TIENDA GAMEEXPRESS









DAVID GONZÁLEZ CÓRDOBA

2024-2025

1-DAW Mañana (Base de Datos)

ÍNDICE

INTRODUCCIÓN	3
MODELO ENTIDAD RELACIÓN	4
MODELO RELACIONAL	8
CARGA MASIVA	15
CONSULTAS	
CONCLUSIÓN	28

INTRODUCCIÓN

Este proyecto se centra en diseñar un modelo para una tienda de videojuegos llamada **GameExpress**, que vende productos tanto en tiendas físicas como en línea. El objetivo principal es organizar y gestionar de manera eficiente las actividades de la tienda, como ventas, pedidos, inventarios, empleados y clientes.

GameExpress busca ofrecer una experiencia de compra sencilla y cómoda, ya sea presencial o digital, adaptándose a las necesidades de los clientes. Por eso, he creado un modelo que cubre las principales necesidades de la tienda:

- **Gestión de juegos**: Cada tienda tiene un stock de juegos que es abastecido por varias empresas proveedoras. Esto asegura que siempre haya juegos disponibles según la demanda.
- **Pedidos**: Los pedidos pueden ser de dos tipos:
 - **Físicos**: El cliente recoge el producto en la tienda.
 - **En línea**: Los productos se entregan en la dirección que indique el cliente.
- **Gestión de empleados**: La tienda cuenta con trabajadores que tienen diferentes tareas, como gestionar pedidos que solicitan los clientes.
- **Clientes y socios**: Se almacena la información de los clientes y, se identifica si son socios o no.
- **Empresa**: Las empresas suministran los juegos que se venden en las tiendas, siendo esenciales para el funcionamiento del negocio.
- **Dirección:** Almacena donde se ubican los clientes, empleados, tiendas, pedidos online y empresas.

Objetivos del proyecto

- 1. Diseñar un **modelo entidad-relación (MER)** que cumpla con las necesidades de gestionar la tienda.
- 2. Garantizar la transformación efectiva del MER al **modelo relacional**, manteniendo la integridad de los datos y las relaciones.
- 3. Implementar y ejecutar la base de datos en herramientas como **MySQL Workbench** y **DBeaver**.
- 4. Generar una base de datos funcional con una carga masiva de datos (mínimo 2 con 1000 registros) que permita pruebas realistas.
- 5. Crear **5 consultas SQL multitabla**, agrupadas y/o subconsultas, **2 vistas**, **2 funciones**, **3 procedimientos y 2 triggers** para evaluar el rendimiento y la utilidad de la base de datos en contextos reales.
- 6. Subir 3 ficheros a GitHub (esquema, datos y consultas).
- 7. Alojar la base de datos a AWS.

MODELO ENTIDAD RELACIÓN

El modelo entidad-relación de GameExpress incluye las entidades principales que intervienen en el funcionamiento de la tienda, así como las relaciones entre ellas. Estas entidades representan tanto los aspectos físicos (juegos, tiendas) como los elementos administrativos (empleados, clientes, proveedores, pedidos, direcciones). Las relaciones entre las entidades permiten realizar procesos clave como la venta, la gestión de pedidos y el abastecimiento por parte de las empresas.

A continuación, se presenta el diagrama entidad-relación:

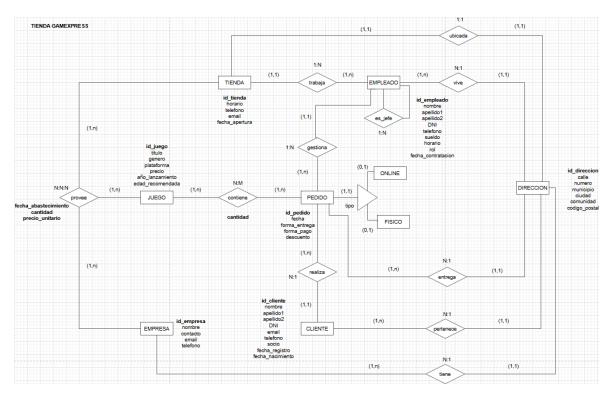


Diagrama entidad-relación de la tienda GameExpress.

Explicación detallada del modelo

Entidades Principales

Tienda

Representa las tiendas de GameExpress. Atributos:

- id_tienda: Clave primaria.
- horario, teléfono, email, fecha_apertura.

Juego

Representa los videojuegos disponibles en cada tienda. Atributos:

- **id_juego**: Clave primaria.
- título, género, plataforma, precio, año_lanzamiento, edad_recomendada.

Empleado

Representa al personal que trabaja en cada tienda. Atributos:

- **id_empleado**: Clave primaria.
- nombre, apellido1, apellido2, DNI, teléfono, sueldo, horario, rol, fecha_contratación.

Cliente

Representa a los clientes de la tienda. Atributos:

- **id_cliente**: Clave primaria.
- nombre, apellido1, apellido2, DNI, email, teléfono, fecha_registro, fecha_nacimiento, socio (indica si es cliente frecuente).

Empresa

Representa a las empresas que abastecen a GameExpress de videojuegos. Atributos:

- **id_empresa**: Clave primaria.
- nombre, contacto, email, teléfono.

Pedido

Registra las compras realizadas por los clientes. Atributos:

- **id_pedido**: Clave primaria.
- fecha, forma_entrega, forma_pago, descuento.

Estos pedidos se subdividen en dos:

- **Físicos**: El cliente recoge el producto en la tienda.
- Online: Los productos se entregan en el domicilio del cliente.

Dirección

Almacena información sobre las direcciones (ubicaciones) de todo. Atributos:

- **id_ dirección**: Clave primaria.
- calle, número, municipio, ciudad, comunidad, código_postal.

Relaciones Principales

Provee (Tienda - Juego - Empresa)

Indica qué videojuegos están disponibles en cada tienda y las empresas que lo abastecen. Atributos:

• cantidad, fecha_abastecimiento, precio_unitario.

Contiene (Pedido - Juego)

Relaciona un pedido con los videojuegos adquiridos. Atributos:

cantidad.

Realiza (Cliente - Pedido)

Relaciona un cliente con los pedidos que ha realizado.

Trabaja (Empleado - Tienda)

Relaciona a los empleados con la tienda donde trabajan.

Es jefe (Empleado - Empleado)

Define relación reflexiva entre empleados (qué empleado es jefe de otro).

Vive (Empleado/Cliente - Dirección)

Relaciona a los empleados y clientes con sus respectivas direcciones.

Entrega (Pedido - Dirección)

Relaciona el pedido donde se entrega en que ubicación.

Ubicada (Tienda/Empresa - Dirección)

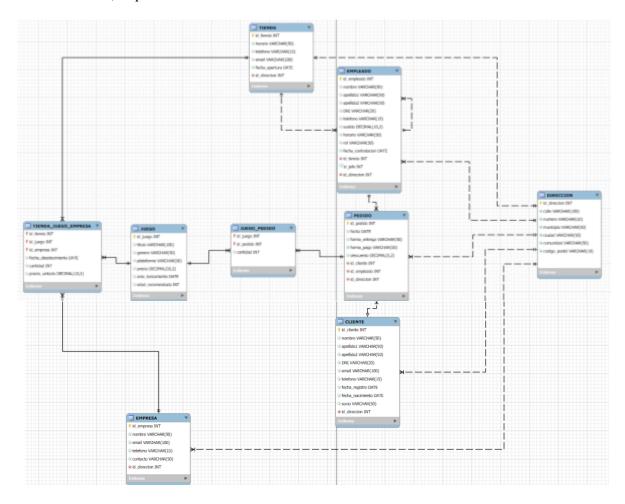
Relaciona a las tiendas y empresas con sus respectivas ubicaciones.

Gestiona (Empleado - Pedido)

Relaciona al empleado con los pedidos que gestiona.

MODELO RELACIONAL

A continuación, se presenta el modelo relacional:



Modelo relacional de la tienda GameExpress (Workbench).

Script de creación de tablas

```
-- MySQL Script generated by MySQL Workbench
-- Sat Jan 4 19:59:03 2025
-- Model: New Model Version: 1.0
-- MySQL Workbench Forward Engineering
SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;
SET @OLD_SQL_MODE=@@SQL_MODE,
SQL_MODE='ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,NO_ZERO_IN_DATE,NO_ZERO_DATE,
ERROR_FOR_DIVISION_BY_ZERO,NO_ENGINE_SUBSTITUTION';
------
-- Schema GameExpress
.. .....
CREATE SCHEMA IF NOT EXISTS GameExpress DEFAULT CHARACTER SET utf8;
USE GameExpress:
-- Table GameExpress.DIRECCION
------
CREATE TABLE IF NOT EXISTS GameExpress.DIRECCION (
id_direccion INT NOT NULL AUTO_INCREMENT,
calle VARCHAR(100) NOT NULL,
numero VARCHAR(10) NOT NULL,
municipio VARCHAR(50) NOT NULL,
ciudad VARCHAR(50) NOT NULL,
comunidad VARCHAR(50) NOT NULL,
codigo_postal VARCHAR(10) NOT NULL,
PRIMARY KEY (id_direccion)
) ENGINE = InnoDB;
------
-- Table GameExpress.TIENDA
------
CREATE TABLE IF NOT EXISTS GameExpress.TIENDA (
id_tienda INT NOT NULL AUTO_INCREMENT,
horario VARCHAR(50) NOT NULL,
telefono VARCHAR(15) NOT NULL,
email VARCHAR(100) NOT NULL,
fecha_apertura DATE NOT NULL,
id_direccion INT NOT NULL,
PRIMARY KEY (id_tienda),
CONSTRAINT fk_TIENDA_DIRECCION
 FOREIGN KEY (id_direccion)
 REFERENCES GameExpress.DIRECCION (id_direccion)
 ON DELETE CASCADE
 ON UPDATE CASCADE
) ENGINE = InnoDB;
______
```

-- Table GameExpress.EMPLEADO ------CREATE TABLE IF NOT EXISTS GameExpress.EMPLEADO (id empleado INT NOT NULL AUTO INCREMENT, nombre VARCHAR(50) NOT NULL, apellido1 VARCHAR(50) NOT NULL, apellido2 VARCHAR(50) NOT NULL, DNI VARCHAR(20) NOT NULL, telefono VARCHAR(15) NOT NULL, sueldo DECIMAL(10,2) NOT NULL, horario VARCHAR(50) NOT NULL, rol VARCHAR(50) NOT NULL, fecha_contratacion DATE NOT NULL, id_tienda INT NOT NULL, id_jefe INT NULL, id_direccion INT NOT NULL, PRIMARY KEY (id_empleado), **CONSTRAINT fk_EMPLEADO_TIENDA** FOREIGN KEY (id_tienda) REFERENCES GameExpress.TIENDA (id_tienda) ON DELETE CASCADE ON UPDATE CASCADE, **CONSTRAINT fk_EMPLEADO_DIRECCION FOREIGN KEY (id_direccion)** REFERENCES GameExpress.DIRECCION (id_direccion) ON DELETE CASCADE ON UPDATE CASCADE, **CONSTRAINT fk_EMPLEADO_JEFE FOREIGN KEY (id_jefe)** REFERENCES GameExpress.EMPLEADO (id_empleado) ON DELETE CASCADE **ON UPDATE CASCADE**) **ENGINE** = InnoDB; -- Table GameExpress.CLIENTE CREATE TABLE IF NOT EXISTS GameExpress.CLIENTE (id_cliente INT NOT NULL AUTO_INCREMENT, nombre VARCHAR(50) NOT NULL, apellido1 VARCHAR(50) NOT NULL, apellido2 VARCHAR(50) NOT NULL, DNI VARCHAR(20) NOT NULL, email VARCHAR(100) NOT NULL, telefono VARCHAR(15) NOT NULL, fecha_registro DATE NOT NULL, fecha_nacimiento DATE NOT NULL, socio VARCHAR(50) NOT NULL, id_direccion INT NOT NULL,

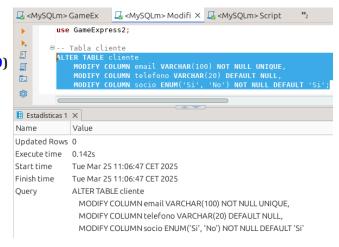
```
PRIMARY KEY (id_cliente),
CONSTRAINT fk_CLIENTE_DIRECCION
 FOREIGN KEY (id_direccion)
 REFERENCES GameExpress.DIRECCION (id_direccion)
 ON DELETE CASCADE
 ON UPDATE CASCADE
) ENGINE = InnoDB;
------
-- Table GameExpress.PEDIDO
------
CREATE TABLE IF NOT EXISTS GameExpress.PEDIDO (
id_pedido INT NOT NULL AUTO_INCREMENT,
fecha DATE NOT NULL,
forma_entrega VARCHAR(50) NOT NULL,
forma_pago VARCHAR(50) NOT NULL,
descuento DECIMAL(5,2) NULL,
id_cliente INT NOT NULL,
id_empleado INT NOT NULL,
id_direccion INT NOT NULL,
PRIMARY KEY (id_pedido),
CONSTRAINT fk_PEDIDO_CLIENTE
 FOREIGN KEY (id_cliente)
 REFERENCES GameExpress.CLIENTE (id_cliente)
 ON DELETE CASCADE
 ON UPDATE CASCADE,
CONSTRAINT fk_PEDIDO_EMPLEADO
 FOREIGN KEY (id_empleado)
 REFERENCES GameExpress.EMPLEADO (id_empleado)
 ON DELETE CASCADE
 ON UPDATE CASCADE.
CONSTRAINT fk_PEDIDO_DIRECCION
 FOREIGN KEY (id_direccion)
 REFERENCES GameExpress.DIRECCION (id_direccion)
 ON DELETE CASCADE
 ON UPDATE CASCADE
) ENGINE = InnoDB;
------
-- Table GameExpress.EMPRESA
CREATE TABLE IF NOT EXISTS GameExpress.EMPRESA (
id empresa INT NOT NULL AUTO INCREMENT,
nombre VARCHAR(50) NOT NULL,
email VARCHAR(100) NOT NULL,
telefono VARCHAR(15) NOT NULL,
contacto VARCHAR(50) NOT NULL,
id_direccion INT NOT NULL,
PRIMARY KEY (id_empresa),
CONSTRAINT fk_EMPRESA_DIRECCION
```

```
FOREIGN KEY (id_direccion)
 REFERENCES GameExpress.DIRECCION (id_direccion)
 ON DELETE CASCADE
 ON UPDATE CASCADE
) ENGINE = InnoDB;
------
-- Table GameExpress.JUEGO
------
CREATE TABLE IF NOT EXISTS GameExpress.JUEGO (
id_juego INT NOT NULL AUTO_INCREMENT,
titulo VARCHAR(100) NOT NULL,
genero VARCHAR(50) NOT NULL,
plataforma VARCHAR(50) NOT NULL,
precio DECIMAL(10,2) NOT NULL,
anio_lanzamiento DATE NOT NULL,
edad_recomendada INT NOT NULL,
PRIMARY KEY (id_juego)
) ENGINE = InnoDB;
------
-- Table GameExpress.TIENDA_JUEGO_EMPRESA
------
CREATE TABLE IF NOT EXISTS GameExpress.TIENDA_JUEGO_EMPRESA (
id_tienda INT NOT NULL,
id_juego INT NOT NULL,
id_empresa INT NOT NULL,
fecha_abastecimiento DATE NOT NULL,
cantidad INT NOT NULL,
precio_unitario DECIMAL(10,2) NOT NULL,
PRIMARY KEY (id_tienda, id_juego, id_empresa),
CONSTRAINT fk_TIENDA_JUEGO_EMPRESA_TIENDA
 FOREIGN KEY (id_tienda)
 REFERENCES GameExpress.TIENDA (id_tienda)
 ON DELETE CASCADE
 ON UPDATE CASCADE,
CONSTRAINT fk_TIENDA_JUEGO_EMPRESA_JUEGO
 FOREIGN KEY (id_juego)
 REFERENCES GameExpress.JUEGO (id_juego)
 ON DELETE CASCADE
 ON UPDATE CASCADE.
CONSTRAINT fk_TIENDA_JUEGO_EMPRESA_EMPRESA
 FOREIGN KEY (id empresa)
 REFERENCES GameExpress.EMPRESA (id_empresa)
 ON DELETE CASCADE
 ON UPDATE CASCADE
) ENGINE = InnoDB;
------
-- Table GameExpress.JUEGO_PEDIDO
______
```

```
CREATE TABLE IF NOT EXISTS GameExpress.JUEGO_PEDIDO (
id_juego INT NOT NULL,
id_pedido INT NOT NULL,
cantidad INT NOT NULL,
PRIMARY KEY (id_juego, id_pedido),
CONSTRAINT fk_JUEGO_PEDIDO_JUEGO
 FOREIGN KEY (id_juego)
 REFERENCES GameExpress.JUEGO (id_juego)
 ON DELETE CASCADE
 ON UPDATE CASCADE.
CONSTRAINT fk_JUEGO_PEDIDO_PEDIDO
 FOREIGN KEY (id_pedido)
 REFERENCES GameExpress.PEDIDO (id_pedido)
 ON DELETE CASCADE
 ON UPDATE CASCADE
) ENGINE = InnoDB;
SET SQL_MODE=@OLD_SQL_MODE;
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
```

Modificación de tablas

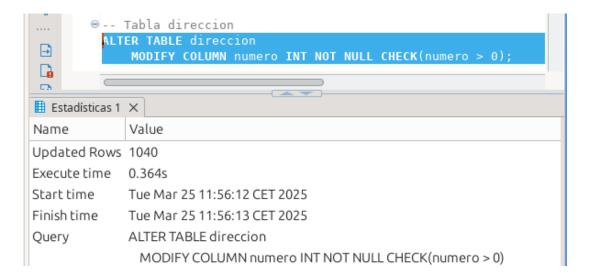
-- Tabla cliente
ALTER TABLE cliente
MODIFY COLUMN email VARCHAR(100)
NOT NULL UNIQUE,
MODIFY COLUMN telefono VARCHAR(20)
DEFAULT NULL,
MODIFY COLUMN socio ENUM('Si', 'No')
NOT NULL DEFAULT 'Si';



-- Tabla direccion

ALTER TABLE direction

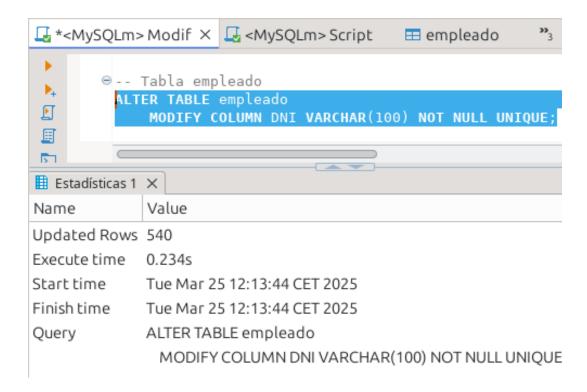
MODIFY COLUMN numero INT NOT NULL CHECK(numero > 0);



-- Tabla empleado

ALTER TABLE empleado

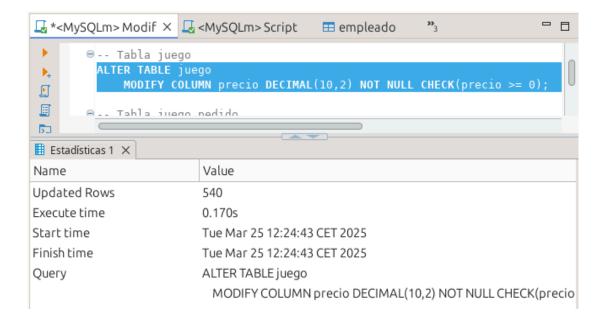
MODIFY COLUMN DNI VARCHAR(100) NOT NULL UNIQUE;



-- Tabla juego

ALTER TABLE juego

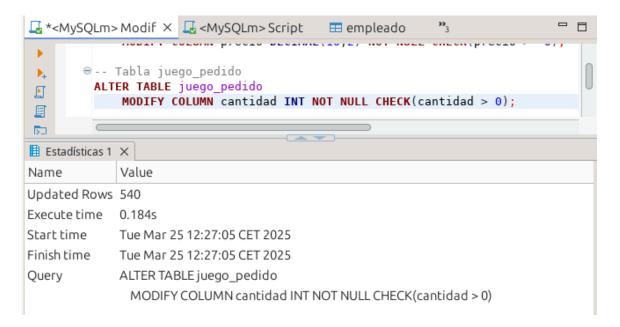
MODIFY COLUMN precio DECIMAL(10,2) NOT NULL CHECK(precio >= 0);



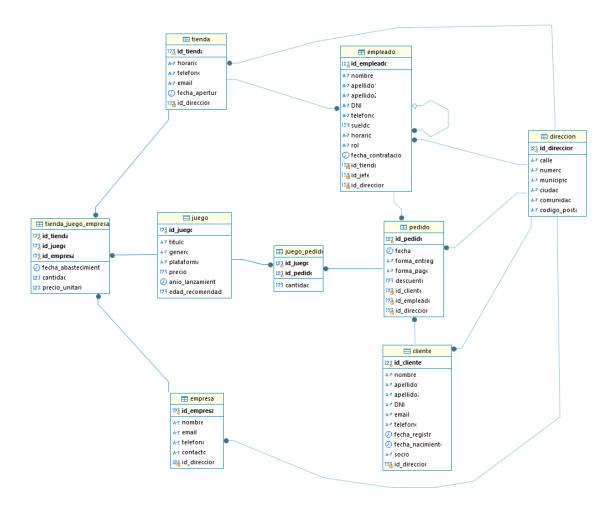
-- Tabla juego_pedido

ALTER TABLE juego_pedido

MODIFY COLUMN cantidad INT NOT NULL CHECK(cantidad > 0);



Ejecución del script en DBeaver

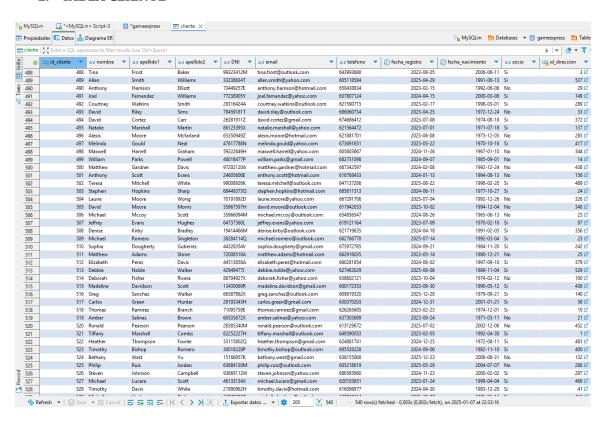


Script ejecutado de la tienda GameExpress (DBeaver).

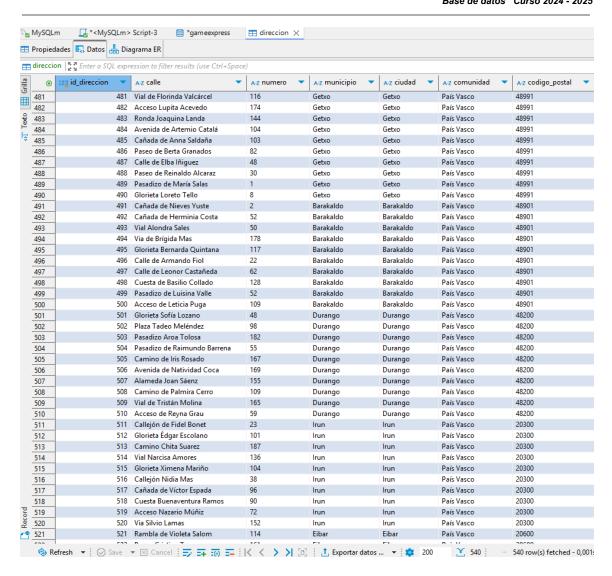
CARGA MASIVA

A continuación, se presenta la carga masiva de cada tabla con una media de 500 registros para cada una.

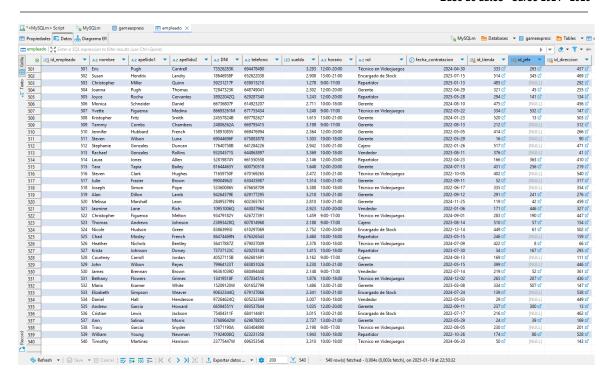
1. TABLA CLIENTE



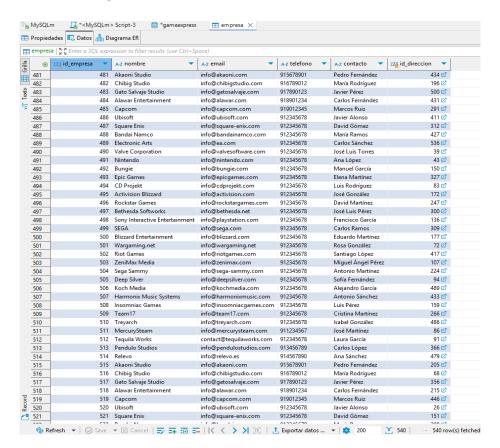
2. TABLA DIRECCIÓN (1000 registros)



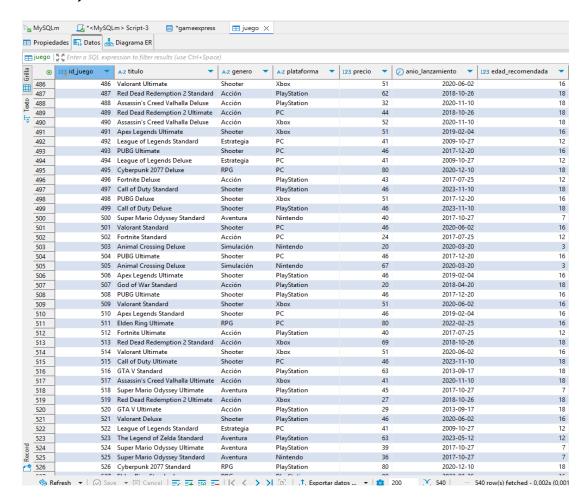
3. TABLA EMPLEADO



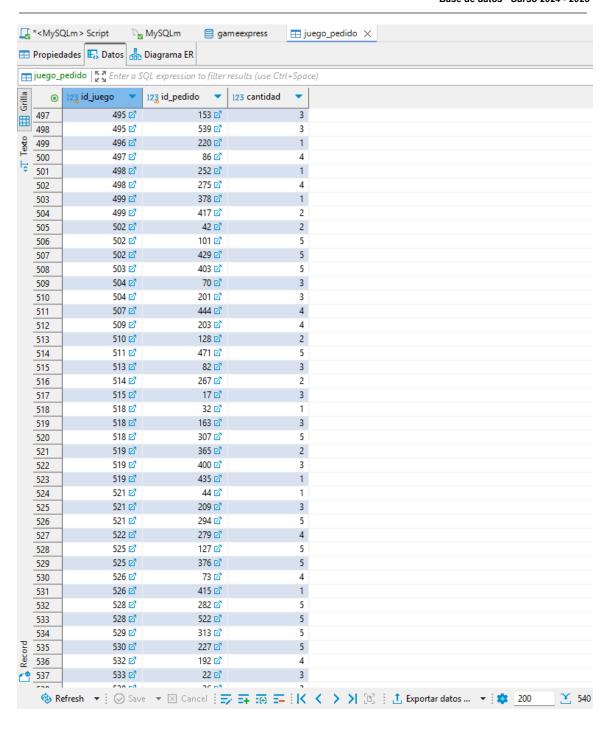
4. TABLA EMPRESA



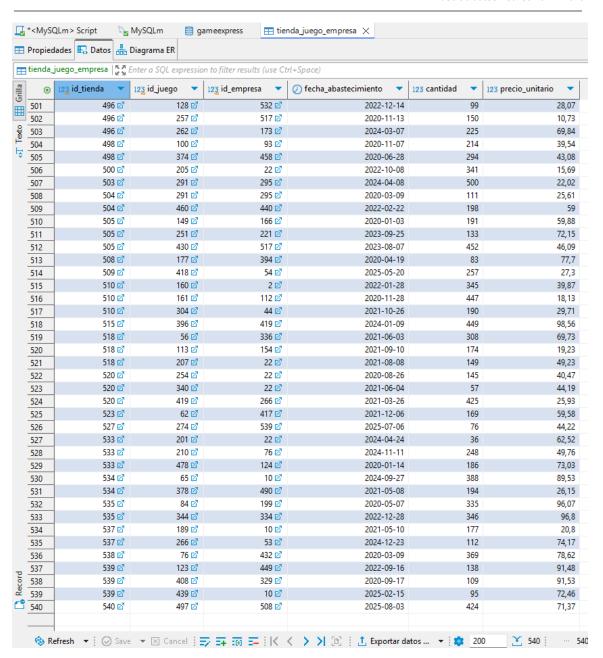
5. TABLA JUEGO



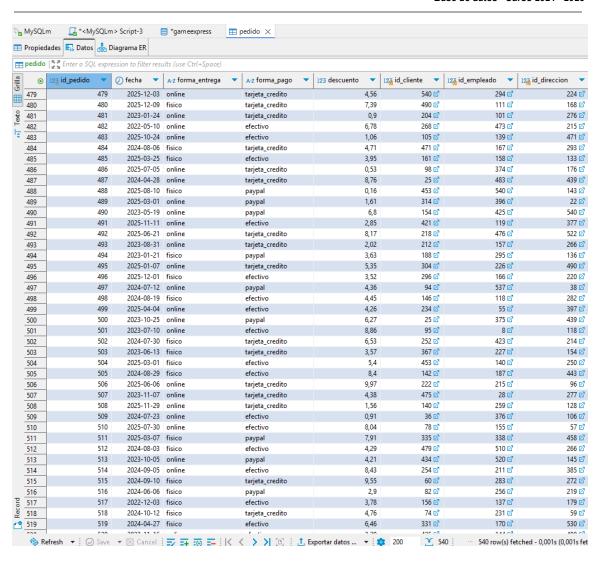
6. TABLA JUEGO_PEDIDO



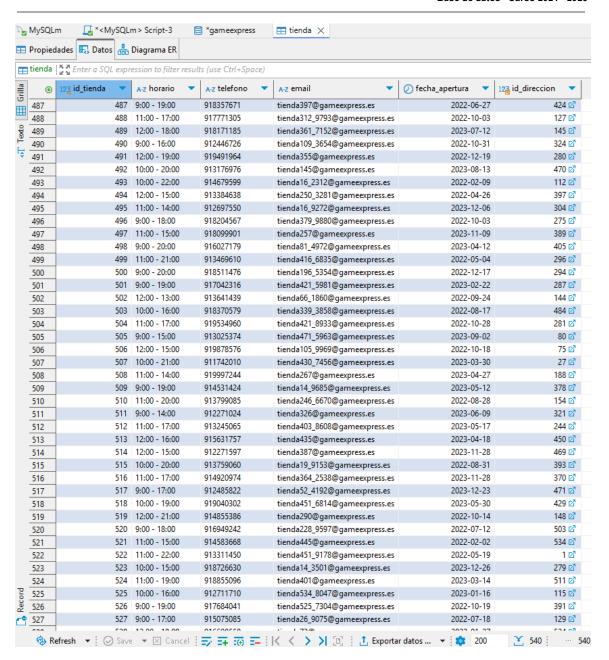
7. TABLA TIENDA_JUEGO_EMPRESA



8. TABLA PEDIDO



9. TABLA TIENDA



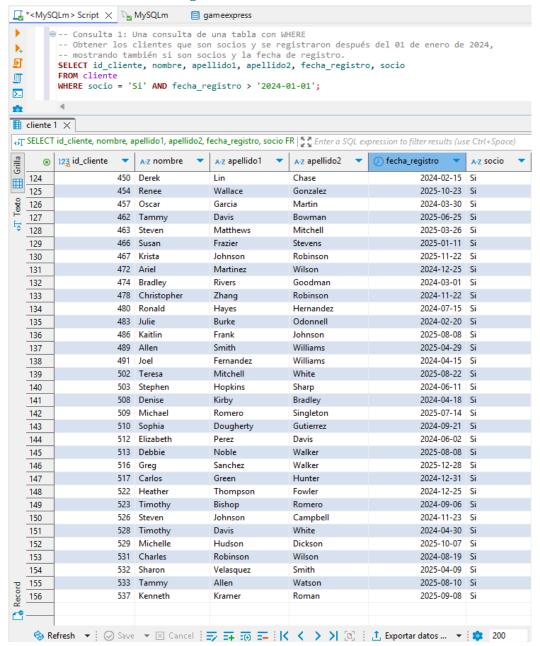
CONSULTAS

- -- Consulta 1: Una consulta de una tabla con WHERE
- -- Obtener los clientes que son socios y se registraron después del 01 de enero de 2024,
- -- mostrando también si son socios y la fecha de registro.

SELECT id_cliente, nombre, apellido1, apellido2, fecha_registro, socio

FROM cliente

WHERE socio = 'Si' AND fecha_registro > '2024-01-01';

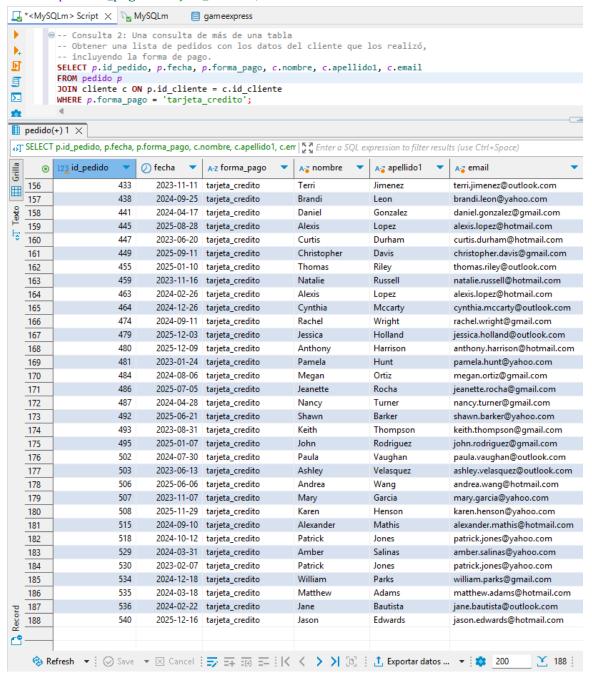


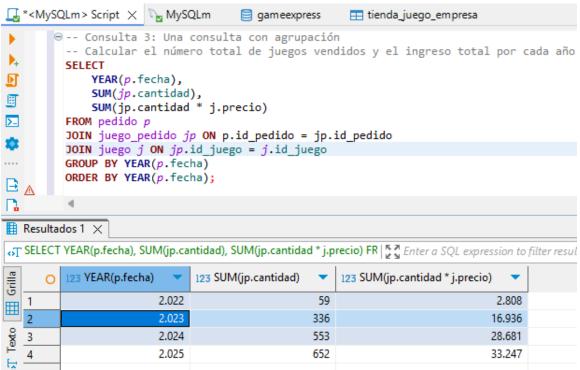
- -- Consulta 2: Una consulta de más de una tabla
- -- Obtener una lista de pedidos con los datos del cliente que los realizó,
- -- incluyendo la forma de pago.

SELECT *p*.id_pedido, *p*.fecha, *p*.forma_pago, *c*.nombre, *c*.apellido1, *c*.email **FROM** pedido *p*

JOIN cliente c **ON** p.id_cliente = c.id_cliente

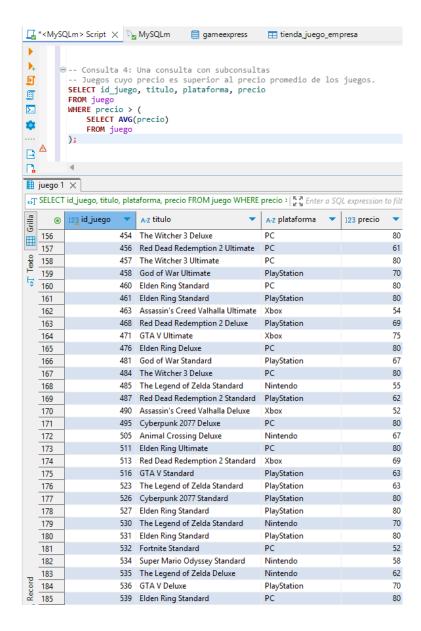
WHERE *p*.forma_pago = 'tarjeta_credito';





- -- Consulta 4: Una consulta con subconsultas
- -- Juegos cuyo precio es superior al precio promedio de los juegos.

```
SELECT id_juego, titulo, plataforma, precio
FROM juego
WHERE precio > (
    SELECT AVG(precio)
    FROM juego
);
```



- -- Consulta 5: Una que combine varias anteriores
- -- Listar los juegos más vendidos en 2025 y las tiendas donde más se vendieron

SELECT *j*.titulo **AS** *juego*, t.id_tienda **AS** *tienda*, **SUM**(*jp*.cantidad) **AS** *total_vendidos* **FROM** pedido *p*

JOIN juego_pedido *jp* **ON** *p*.id_pedido = *jp*.id_pedido

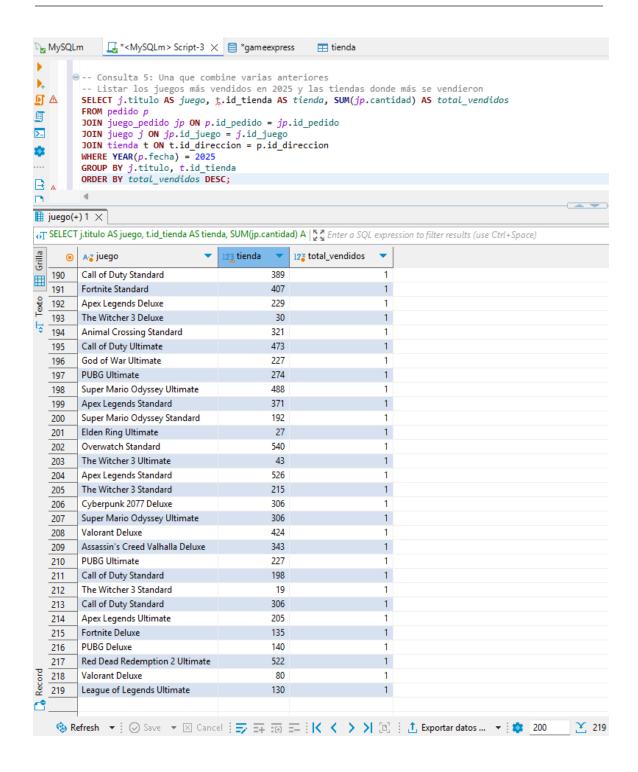
JOIN juego *j* **ON** *jp*.id_juego = *j*.id_juego

JOIN tienda t **ON** t.id_direccion = p.id_direccion

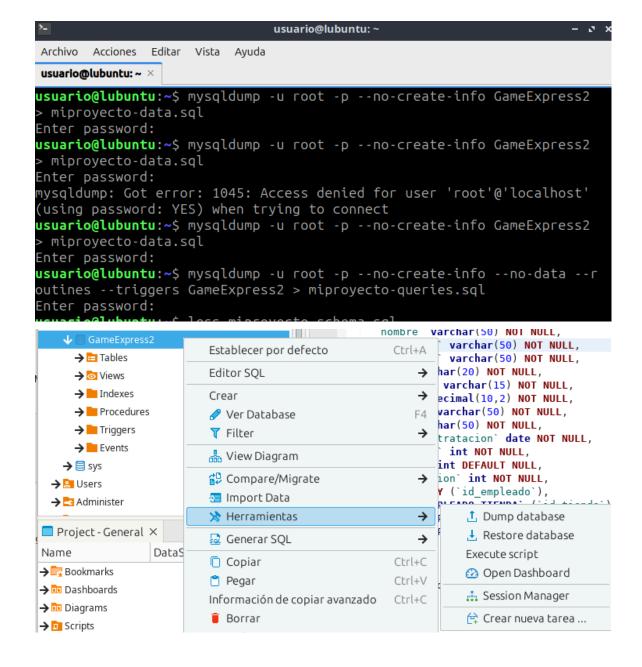
WHERE YEAR(p.fecha) = 2025

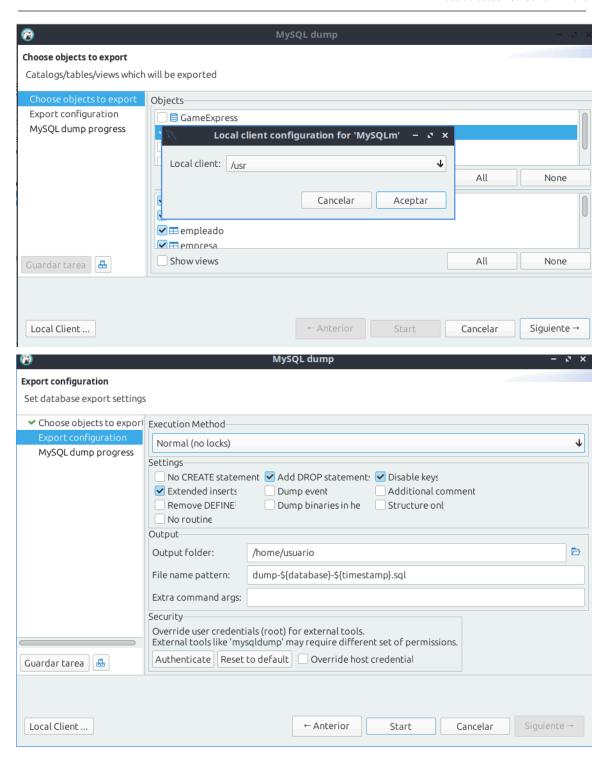
GROUP BY *j*.titulo, *t*.id_tienda

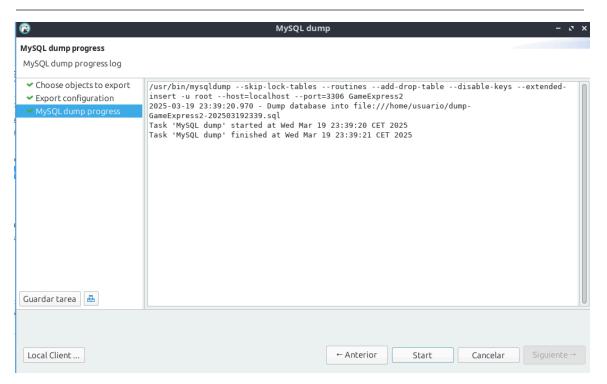
ORDER BY total_vendidos **DESC**;



PROYECTO (2 trimestre)







CONCLUSIÓN

Conclusión del Proyecto

En conclusión, este proyecto me ha permitido desarrollar habilidades técnicas clave, como el diseño conceptual y lógico de bases de datos, la implementación práctica en sistemas de gestión de bases de datos y la manipulación de datos mediante SQL. Además, me ayudó a reforzar mi capacidad de resolver problemas.

Problema con el Campo jefe en la Tabla empleado

Durante la inserción de datos en la tabla empleado, surgieron errores relacionados con la clave foránea id_jefe. Este campo hacía referencia al id_empleado de otro registro, creando dependencias complejas. El principal problema fue:

• Duplicados en id_empleado, que invalidaron las referencias de id_jefe.

Se ha eliminado el costo_total ya que habría que realizar operaciones muy complejas y haciendo consultas podremos descubrirlo.