.cantidad * dv.Precio_unitario) AS ingresos_totales
from AUTOR a inner join LIBRO_EN_VENTA l
on a.ID_autor = l.id_autor
inner join DETALLE_VENTA dv
on l.ISBN = dv.ISBN
group by a.ID_autor, autor
order by ingresos_totales desc;

select * from autores_top_ingresos;
```

ID_autor	autor	ingresos_totales
344	Alessandro Pepis Izaguirre	1,029
420	Barbra Goudman Ferrández	180
431	Dareen Gyurko Noriega	156
479	Yehudi Shoulders Montoya	150
152	Welby Mattevi Falcón	144

## Funciones

### Función de cálculo de stock actual

-- Esta función calcula el stock de un libro

**delimiter //**

**create function** stock\_libro(isbn\_input **varchar**(20))

**returns int**

**deterministic**

**begin**

**declare** stock\_actual **int**;

**select** **l**.Stock **into** stock\_actual

**from** LIBRO\_EN\_VENTA **l**

**where** **l**.ISBN = isbn\_input;

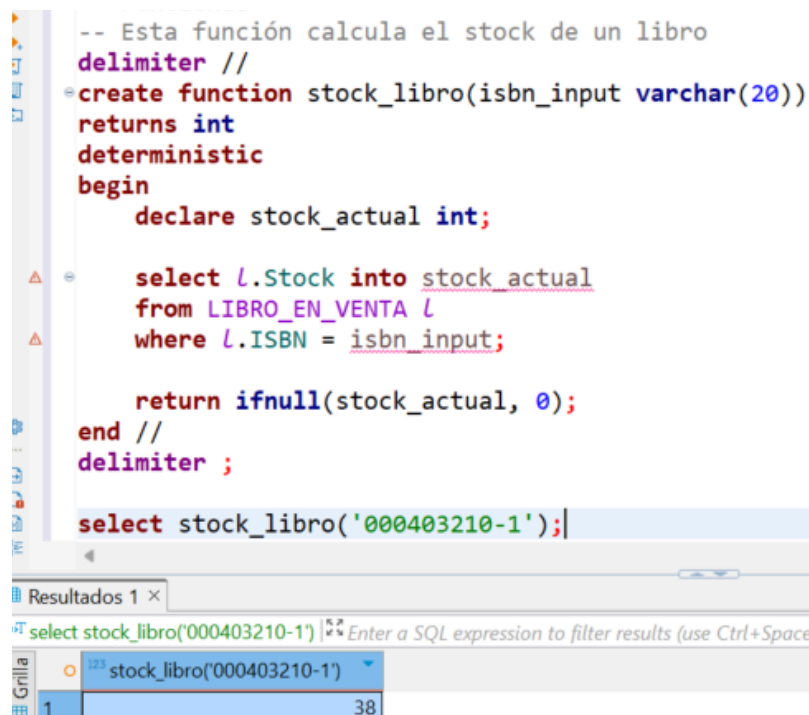
**return ifnull**(stock\_actual, 0);

**end //**

**delimiter ;**

-- Ejecutar

**select** stock\_libro('000403210-1');



The screenshot shows a SQL IDE with a code editor and a results pane. The code editor contains the following SQL code:

```
-- Esta función calcula el stock de un libro
delimiter //
create function stock_libro(isbn_input varchar(20))
returns int
deterministic
begin
    declare stock_actual int;
    select l.Stock into stock_actual
    from LIBRO_EN_VENTA l
    where l.ISBN = isbn_input;

    return ifnull(stock_actual, 0);
end //
delimiter ;

select stock_libro('000403210-1');
```

The results pane shows the output of the query:

Grilla	stock_libro('000403210-1')
1	38

## Función de obtención de nombre completo de cliente

-- Esta función es para obtener el nombre completo de un cliente

**delimiter //**

**create function** nombre\_completoCL(cliente\_id **int**)

**returns varchar**(255)

**deterministic**

**begin**

**declare** nombre\_completo **varchar**(255);

**select** **CONCAT\_WS**(' ', c.Nombre, c.Apellido1, c.Apellido2) **into**

nombre\_completo

**from** CLIENTE c

**where** c.ID\_cliente = cliente\_id;

**return** nombre\_completo;

**end //**

**delimiter ;**

-- Ejecutar


**select** nombre\_completoCL(2);

```
-- Esta funcion es para obtener el nombre completo de un cliente
delimiter //
create function nombre_completoCL(cliente_id int)
returns varchar(255)
deterministic
begin
    declare nombre_completo varchar(255);

    select CONCAT_WS(' ', c.Nombre, c.Apellido1, c.Apellido2) into nombre_completo
    from CLIENTE c
    where c.ID_cliente = cliente_id;

    return nombre_completo;
end //
delimiter ;

-- Ejecutar
select nombre_completoCL(2);
```



Resultados 1 x

select nombre\_completoCL(2) | Enter a SQL expression to filter results (use Ctrl+Space)

	nombre_completoCL(2)
1	Dani Puente Domin

## Procedimientos

### Procedimiento de inserción de libro

-- Este procedimiento sirve para añadir un nuevo libro

**delimiter //**

**create procedure** agregar\_libro (

**in** p\_isbn **varchar**(20),

**in** p\_titulo **varchar**(255),

**in** p\_id\_autor **int**,

**in** p\_editorial **varchar**(100),

**in** p\_stock **int**,

**in** p\_fecha **date**,

**in** p\_precio **decimal**(10,2)

)

**begin**

**insert into** LIBRO\_EN\_VENTA (ISBN, Titulo, id\_autor, Editorial, Stock,  
Fecha\_publicacion, Precio\_recomendado)

**values** (p\_isbn, p\_titulo, p\_id\_autor, p\_editorial, p\_stock, p\_fecha, p\_precio);

**end //**

**delimiter ;**

-- Ejecutar

-- Primero añadir el libro en guion\_libro

```
call agregar_libro('9788427053083', 'Un verano en el campamento', 2, 'Siruela', 50,
'2024-08-04', 20.00);
```

**select** \*

**from** LIBRO\_EN\_VENTA /

**where** /.Titulo = 'Un verano en el campamento';

```
-- Procedimientos
-- Este procedimiento sirve para añadir un nuevo libro
delimiter //
create procedure agregar_libro (
    in p_isbn varchar(20),
    in p_titulo varchar(255),
    in p_id_autor int,
    in p_editorial varchar(100),
    in p_stock int,
    in p_fecha date,
    in p_precio decimal(10,2)
)
begin
    insert into LIBRO_EN_VENTA (ISBN, Titulo, id_autor, Editorial, Stock, Fecha_publicacion, Pr
    values (p_isbn, p_titulo, p_id_autor, p_editorial, p_stock, p_fecha, p_precio);
end //
delimiter ;

-- Ejecutar
insert into GUION_LIBRO (Titulo, id_autor, sinopsis)
values ('Un verano en el campamento', 2, '"Un verano en el campamento" cuenta las aventuras de

call agregar_libro('9788427053083', 'Un verano en el campamento', 2, 'Siruela', 50, '2024-08-04

select *
from LIBRO_EN_VENTA L
where L.Titulo = 'Un verano en el campamento';
```

BRO\_EN\_VENTA 1 x

select \* from LIBRO\_EN\_VENTA | where | Enter a SQL expression to filter results (use Ctrl+Space)

	ISBN	Titulo	id_autor	Editorial	Stock	Fecha_publicacion	Precio_recomendado
1	9788427053083	Un verano en el campamento	2	Siruela	50	2024-08-04	20

## Procedimiento de actualización de stock

-- Este procedimiento es para actualizar el stock de un libro

**delimiter //**

**create procedure** actualizar\_stock (

**in** p\_isbn **varchar**(20),

**in** p\_nuevo\_stock **int**

)

**begin**

**update** LIBRO\_EN\_VENTA l

**set** Stock = p\_nuevo\_stock

**where** l.ISBN = p\_isbn;

**end //**

**delimiter ;**

-- Ejecutar

**call** actualizar\_stock('000080142-9', 70);

**select \***

**from** LIBRO\_EN\_VENTA l

**where** l.ISBN = '000080142-9';

The screenshot shows a SQL IDE interface with a script editor and a results pane. The script editor contains the following SQL code:

```
-- Este procedimiento es para actualizar el stock de un libro
delimiter //
create procedure actualizar_stock (
    in p_isbn varchar(20),
    in p_nuevo_stock int
)
begin
    update LIBRO_EN_VENTA l
    set Stock = p_nuevo_stock
    where l.ISBN = p_isbn;
end //
delimiter ;

-- Ejecutar
call actualizar_stock('000080142-9', 70);

select *
from LIBRO_EN_VENTA l
where l.ISBN = '000080142-9';
```

The results pane displays the output of the query, showing a table with the following data:

ISBN	Titulo	id_autor	Editorial	Stock	Fecha_publicacion	Precio_recomenda
000080142-9	Three on a Weekend	412	Ediciones Salamandra	70	2015-03-06	

Below the results table, there is a status bar showing the number of rows (200) and a button to export data. The interface also includes tabs for 'BRO\_EN\_VENTA 1' and 'ETALLE\_VENTA', and a menu for 'opiedades' with options for 'Datos' and 'Diagrama'.

## Procedimiento de consulta de detalles de venta

-- Este procedimiento es para mostrar todos los detalles de una venta por id

**delimiter //**

**create procedure** detalle\_venta (**in** p\_id\_venta **INT**)

**begin**

**select** dv.id\_venta , dv.ISBN , lev.Titulo , concat\_ws(' ', a.Nombre,  
a.Apellido1, a.Apellido2) **as** autor, dv.cantidad , dv.Precio\_unitario ,  
v.Precio\_total **as** subtotal

**from** DETALLE\_VENTA dv **inner join** LIBRO\_EN\_VENTA lev

**on** dv.ISBN = lev.ISBN

**inner join** AUTOR a

**on** lev.id\_autor = a.ID\_autor

**inner join** VENTA v

**on** dv.id\_venta = v.ID\_venta

**where** dv.id\_venta = p\_id\_venta;

**end //**

**delimiter ;**

-- Ejecutar

**call** detalle\_venta(**100**);

```

delimiter //
create procedure detalle_venta (in p_id_venta INT)
begin
select dv.id_venta , dv.ISBN , lev.Titulo , concat_ws(' ', a.Nombre,
a.Apellido1, a.Apellido2) as autor, dv.cantidad , dv.Precio_unitario ,
v.Precio_total as subtotal
from DETALLE_VENTA dv inner join LIBRO_EN_VENTA lev
on dv.ISBN = lev.ISBN
inner join AUTOR a
on lev.id_autor = a.ID_autor
inner join VENTA v
on dv.id_venta = v.ID_venta
where dv.id_venta = p_id_venta;
end //
delimiter ;

call detalle_venta(100);

```

LE\_VENTA(+) 1 × Estadísticas 1

detalle\_venta(100) | Enter a SQL expression to filter results (use Ctrl+Space)

id_venta	ISBN	Titulo	autor	cantidad	Precio_unitario	subtotal
100	467141847-9	Revenge of the Nerds III: The Next Gen	Damara Overall Saura	1	20.00	40
100	477453554-0	Butterflies Have No Memories	Deanne Edgworth Huerta	1	20.00	40



## Triggers

### Trigger de cálculo automático del subtotal

#### Inserción

**delimiter //**

**create trigger** actualizar\_precioTotal\_insert

**after insert on** DETALLE\_VENTA

**for each row**

**begin**

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

**select sum**(cantidad \* precio\_unitario) **into** total

**from** DETALLE\_VENTA

**where** id\_venta = new.id\_venta;

**update** VENTA

**set** Precio\_total = total

**where** ID\_venta = new.id\_venta;

**end //**

**delimiter ;**

-- Ejecutar

**insert into** DETALLE\_VENTA (id\_venta, ISBN, cantidad, Precio\_unitario)

**values** (1, '000080142-9', 1, 30.00);

The screenshot shows a database management interface. At the top, a SQL query is entered in a text area:

```
insert into DETALLE_VENTA (id_venta, ISBN, cantidad, Precio_unitario)
values (1, '000080142-9', 1, 30.00);
```

Below the query, a table titled "DETALLE\_VENTA 1" displays the results of the query. The table has four columns: id\_venta, ISBN, cantidad, and Precio\_unitario. The first row shows the values: 1, 000080142-9, 1, and 30.00.

Below the table, there are buttons for "Refresh", "Save", "Cancel", and "Exportar datos...". The "Exportar datos..." button is highlighted, and a dialog box is open with the value "200" entered.

At the bottom, there is a section for "LIBRO\_EN\_VENTA" with a "Configure" button. Below this, there is a section for "Propiedades" with tabs for "Datos" and "Diagrama ER". The "Datos" tab is selected, and it shows a table with the same columns as the one above, but with different values: 1, 014223511-3, 1, and 15.00.

## Actualización

-- Para cuando se actualice

**delimiter //**

**create trigger** actualizar\_precioTotal\_update

**after update on** DETALLE\_VENTA

**for each row**

**begin**

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

**select** **sum**(cantidad \* precio\_unitario) **into** total

**from** DETALLE\_VENTA

**where** id\_venta = new.id\_venta;

**update** VENTA

**set** Precio\_total = total

**where** ID\_venta = new.id\_venta;

**end //**

**delimiter ;**

-- Ejecutar

**update** DETALLE\_VENTA **set** cantidad = 2

**where** id\_venta = 1 **and** ISBN = '014223511-3';

The screenshot shows a database management interface. At the top, a SQL query is entered in a text area:

```
update DETALLE_VENTA
set cantidad = 2
where id_venta = 1 and ISBN = '014223511-3';
```

Below the query, a table titled "DETALLE\_VENTA 1" displays the data after the update. The table has four columns: id\_venta, ISBN, cantidad, and Precio\_unitario. The data is as follows:

	id_venta	ISBN	cantidad	Precio_unitario
1	1	000080142-9	1	30.00
2	1	014223511-3	2	15.00
3	1	495287228-9	1	15.00

Below the table, there are buttons for "Refresh", "Save", "Cancel", and navigation icons. A status bar indicates "3 row(s) fetched - 0,111s, on 2025-01-10 10:10:10".

At the bottom, there is a section for "LIBRO\_EN\_VENTA" with a tab for "DETALLE\_VENTA". The "Propiedades" tab is selected, showing a diagram of the database structure. The "Datos" tab is also visible, showing the data for the selected table.

## Trigger de validación de stock disponible

-- evitar que se venda más stock del disponible

**delimiter //**

**create trigger** verificar\_stock

**before insert on** DETALLE\_VENTA

**for each row**

**begin**

**declare** stock\_disponible **int**;

**select** Stock **into** stock\_disponible

**from** LIBRO\_EN\_VENTA

**where** ISBN = new.ISBN;

**if** stock\_disponible **is null then**

**signal sqlstate** '45000'

**set** message\_text = 'ISBN no encontrado en el inventario';

**elseif** new.cantidad > stock\_disponible **then**

**signal sqlstate** '45000'

**set** message\_text = 'No hay suficiente stock para completar la venta';

**end if**;

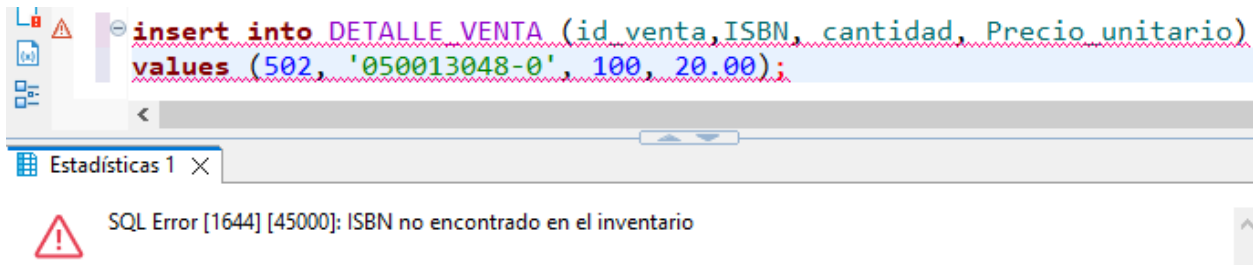
**end //**

**delimiter ;**

-- Ejecutar

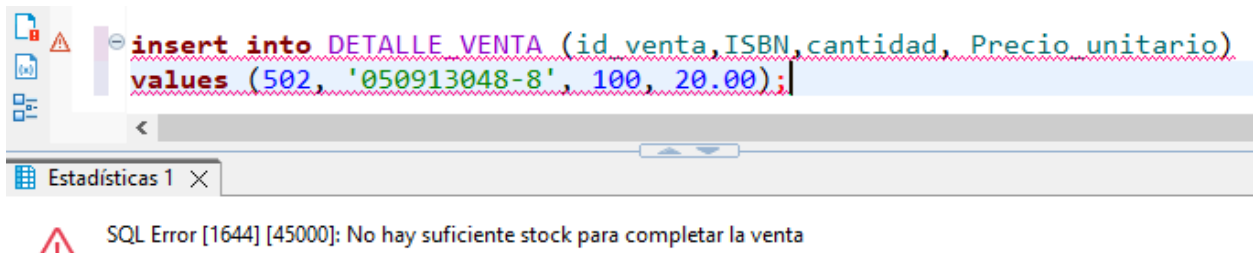
-- Opción 1

**insert into** DETALLE VENTA (id\_venta,ISBN,cantidad,Precio unitario)  
**values** (502,'050013048-0',100,20.00);



-- Opción 2

**insert into** DETALLE VENTA (id\_venta,ISBN,cantidad,Precio unitario)  
**values** (502,'050913048-8',100,20.00);



## Trigger de actualización automática del stock tras la venta

-- actualizar el stock automáticamente al vender

**delimiter //**

**create trigger** actualizar\_stock

**after insert on** DETALLE\_VENTA

**for each row**

**begin**

**update** LIBRO\_EN\_VENTA

**set** Stock = Stock - **new**.cantidad

**where** ISBN = **new**.ISBN;

**end //**

**delimiter ;**

-- Ejecutar

**insert into** DETALLE\_VENTA (id\_venta,ISBN,cantidad, Precio\_unitario)

**values** (500, '000403210-1',3, 20.00);

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

```
-- Ejecutar
insert into DETALLE_VENTA (id_venta,ISBN,cantidad, Precio_unitario)
values (500, '000403210-1',3, 20.00);

select *
from LIBRO_EN_VENTA lev
WHERE lev.ISBN = '000403210-1';
```

Below the editor, there's a table viewer for 'LIBRO\_EN\_VENTA 1'. It shows a single row with the following data:

	ISBN	Titulo	id_autor	Editorial	Stock
1	000403210-1	Ladrones	182	Siruela	35

Below the table viewer, there's a status bar indicating '1 row(s) fetched - 0,109s, on 2025-05-05 at 10:06:43'. At the bottom, there's another table viewer for '\*LIBRO\_EN\_VENTA'. It shows two rows of data:

	ISBN	Titulo	id_autor	Editorial	Stock
1	000080142-9	Three on a Weekend	412	Ediciones Salamandra	70
2	000403210-1	Ladrones	182	Siruela	38

## GitHub

Para facilitar el acceso y la organización del proyecto, he creado un repositorio en GitHub, donde he subido todo lo necesario relacionado con el desarrollo y documentación del mismo, el enlace al repositorio es:

[Repositorio Proyecto Bases de datos](#)

## AWS

Para el desarrollo y pruebas del proyecto, utilicé un servidor virtual de Amazon Web Services mediante el servicio EC2. En esta instancia instalé MySQL y configuré el acceso remoto para poder gestionar la base de datos desde DBeaver.

La dirección IP pública del servidor es: [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.