CODERHOUSE

MySQL

Comissionn 81860
Julian Daud

PROJECT:
E-COMMERCE MUSIC STORE

1) Project Description

The project develops the database for an e-commerce store focused on musical instruments. The system allows customer registration, product catalog management, purchase order generation, and payment tracking. The database organizes and centralizes all the information required for the proper operation of the website.

2) Problem Statement

The store uses separate files to manage data, which causes

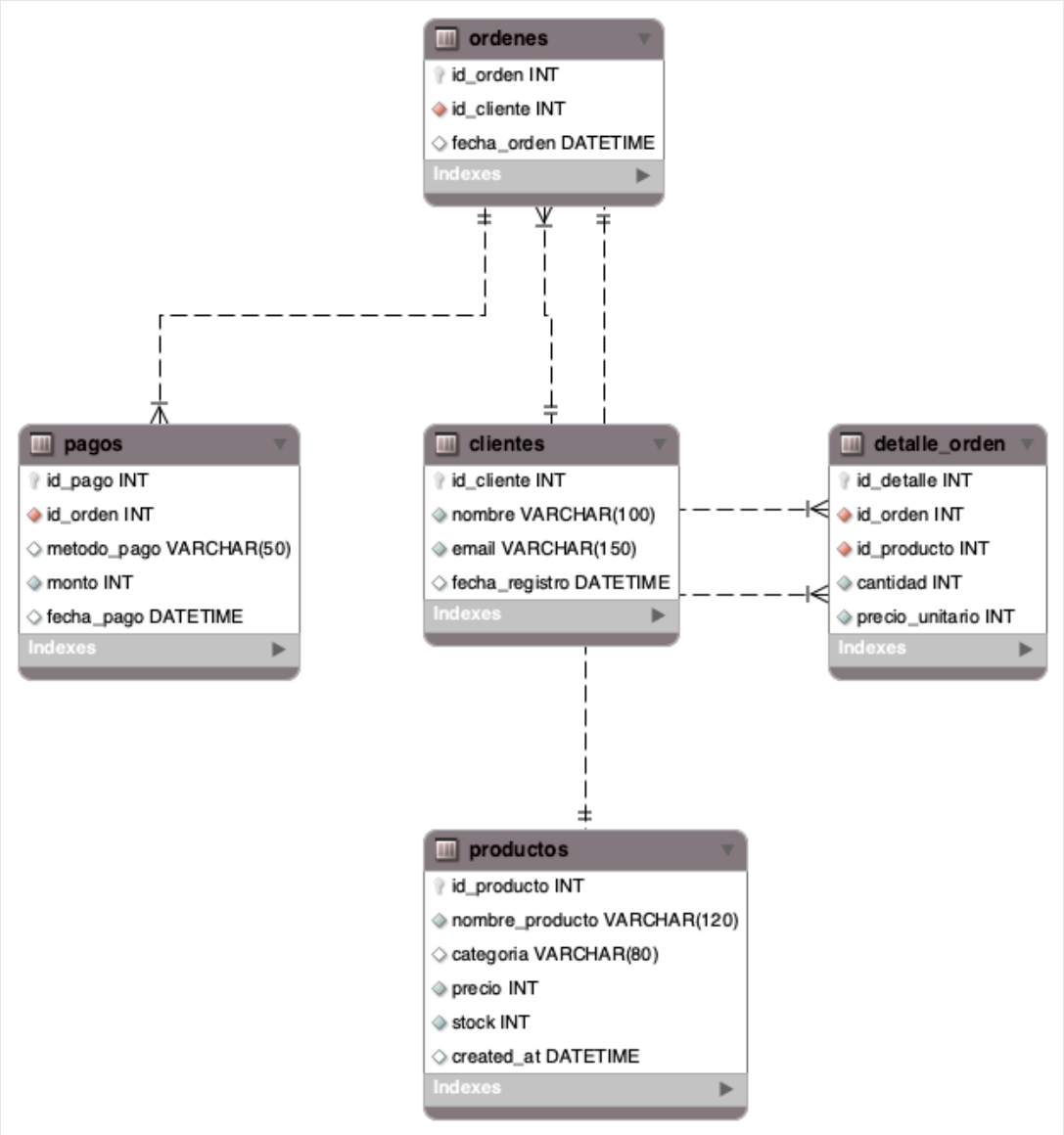
1. Difficulty tracking sales.
2. Stock errors and inconsistencies.
3. Duplicate or incomplete data.
4. Lack of a clear payment record.

A relational database is needed to unify and organize all the information efficiently.

3) Business Model

The company sells musical instruments through an e-commerce platform. It offers products with stock availability, registered customers, orders with multiple items, and a payment system. The business generates revenue through direct sales and requires a clear data structure to operate efficiently.

ER MODEL DIAGRAM



DATABASE TABLES

1)CUSTOMERS:

Stores the basic information of customers who register and make purchases on the e-commerce platform.

Nombre del campo	Abreviatura	Tipo de dato	Clave	Descripción
id_cliente	id_cliente	INT	PK	Identificador único del cliente
nombre	nombre	VARCHAR(100)	—	Nombre del cliente
email	email	VARCHAR(150)	—	Correo electrónico del cliente
fecha_registro	fecha_registro	DATETIME	—	Fecha en que el cliente se registró

2) PRODUCTS:

Contains the catalog of instruments and accessories available for sale.

Nombre del campo	Abreviatura	Tipo de dato	Clave	Descripción
id_producto	id_producto	INT	PK	Identificador único del
nombre_producto	nombre_producto	VARCHAR(120)	—	Nombre del producto o
categoria	categoria	VARCHAR(80)	—	Tipo/categoría del instrumento
precio	precio	INT	—	Precio del producto
stock	stock	INT	—	Cantidad disponible en
created_at	created_at	DATETIME	—	Fecha en que se agregó el

3) ORDERS:

Records each purchase made by the customers.

Nombre del campo	Abreviatura	Tipo de dato	Clave	Descripción
id_orden	id_orden	INT	PK	Identificador único de la orden
id_cliente	id_cliente	INT	FK	Cliente que realizó la compra
fecha_orden	fecha_orden	DATETIME	—	Fecha en que se generó la orden

4) ORDER_DETAILS:

Records the products included in each order.

Nombre del campo	Abreviatura	Tipo de dato	Clave	Descripción
id_detalle	id_detalle	INT	PK	Identificador único del detalle
id_orden	id_orden	INT	FK	Orden asociada
id_producto	id_producto	INT	FK	Producto incluido
cantidad	cantidad	INT	—	Cantidad comprada
precio_unitario	precio_unitario	INT	—	Precio del producto en el momento de la compra

5) PAYMENTS:

Records the payments made for each order.

Nombre del campo	Abreviatura	Tipo de dato	Clave	Descripción
id_pago	id_pago	INT	PK	Identificador único del pago
id_orden	id_orden	INT	FK	Orden asociada al pago
metodo_pago	metodo_pago	VARCHAR(50)	—	Método de pago utilizado
monto	monto	INT	—	Importe del pago
fecha_pago	fecha_pago	DATETIME	—	Fecha en que se realizó el pago

FIRST DELIVERY CORRECTIONS

Adjustments were made to the database:

```sql
#== CORRECIONES ==#



ALTER TABLE clientes
ADD UNIQUE (email),
ADD COLUMN ciudad VARCHAR (100),
ADD COLUMN pais VARCHAR (100);


ALTER TABLE ordenes
ADD COLUMN total DECIMAL(10,2);


ALTER TABLE productos
MODIFY precio DECIMAL (10, 2);

ALTER TABLE detalle_orden
MODIFY precio_unitario DECIMAL (10, 2);
```

- Customers: unique email and new columns for **city and country**.
- Orders: new column **total** for the order amount.
- Products / Order_Details: prices changed to **DECIMAL(10,2)** for greater precision.

These changes improve data integrity and enable the new functionalities of the project.

DATA LOADING INTO THE DATABASE

Shows the **INSERT** commands used to add data to the tables, along with screenshots of their execution in MySQL.

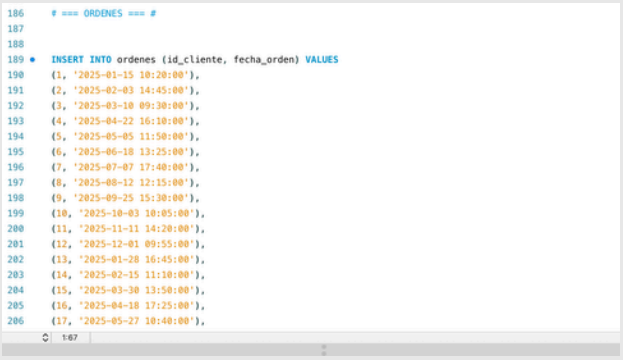
1) PAYMENTS

```
# === PAGOS === #

INSERT INTO pagos (id_orden, metodo_pago, monto, fecha_pago) VALUES
(1, 'Tarjeta de crédito', 68.00, '2025-01-15 11:00:00'),
(2, 'Venmo', 79.00, '2025-02-03 15:00:00'),
(3, 'Zelle', 72.00, '2025-03-10 10:00:00'),
(4, 'Tarjeta de crédito', 87.50, '2025-04-22 17:00:00'),
(5, 'Venmo', 71.00, '2025-05-05 12:30:00'),
(6, 'Zelle', 57.50, '2025-06-18 14:00:00'),
(7, 'Tarjeta de crédito', 82.50, '2025-07-07 18:30:00'),
(8, 'Venmo', 70.00, '2025-08-12 13:30:00'),
(9, 'Zelle', 73.00, '2025-09-25 16:00:00'),
(10, 'Tarjeta de crédito', 54.00, '2025-10-03 11:00:00');


#=== VISTAS ===#

#=1=#

CREATE VIEW vw_detalle_compra AS
SELECT  p.categoria,
```

4)CUSTOMERS

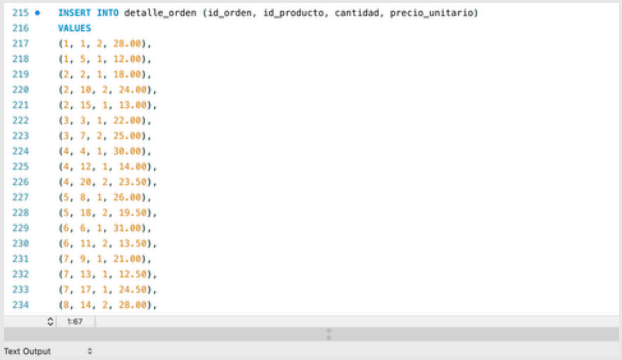
```
INSERT INTO clientes (nombre, email, fecha_registro, ciudad, pais)
VALUES
('Raul Salvatierra', 'raul.salvatierra.35@gmail.com', NOW(), 'Brooklyn', 'USA'),
        ('Maria Perez', 'maria@hotmail.com', NOW(), 'Buenos Aires', 'Argentina'),
        ('Carlos Lopez', 'carlos@gmail.com', NOW(), 'Madrid', 'España'),
        ('Gian Tenorio', 'gian.tenorio6@gmail.com', NOW(), 'Cali', 'Colombia'),
        ('Brian Scheller', 'bscheller@gmail.com', NOW(), 'Brooklyn', 'USA'),
        ('Juan Perez', 'juan.perez@gmail.com', NOW(), 'Buenos Aires', 'Argentina'),
        ('Maria Gonzalez', 'maria.gonzalez@yahoo.com', NOW(), 'Santiago', 'Chile'),
        ('Carlos Ramirez', 'carlos.ramirez@hotmail.com', NOW(), 'Lima', 'Peru'),
        ('Ana Fernandez', 'ana.fernandez@gmail.com', NOW(), 'Bogotá', 'Colombia'),
        ('Luis Martinez', 'luis.martinez@yahoo.com', NOW(), 'Quito', 'Ecuador'),
        ('Sofia Lopez', 'sofia.lopez@hotmail.com', NOW(), 'Montevideo', 'Uruguay'),
        ('Diego Sanchez', 'diego.sanchez@gmail.com', NOW(), 'Asunción', 'Paraguay'),
    ('Camila Torres', 'camila.torres@yahoo.com', NOW(), 'La Paz', 'Bolivia'),
        ('Jose Castillo', 'jose.castillo@hotmail.com', NOW(), 'Caracas', 'Venezuela'),
        ('Valentina Herrera', 'valentina.herrera@gmail.com', NOW(), 'Ciudad de Mexico', 'Mexico'),
        ('Ricardo Morales', 'ricardo.morales@yahoo.com', NOW(), 'Guadalajara', 'Mexico'),
        ('Fernanda Diaz', 'fernanda.diaz@hotmail.com', NOW(), 'Medellín', 'Colombia'),
        ('Gabriel Ortiz', 'gabriel.ortiz@gmail.com', NOW(), 'Santo Domingo', 'Republica Dominicana')
```

2)ORDERS



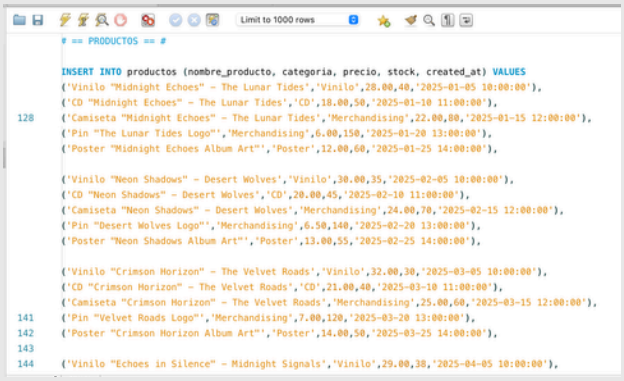
```
# === ORDENES === #


INSERT INTO ordenes (id_cliente, fecha_orden) VALUES
(1, '2025-01-15 10:20:00'),
(2, '2025-02-03 14:45:00'),
(3, '2025-03-10 09:30:00'),
(4, '2025-04-22 16:10:00'),
(5, '2025-05-05 11:50:00'),
(6, '2025-06-18 13:25:00'),
(7, '2025-07-07 17:40:00'),
(8, '2025-08-12 12:15:00'),
(9, '2025-09-25 15:30:00'),
(10, '2025-10-03 10:05:00'),
(11, '2025-11-11 14:20:00'),
(12, '2025-12-01 09:55:00'),
(13, '2025-01-28 16:45:00'),
(14, '2025-02-15 11:10:00'),
(15, '2025-03-30 13:50:00'),
(16, '2025-04-18 17:25:00'),
(17, '2025-05-27 10:40:00'),
```

5) ORDER DETAIL



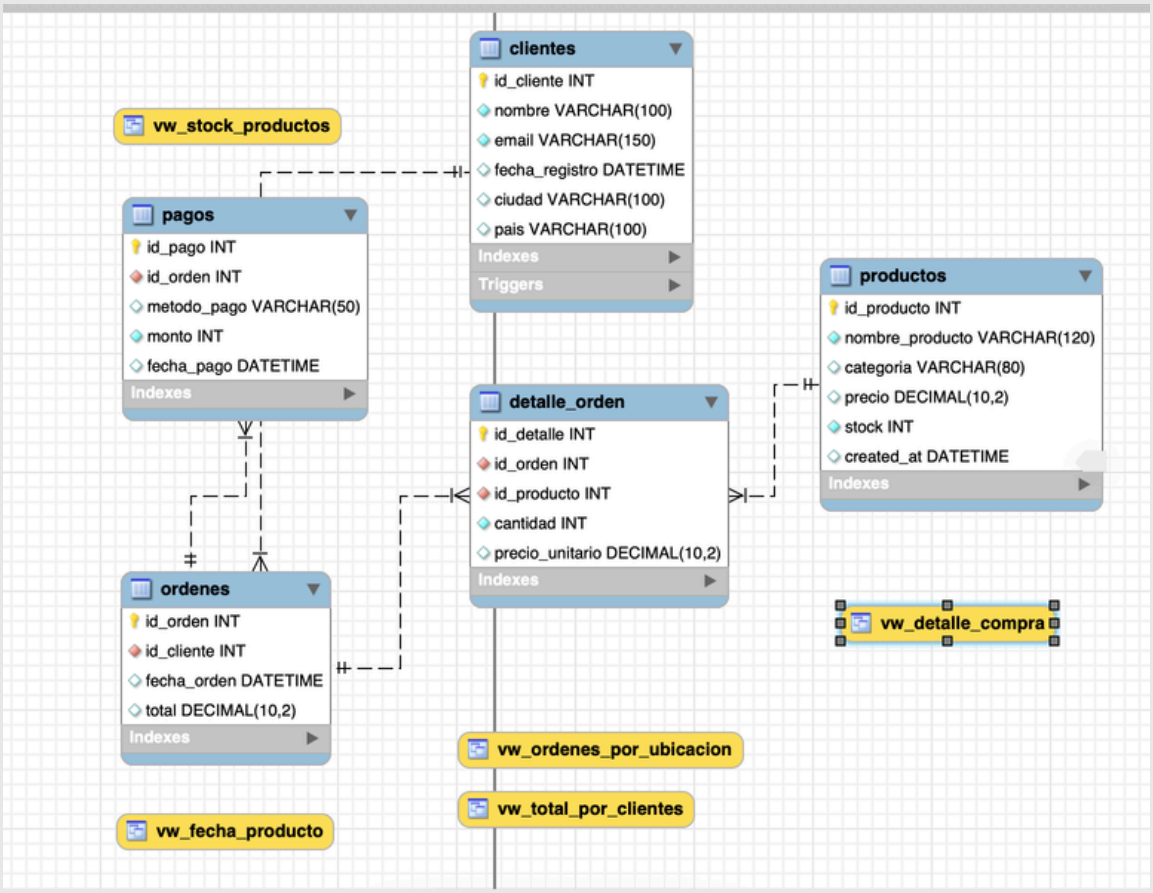
```
INSERT INTO detalle_orden (id_orden, id_producto, cantidad, precio_unitario)
VALUES
(1, 1, 2, 28.00),
(1, 5, 1, 12.00),
(2, 2, 1, 18.00),
(2, 10, 2, 24.00),
(2, 15, 1, 13.00),
(3, 3, 1, 22.00),
(3, 7, 2, 25.00),
(4, 4, 1, 30.00),
(4, 12, 1, 14.00),
(4, 20, 2, 23.50),
(5, 8, 1, 26.00),
(5, 18, 2, 19.50),
(6, 6, 1, 31.00),
(6, 11, 2, 13.50),
(7, 9, 1, 21.00),
(7, 13, 1, 12.50),
(7, 17, 1, 24.50),
(8, 14, 2, 28.00),
```

3) PRODUCTS



```
# == PRODUCTOS == #

INSERT INTO productos (nombre_producto, categoria, precio, stock, created_at) VALUES
('Vinilo "Midnight Echoes" - The Lunar Tides','Vinilo',28.00,40,'2025-01-05 10:00:00'),
('CD "Midnight Echoes" - The Lunar Tides','CD',18.00,50,'2025-01-10 11:00:00'),
('Camiseta "Midnight Echoes" - The Lunar Tides','Merchandising',22.00,80,'2025-01-15 12:00:00'),
('Pin "The Lunar Tides Logo"','Merchandising',6.00,150,'2025-01-20 13:00:00'),
('Poster "Midnight Echoes Album Art"','Poster',12.00,60,'2025-01-25 14:00:00'),

('Vinilo "Neon Shadows" - Desert Wolves','Vinilo',30.00,35,'2025-02-05 10:00:00'),
('CD "Neon Shadows" - Desert Wolves','CD',20.00,45,'2025-02-10 11:00:00'),
('Camiseta "Neon Shadows" - Desert Wolves','Merchandising',24.00,70,'2025-02-15 12:00:00'),
('Pin "Desert Wolves Logo"','Merchandising',6.50,140,'2025-02-20 13:00:00'),
('Poster "Neon Shadows Album Art"','Poster',13.00,55,'2025-02-25 14:00:00'),

('Vinilo "Crimson Horizon" - The Velvet Roads','Vinilo',32.00,30,'2025-03-05 10:00:00'),
('CD "Crimson Horizon" - The Velvet Roads','CD',21.00,40,'2025-03-10 11:00:00'),
('Camiseta "Crimson Horizon" - The Velvet Roads','Merchandising',25.00,60,'2025-03-15 12:00:00'),
('Pin "Velvet Roads Logo"','Merchandising',7.00,120,'2025-03-20 13:00:00'),
('Poster "Crimson Horizon Album Art"','Poster',14.00,50,'2025-03-25 14:00:00'),

('Vinilo "Echoes in Silence" - Midnight Signals','Vinilo',29.00,38,'2025-04-05 10:00:00'),
```

UPDATED ER MODEL

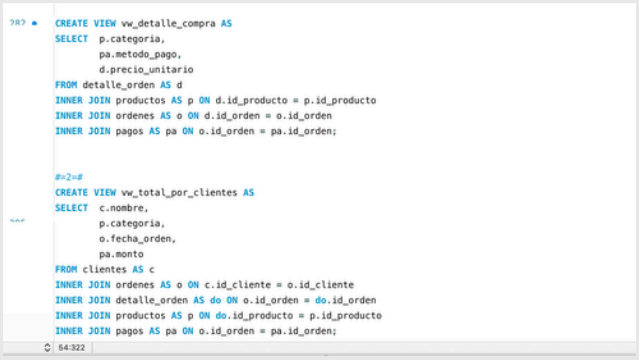


The original tables were adjusted by adding location and total columns, and data types were corrected to better reflect prices and amounts, while maintaining table relationships and optimizing database integrity.

VIEWS

The results of the created views are presented, showing that they **return the expected information correctly.**

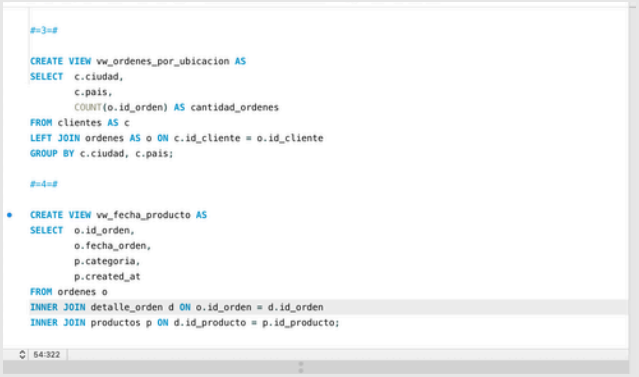
VIEWS 1 & 2



```sql
CREATE VIEW vw_detalle_compra AS
SELECT  p.categoria,
        pa.metodo_pago,
        d.precio_unitario
FROM detalle_orden AS d
INNER JOIN productos AS p ON d.id_producto = p.id_producto
INNER JOIN ordenes AS o ON d.id_orden = o.id_orden
INNER JOIN pagos AS pa ON o.id_orden = pa.id_orden;


#=2=#
CREATE VIEW vw_total_por_clientes AS
SELECT  c.nombre,
        p.categoria,
        o.fecha_orden,
        pa.monto
FROM clientes AS c
INNER JOIN ordenes AS o ON c.id_cliente = o.id_cliente
INNER JOIN detalle_orden AS do ON o.id_orden = do.id_orden
INNER JOIN productos AS p ON do.id_producto = p.id_producto
INNER JOIN pagos AS pa ON o.id_orden = pa.id_orden;
```

1. **VW_DETALLE_COMPRA:** Shows the **product category, payment method, and unit price** for each order detail. Tables: **Order_Details, Products, Orders, Payments**
2. **VW_TOTAL_POR_CLIENTES**: Shows the customer's name, product category, order date, and amount paid for each purchase. **Tables: Customers, Orders, Order_Details, Products, Payments**

VIEWS 3 & 4



```sql
#=3=#

CREATE VIEW vw_ordenes_por_ubicacion AS
SELECT  c.ciudad,
        c.pais,
        COUNT(o.id_orden) AS cantidad_ordenes
FROM clientes AS c
LEFT JOIN ordenes AS o ON c.id_cliente = o.id_cliente
GROUP BY c.ciudad, c.pais;

#=4=#

CREATE VIEW vw_fecha_producto AS
SELECT  o.id_orden,
        o.fecha_orden,
        p.categoria,
        p.created_at
FROM ordenes o
INNER JOIN detalle_orden d ON o.id_orden = d.id_orden
INNER JOIN productos p ON d.id_producto = p.id_producto;
```

1. **VW_ORDERS_BY_LOCATION:** Groups orders by **city and country,** showing the number of orders per location. Tables: **Customers, Orders**
2. **VW_DATE_PRODUCT:** Shows the date of each order along with the product category and the product creation date. Tables: **Orders, Order_Details, Products**

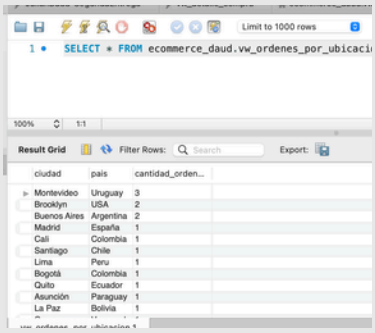
VIEWS 5

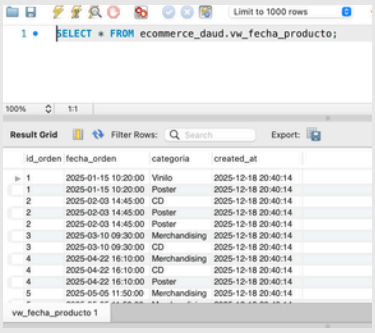
```sql
#=5=#

CREATE VIEW vw_stock_productos AS
SELECT  p.nombre_producto,
        p.stock,
        d.precio_unitario
FROM productos p
JOIN detalle_orden d ON p.id_producto = d.id_producto;
```

1. **VW_STOCK_PRODUCTOS**: Shows the available stock of each product in the inventory. Tables: **Products**

TEST QUERIES

```sql
CREATE VIEW vw_stock_productos AS
SELECT  p.nombre_producto,
        p.stock,
        d.precio_unitario
FROM productos p
JOIN detalle_orden d ON p.id_producto = d.id_producto;


# EJECUSIONES #

SELECT * FROM vw_detalle_compra
WHERE metodo_pago IN ('Venmo');

SELECT * FROM vw_total_por_clientes
WHERE categoria IN ('Vinilo');

SELECT * FROM vw_ordenes_por_ubicacion
WHERE pais IN ('Argentina');
```

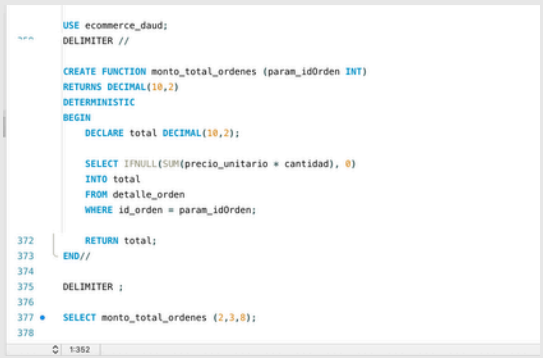




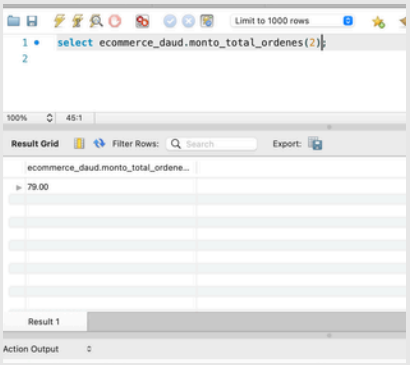
Test queries are shown on the **views**, **functions,** and **stored procedures**, verifying that they return the expected data according to the project logic.

FUNCTIONS

TOTAL ORDER AMOUNT



```
USE ecommerce_daud;
DELIMITER //

CREATE FUNCTION monto_total_ordenes (param_idOrden INT)
RETURNS DECIMAL(10,2)
DETERMINISTIC
BEGIN
    DECLARE total DECIMAL(10,2);

    SELECT IFNULL(SUM(precio_unitario * cantidad), 0)
    INTO total
    FROM detalle_orden
    WHERE id_orden = param_idOrden;

    RETURN total;
END//

DELIMITER ;

SELECT monto_total_ordenes (2,3,8);
```



```
select ecommerce_daud.monto_total_ordenes(2);
```

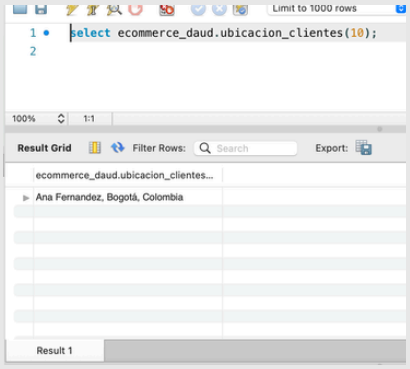
Result Grid

ecommerce_daud.monto_total_ordene...
79.00

Calculates the total amount of an order by multiplying the unit price by the quantity of each product in Order_Details.
Tables: **Order_Details**
Purpose: Allows quickly obtaining the total for a specific order, avoiding manual summation of the products.

CUSTOMER LOCATION

```
DELIMITER //

CREATE FUNCTION ubicacion_clientes(param_idCliente INT)
RETURNS VARCHAR(250)
DETERMINISTIC
BEGIN
    DECLARE info VARCHAR(250);
    SELECT CONCAT(nombre, ', ', ciudad, ', ', pais) INTO info
    FROM clientes
    WHERE id_cliente = param_idCliente;

    RETURN info;
END//

DELIMITER ;

SELECT ubicacion_clientes (2);

#=== SP ===#

# = 1 = #
```



```
select ecommerce_daud.ubicacion_clientes(10);
```

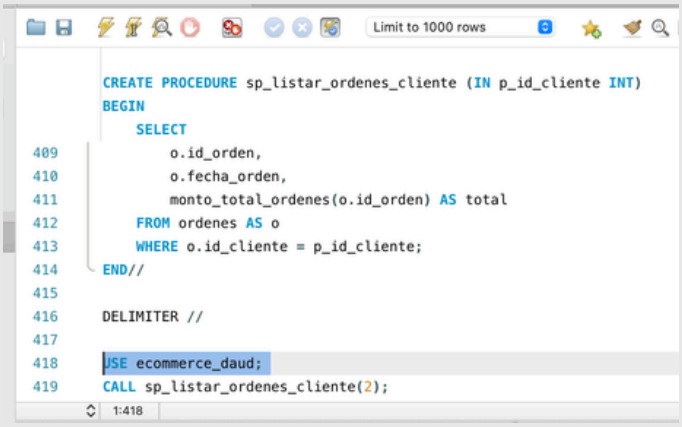
Result Grid

ecommerce_daud.ubicacion_clientes...
Ana Fernandez, Bogotá, Colombia

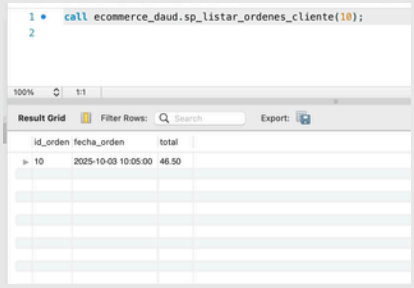
Returns the **customer's full name** along with their **city and country** as a single string.
Tables: **Customers**
Purpose: Makes it easy to display complete customer information in reports or views without multiple joins or concatenations.

STORED PROCEDURES

LIST CUSTOMER ORDERS



```
CREATE PROCEDURE sp_listar_ordenes_cliente (IN p_id_cliente INT)
BEGIN
    SELECT
        o.id_orden,
        o.fecha_orden,
        monto_total_ordenes(o.id_orden) AS total
    FROM ordenes AS o
    WHERE o.id_cliente = p_id_cliente;
END//

DELIMITER //

USE ecommerce_daud;
CALL sp_listar_ordenes_cliente(2);
```



```
call ecommerce_daud.sp_listar_ordenes_cliente(10);
```

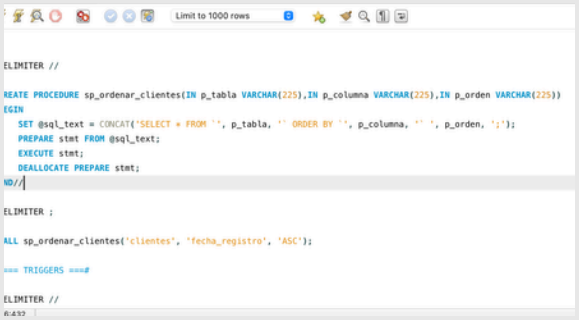
id_orden	fecha_orden	total
10	2025-10-03 10:05:00	46.50

Displays all orders for a specific customer along with the date and total of each order.

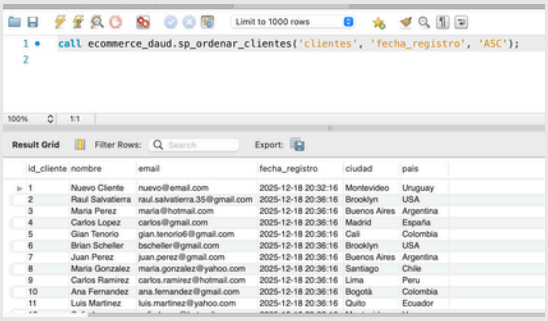
Tables: **Orders, Order_Details** (through the total_order_amount function)

Purpose: Allows quickly querying a customer's purchase history without complex joins.

SORT CUSTOMERS



```
ELIMITER //

REATE PROCEDURE sp_ordenar_clientes(IN p_tabla VARCHAR(225),IN p_columna VARCHAR(225),IN p_orden VARCHAR(225))
EGIN
   SET @sql_text = CONCAT('SELECT * FROM ', p_tabla, ' ORDER BY ', p_columna, ' ', p_orden, ';');
   PREPARE stmt FROM @sql_text;
   EXECUTE stmt;
   DEALLOCATE PREPARE stmt;
ND//

ELIMITER ;

ALL sp_ordenar_clientes('clientes', 'fecha_registro', 'ASC');

=== TRIGGERS ===#

ELIMITER //
```



```
call ecommerce_daud.sp_ordenar_clientes('clientes', 'fecha_registro', 'ASC');
```

id_cliente	nombre	email	fecha_registro	ciudad	pais
1	Nuevo Cliente	nuevo@email.com	2025-12-18 20:32:16	Montevideo	Uruguay
2	Raul Salvatierra	raul.salvatierra.35@gmail.com	2025-12-18 20:36:16	Brooklyn	USA
3	Maria Perez	maria@hotmail.com	2025-12-18 20:36:16	Buenos Aires	Argentina
4	Carlos Lopez	carlos@gmail.com	2025-12-18 20:36:16	Madrid	España
5	Gian Tenorio	gian.tenorio6@gmail.com	2025-12-18 20:36:16	Cali	Colombia
6	Brian Scheller	bscheller@gmail.com	2025-12-18 20:36:16	Brooklyn	USA
7	Juan Perez	juan.perez@gmail.com	2025-12-18 20:36:16	Buenos Aires	Argentina
8	Maria Gonzalez	maria.gonzalez@yahoo.com	2025-12-18 20:36:16	Santiago	Chile
9	Carlos Ramirez	carlos.ramirez@hotmail.com	2025-12-18 20:36:16	Lima	Peru
10	Ana Fernandez	ana.fernandez@gmail.com	2025-12-18 20:36:16	Bogotá	Colombia
11	Luis Martinez	luis.martinez@yahoo.com	2025-12-18 20:36:16	Quito	Ecuador

Dynamically sorts any table by the specified **column and order.**

Tables: Can be applied to any table passed as a parameter.

Purpose: Allows listing data in an organized way without creating multiple fixed queries, enabling flexible and reusable queries.

TRIGGERS

UPDATE STOCK

```
440  DELIMITER //
441
442  CREATE TRIGGER trg_actualizar_stock
443  AFTER INSERT ON detalle_orden
444  FOR EACH ROW
445  BEGIN
446      UPDATE productos
447      SET stock = stock - NEW.cantidad
448      WHERE id_producto = NEW.id_producto;
449  END//
450
451  DELIMITER ;
452
453  SELECT stock FROM productos WHERE id_producto = 15;
454
455
456  DELIMITER //
```

37:447

Automatically updates a product's stock whenever a **new order** detail is inserted.

Tables: Order_Details, Products

Purpose: Keeps the inventory up-to-date without manual calculations after each sale.

MARK PAYMENT

```
DELIMITER //

CREATE TRIGGER trg_marcar_pago
AFTER UPDATE ON pagos
FOR EACH ROW
BEGIN
    IF NEW.estado_pago = 'Pagado' THEN
        UPDATE ordenes
        SET total = NEW.monto
        WHERE id_orden = NEW.id_orden;
    END IF;
END//

DELIMITER ;

SELECT * FROM pagos WHERE id_pago = 70;
SELECT * FROM ordenes WHERE id_orden = (SELECT id_orden FROM pagos WHERE id_pago = 70);
```

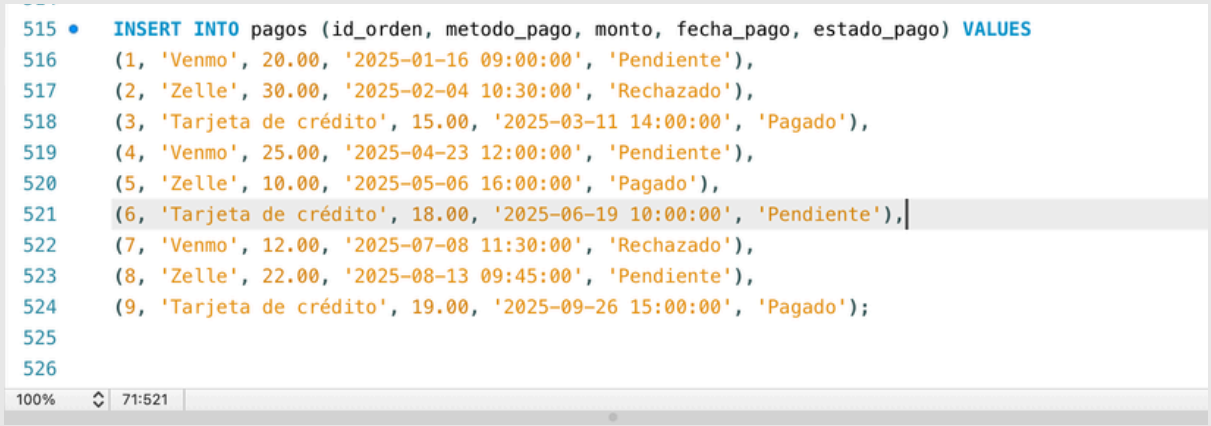
Trigger that executes when a record in the **Payments** table is updated. When a payment status changes to "Paid," the corresponding total in the **Orders** table is updated, ensuring consistency between payments and orders.

SECOND DELIVERY CORRECTIONS

Adjustments were made to the database:

```sql
ALTER TABLE pagos
MODIFY COLUMN estado_pago ENUM('Pagado','Pendiente','Rechazado');


UPDATE pagos
SET estado_pago = 'Pagado'
WHERE id_pago >= 1
  AND fecha_pago IS NOT NULL;


UPDATE pagos
SET estado_pago = 'Pendiente'
WHERE id_pago IN (3, 7, 15);

UPDATE pagos
SET estado_pago = 'Rechazado'
WHERE id_pago IN (9, 21);
```

- The payment_status column in the Payments table was changed to **ENUM** type **(Paid, Pending, Rejected)**.
- Records were updated to reflect the actual **status** of each order.
- Adjustments were made to simplify payment tracking.



```sql
INSERT INTO pagos (id_orden, metodo_pago, monto, fecha_pago, estado_pago) VALUES
(1, 'Venmo', 20.00, '2025-01-16 09:00:00', 'Pendiente'),
(2, 'Zelle', 30.00, '2025-02-04 10:30:00', 'Rechazado'),
(3, 'Tarjeta de crédito', 15.00, '2025-03-11 14:00:00', 'Pagado'),
(4, 'Venmo', 25.00, '2025-04-23 12:00:00', 'Pendiente'),
(5, 'Zelle', 10.00, '2025-05-06 16:00:00', 'Pagado'),
(6, 'Tarjeta de crédito', 18.00, '2025-06-19 10:00:00', 'Pendiente'),
(7, 'Venmo', 12.00, '2025-07-08 11:30:00', 'Rechazado'),
(8, 'Zelle', 22.00, '2025-08-13 09:45:00', 'Pendiente'),
(9, 'Tarjeta de crédito', 19.00, '2025-09-26 15:00:00', 'Pagado');
```

Additional payments were added using different **order_id and payment_id** values to reflect that an order can have multiple payments with varying statuses **(Paid, Pending, Rejected)**

```sql
ALTER TABLE pagos
MODIFY monto DECIMAL(10,2);
```

The amount field in the payments table was changed to **DECIMAL(10,2)** to correctly represent monetary values, allow **decimals**, and keep consistency in the data model.

```sql
CREATE OR REPLACE VIEW vw_stock_productos AS
SELECT
    nombre_producto,
    stock
FROM productos;
```

The **vw_stock_productos** view was replaced using **CREATE OR REPLACE VIEW** so that it selects only **nombre_producto** and stock from the productos table, removing the join with **detalle_orden** and the precio_unitariocolumn.

```sql
# === DATOS === #

USE ecommerce_daud;

#=== CLIENTES ===#

USE ecommerce_daud;
```

The **USE** command was corrected to point to the correct database **ecommerce_daud** before operating on the clientestable.

The call to the **monto_total_ordenes** function was modified to execute one **id_orden** at a time, respecting the function definition that accepts a single parameter.

```sql
    RETURN total;
END//

DELIMITER ;

SELECT monto_total_ordenes(2);
SELECT monto_total_ordenes(3);
SELECT monto_total_ordenes(8);
```

ACTUALIZACION DE TRIGGER

The **agregar_clientes** trigger was removed because it required creating a new table that was not part of the original design and could alter the database dynamics. Instead, a new trigger **focused on payment and order management** was implemented, maintaining simpler logic and alignment with the defined data model.

DATA UPDATE AND INSERT

New records were added to achieve a **more balanced and realistic data distribution**, allowing better testing of the model's functionality. The final dataset includes **70 customers, 115 orders, 131 order detail records, 200 products, and 115 payments.**

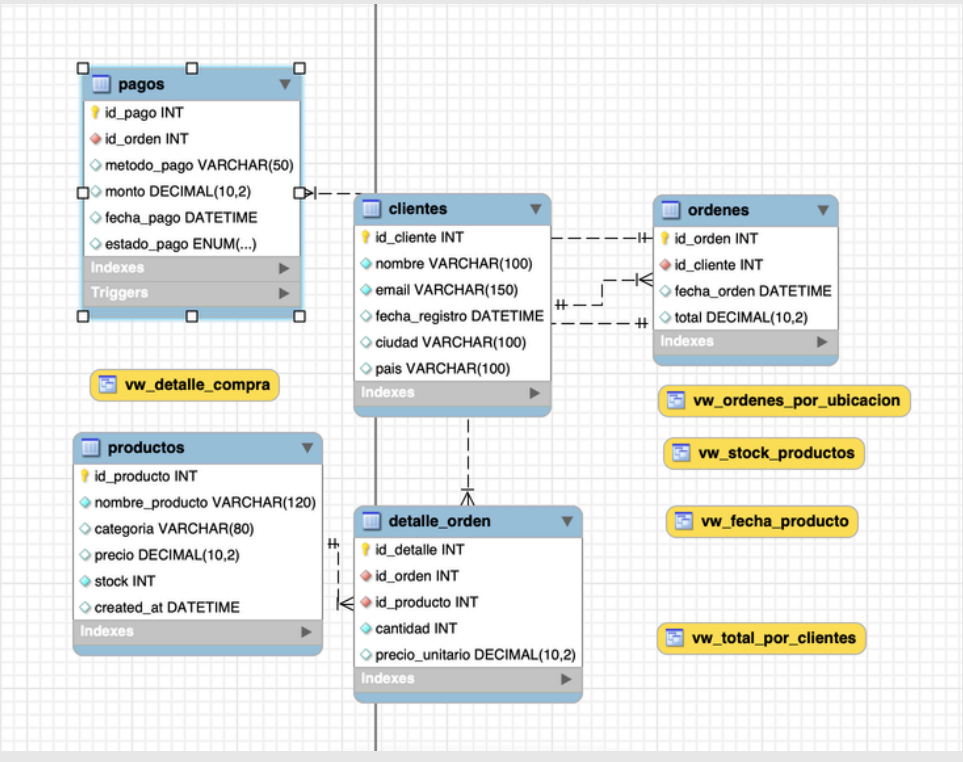
TOTAL ORDERS

Result Grid | Filter Rows: Search | Edit: | Export/Import:

id_orden	id_cliente	fecha_orden	total
1	5	2025-01-01 10:00:00	21.00
2	12	2025-01-02 11:15:00	12.00
3	7	2025-01-03 12:30:00	24.00
4	20	2025-01-04 13:45:00	20.00
5	3	2025-01-05 14:00:00	7.00
6	42	2025-01-06 15:10:00	23.00
7	18	2025-01-07 16:20:00	14.50
8	31	2025-01-08 17:30:00	13.50
9	50	2025-01-09 09:45:00	32.00
10	25	2025-01-10 10:50:00	22.00
11	8	2025-01-11 11:05:00	14.00
12	3	2025-01-12 12:15:00	22.00
13	17	2025-01-13 13:25:00	6.50
14	9	2025-01-14 14:35:00	6.50

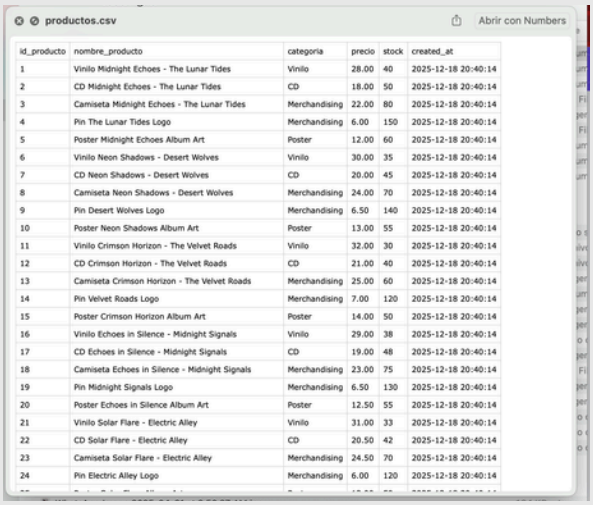
The total field in the ordenes table was updated based on each product's price and the quantity recorded in the **detalle_orden** table, allowing the correct t**otal amount for each order** to be reflected.

FINAL ER MODEL

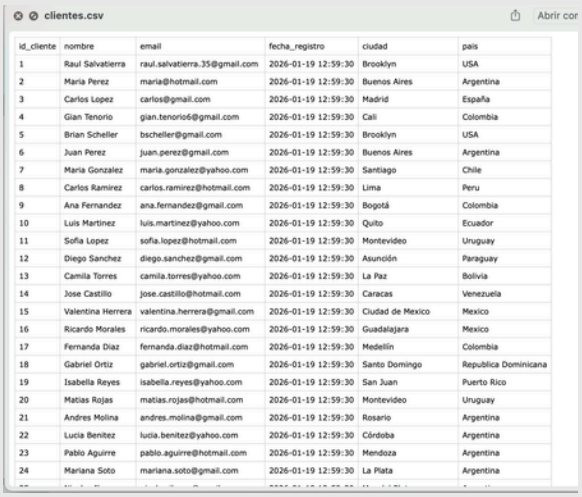


SUPPLEMENTARY MATERIAL

The tables **were exported** from **MySQL Workbench in CSV format**, then opened and organized in **Google Sheets** to build a working dataset. The dataset was later used in **Power BI** to generate visualization charts.

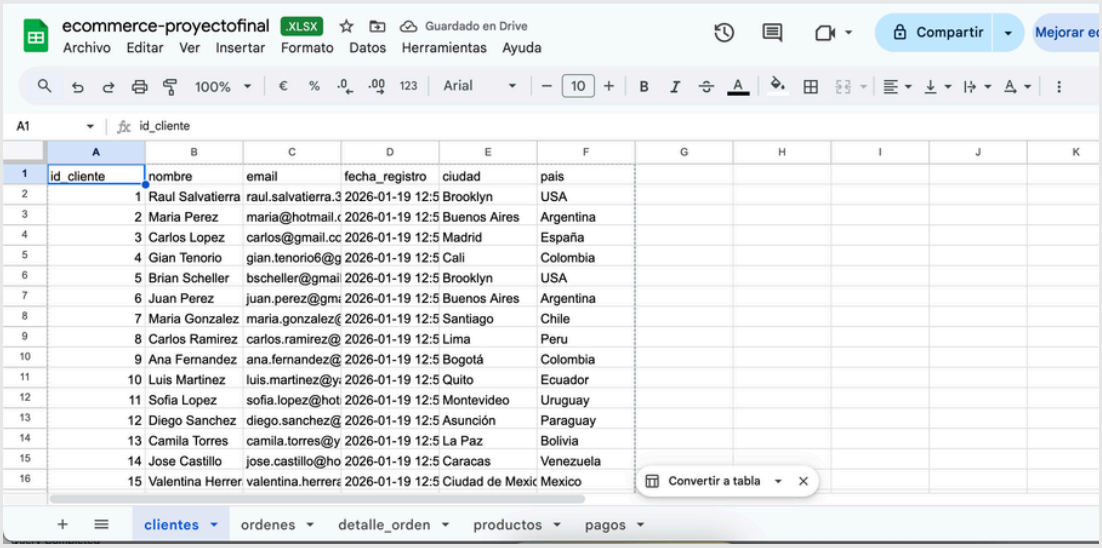

productos.csv — Abrir con Numbers

id_producto	nombre_producto	categoria	precio	stock	created_at
1	Vinilo Midnight Echoes - The Lunar Tides	Vinilo	28.00	40	2025-12-18 20:40:14
2	CD Midnight Echoes - The Lunar Tides	CD	18.00	50	2025-12-18 20:40:14
3	Camiseta Midnight Echoes - The Lunar Tides	Merchandising	22.00	80	2025-12-18 20:40:14
4	Pin The Lunar Tides Logo	Merchandising	6.00	150	2025-12-18 20:40:14
5	Poster Midnight Echoes Album Art	Poster	12.00	60	2025-12-18 20:40:14
6	Vinilo Neon Shadows - Desert Wolves	Vinilo	30.00	35	2025-12-18 20:40:14
7	CD Neon Shadows - Desert Wolves	CD	20.00	45	2025-12-18 20:40:14
8	Camiseta Neon Shadows - Desert Wolves	Merchandising	24.00	70	2025-12-18 20:40:14
9	Pin Desert Wolves Logo	Merchandising	6.50	140	2025-12-18 20:40:14
10	Poster Neon Shadows Album Art	Poster	13.00	55	2025-12-18 20:40:14
11	Vinilo Crimson Horizon - The Velvet Roads	Vinilo	32.00	30	2025-12-18 20:40:14
12	CD Crimson Horizon - The Velvet Roads	CD	21.00	40	2025-12-18 20:40:14
13	Camiseta Crimson Horizon - The Velvet Roads	Merchandising	25.00	60	2025-12-18 20:40:14
14	Pin Velvet Roads Logo	Merchandising	7.00	120	2025-12-18 20:40:14
15	Poster Crimson Horizon Album Art	Poster	14.00	50	2025-12-18 20:40:14
16	Vinilo Echoes in Silence - Midnight Signals	Vinilo	29.00	38	2025-12-18 20:40:14
17	CD Echoes in Silence - Midnight Signals	CD	19.00	48	2025-12-18 20:40:14
18	Camiseta Echoes in Silence - Midnight Signals	Merchandising	23.00	75	2025-12-18 20:40:14
19	Pin Midnight Signals Logo	Merchandising	6.50	130	2025-12-18 20:40:14
20	Poster Echoes in Silence Album Art	Poster	12.50	55	2025-12-18 20:40:14
21	Vinilo Solar Flare - Electric Alley	Vinilo	31.00	33	2025-12-18 20:40:14
22	CD Solar Flare - Electric Alley	CD	20.50	42	2025-12-18 20:40:14
23	Camiseta Solar Flare - Electric Alley	Merchandising	24.50	70	2025-12-18 20:40:14
24	Pin Electric Alley Logo	Merchandising	6.00	120	2025-12-18 20:40:14


clientes.csv — Abrir con

id_cliente	nombre	email	fecha_registro	ciudad	pais
1	Raul Salvatierra	raul.salvatierra.35@gmail.com	2026-01-19 12:59:30	Brooklyn	USA
2	Maria Perez	maria@hotmail.com	2026-01-19 12:59:30	Buenos Aires	Argentina
3	Carlos Lopez	carlos@gmail.com	2026-01-19 12:59:30	Madrid	España
4	Gian Tenorio	gian.tenorio6@gmail.com	2026-01-19 12:59:30	Cali	Colombia
5	Brian Scheller	bscheller@gmail.com	2026-01-19 12:59:30	Brooklyn	USA
6	Juan Perez	juan.perez@gmail.com	2026-01-19 12:59:30	Buenos Aires	Argentina
7	Maria Gonzalez	maria.gonzalez@yahoo.com	2026-01-19 12:59:30	Santiago	Chile
8	Carlos Ramirez	carlos.ramirez@hotmail.com	2026-01-19 12:59:30	Lima	Peru
9	Ana Fernandez	ana.fernandez@gmail.com	2026-01-19 12:59:30	Bogotá	Colombia
10	Luis Martinez	luis.martinez@yahoo.com	2026-01-19 12:59:30	Quito	Ecuador
11	Sofia Lopez	sofia.lopez@hotmail.com	2026-01-19 12:59:30	Montevideo	Uruguay
12	Diego Sanchez	diego.sanchez@gmail.com	2026-01-19 12:59:30	Asunción	Paraguay
13	Camila Torres	camila.torres@yahoo.com	2026-01-19 12:59:30	La Paz	Bolivia
14	Jose Castillo	jose.castillo@hotmail.com	2026-01-19 12:59:30	Caracas	Venezuela
15	Valentina Herrera	valentina.herrera@gmail.com	2026-01-19 12:59:30	Ciudad de Mexico	Mexico
16	Ricardo Morales	ricardo.morales@yahoo.com	2026-01-19 12:59:30	Guadalajara	Mexico
17	Fernanda Diaz	fernanda.diaz@hotmail.com	2026-01-19 12:59:30	Medellín	Colombia
18	Gabriel Ortiz	gabriel.ortiz@gmail.com	2026-01-19 12:59:30	Santo Domingo	Republica Dominicana
19	Isabella Reyes	isabella.reyes@yahoo.com	2026-01-19 12:59:30	San Juan	Puerto Rico
20	Matias Rojas	matias.rojas@hotmail.com	2026-01-19 12:59:30	Montevideo	Uruguay
21	Andres Molina	andres.molina@gmail.com	2026-01-19 12:59:30	Rosario	Argentina
22	Lucia Benitez	lucia.benitez@yahoo.com	2026-01-19 12:59:30	Córdoba	Argentina
23	Pablo Aguirre	pablo.aguirre@hotmail.com	2026-01-19 12:59:30	Mendoza	Argentina
24	Mariana Soto	mariana.soto@gmail.com	2026-01-19 12:59:30	La Plata	Argentina

Clientes and productos correspond to two of the five CSV files imported from **MySQL Workbench,** used for data loading within the project.

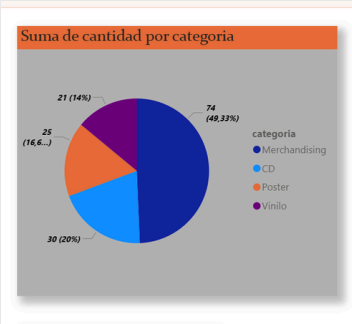


	A	B	C	D	E	F
1	id_cliente	nombre	email	fecha_registro	ciudad	pais
2	1	Raul Salvatierra	raul.salvatierra.3	2026-01-19 12:5	Brooklyn	USA
3	2	Maria Perez	maria@hotmail.c	2026-01-19 12:5	Buenos Aires	Argentina
4	3	Carlos Lopez	carlos@gmail.cc	2026-01-19 12:5	Madrid	España
5	4	Gian Tenorio	gian.tenorio6@g	2026-01-19 12:5	Cali	Colombia
6	5	Brian Scheller	bscheller@gmai	2026-01-19 12:5	Brooklyn	USA
7	6	Juan Perez	juan.perez@gm	2026-01-19 12:5	Buenos Aires	Argentina
8	7	Maria Gonzalez	maria.gonzalez@	2026-01-19 12:5	Santiago	Chile
9	8	Carlos Ramirez	carlos.ramirez@	2026-01-19 12:5	Lima	Peru
10	9	Ana Fernandez	ana.fernandez@	2026-01-19 12:5	Bogotá	Colombia
11	10	Luis Martinez	luis.martinez@y	2026-01-19 12:5	Quito	Ecuador
12	11	Sofia Lopez	sofia.lopez@hot	2026-01-19 12:5	Montevideo	Uruguay
13	12	Diego Sanchez	diego.sanchez@	2026-01-19 12:5	Asunción	Paraguay
14	13	Camila Torres	camila.torres@y	2026-01-19 12:5	La Paz	Bolivia
15	14	Jose Castillo	jose.castillo@ho	2026-01-19 12:5	Caracas	Venezuela
16	15	Valentina Herrer	valentina.herrer	2026-01-19 12:5	Ciudad de Mexic	Mexico

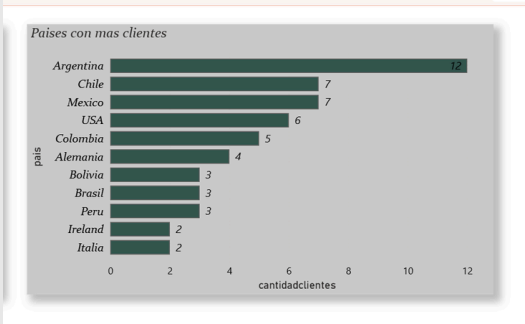
In the **Excel** project, the dataset can be viewed, and it was later imported into **Power BI** for creating charts and visualizations.

POWER BI VISUALIZATIONS

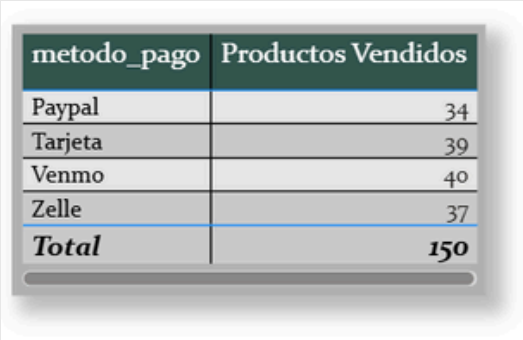
Interactive visualizations were created in **Power BI** using **DAX** measures, aiming to enhance data visibility and analysis, allowing the identification of **patterns, trends, and relevant comparisons** based on the project data.



A pie chart shows the distribution of sales by **product type,** with merchandising accounting for nearly **50% of the total**, while **CDs, posters, and vinyl** share the remaining half in a balanced way.



The bar chart shows the number of **customers by country**, displaying only the 11 countries with the highest number of customers. **Argentina** leads with 12 customers, followed by **Chile** and **Mexico** with 7, the **United States** with 6, and **Colombia** with 5 customers.



metodo_pago	Productos Vendidos
Paypal	34
Tarjeta	39
Venmo	40
Zelle	37
Total	*150*

The **table** shows the **payment method along** with the number of products sold, with **Venmo** leading at 40 units, followed by **Card** with 39, **Zelle** with 37, and **PayPal** with 34 products sold.4 productos vendidos.

https://github.com/julesdaud/mysql-entrega-final