Proyecto Final SQL

DNI: 45.175.053 | Comisión: 59410

E-commerce de videojuegos “Gamehub”

Lautaro Ezequiel Santillan

2024

# **Introducción**

El proyecto aborda el diseño y desarrollo de una base de datos para **GameHub**, un ecommerce especializado en la venta de videojuegos. La base de datos está diseñada para gestionar de manera eficiente la información clave, incluyendo productos (videojuegos), clientes, pedidos, plataformas, y categorías de videojuegos.

## **Modelo de negocio**

**GameHub** es un ecommerce dedicado a la venta de videojuegos para diversas consolas y PC, ofreciendo una amplia gama de títulos en distintas categorías y plataformas.

## **Planteamiento del problema**

En un entorno de ecommerce, la gestión eficiente de productos, pedidos, y clientes es crucial para el éxito del negocio. La falta de una base de datos bien estructurada puede llevar a errores en las transacciones, problemas con el inventario, y dificultades en la gestión de clientes. Este proyecto busca resolver estos problemas mediante el diseño de una base de datos que optimiza todos estos procesos.

## **Objetivo**

El objetivo principal es crear una estructura de base de datos robusta que permita manejar eficientemente las transacciones de compra, administrar el inventario de videojuegos y gestionar la información de los clientes.

# **Diagrama Entidad-Relación (DER)**

# **Modelo Relacional**

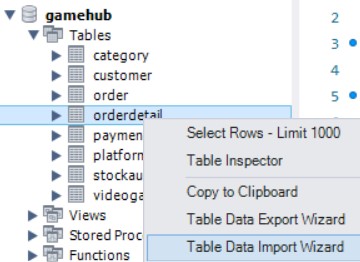
# **Listado de las tablas**

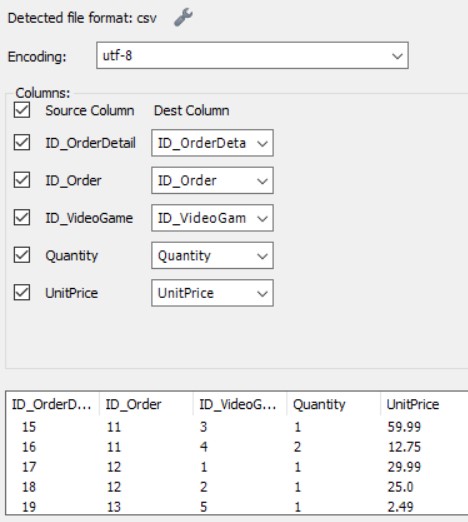
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **TABLAS** | **CAMPOS** | **DESCRIPCIÓN** | **PK / FK** | **TIPO DE DATO** |
| **Customer** | ID\_Customer | Identificador único del cliente | PK | INT |
| FirstName | Nombre del cliente |  | VARCHAR(75) |
| LastName | Apellido del cliente |  | VARCHAR(75) |
| DNI | Número de identificación del cliente |  | INT |
| Email | Dirección de correo electrónico |  | VARCHAR(100) |
| Location | Zona de residencia del cliente |  | VARCHAR(100) |
| Phone | Número de contacto |  | VARCHAR(20) |
| ID\_PaymentMethod | Identificador del método de pago | FK | INT |
| **PaymentMethod** | ID\_PaymentMethod | Identificador único del método de pago | PK | INT |
| Type | Tipo de método de pago (Tarjeta de crédito/débito, PayPal, etc.) |  | VARCHAR(50) |
| **Order** | ID\_Order | Identificador único del pedido | PK | INT |
| OrderDate | Fecha del pedido |  | DATE |
| State | Estado actual del pedido (Pendiente o Entregado) |  | ENUM |
| TotalAmount | Monto total del pedido |  | DECIMAL |
| ID\_Customer | Identificador del cliente que realizó el pedido | FK | INT |
| **VideoGame** | ID\_VideoGame | Identificador único del videojuego | PK | INT |
| Title | Título del videojuego |  | VARCHAR(100) |
| Descriptions | Breve descripción del videojuego |  | VARCHAR(150) |
| ReleaseDate | Fecha de lanzamiento del videojuego |  | DATE |
| Price | Precio del videojuego |  | DECIMAL |
| Stock | Cantidad de unidades disponibles |  | INT |
| ID\_Plataform | Identificador de la plataforma | FK | INT |
| ID\_Category | Identificador de la categoría | FK | INT |
| **Plataform** | ID\_Plataform | Identificador único de la plataforma | PK | INT |
| PlatformName | Nombre de la plataforma de juego (PS5/PS4/XBOX/PC/ETC) |  | VARCHAR(15) |
| **Category** | ID\_Category | Identificador único de la categoría | PK | INT |
| CategoryName | Nombre de la categoría o género |  | VARCHAR(50) |
| **OrderDetail** | ID\_OrderDetail | Identificador único del detalle del pedido | PK | INT |
| ID\_Order | Identificador del pedido | FK | INT |
| ID\_VideoGame | Identificador del videojuego | FK | INT |
| Quantity | Cantidad de unidades compradas |  | INT |
| UnitPrice | Precio del videojuego en el momento de la compra |  | DECIMAL |
| Subtotal | Subtotal de unidades compradas (Cantidad \* Precio\_Unitario) |  | DECIMAL |
| **StockAudit** | ID\_Audit | Identificador único del detalle del pedido | PK | INT |
| ID\_VideoGame | Identificador del videojuego | FK | INT |
| DateStUp | Fecha de actualización del stock |  | DATETIME |
| OldStock | Stock del videojuego antes de la auditoría |  | INT |
| NewStock | Stock del videojuego después de la auditoría |  | INT |

A continuación, encontrarás el enlace donde se han dispuesto los scripts para la creación de las tablas y la inserción de datos:

👉 <https://github.com/LautaroSantillan/Proyecto-Final--SQL-Coderhouse>

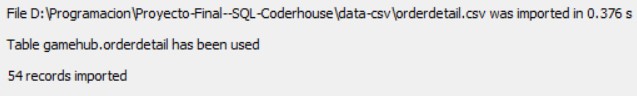
# **Inserción con Importación de datos**

1. Crear archivos como “banco” de datos que contengan toda la información que va a ser trasladada a la Base de Datos. En el caso de este proyecto son ‘**customer.csv**’, ‘**order.csv**’ y ‘**orderdetail.csv**’.
2. Luego seleccionar la tabla donde se le van a insertar los datos y con el click derecho luego seleccionar la opción de “**Table Data Import Wizard**”.
3. Luego seleccionar el path donde está alojado el archivo ‘**.csv’**.
4. A continuación, darle a next y seleccionar que quiere importar los datos a una tabla ya existente.



Y seguir dándole al botón next hasta que cargue todo y realice la importación.

1. Una vez que termine la inserción de datos saltaran unos mensajes diciendo el tiempo que se tardo y cuantos registros se importaron, y listo.



# **Listado de vistas, funciones, sp y triggers**

## **Vistas**

vw\_OrderSummary

* Objetivo: Mostrar un ranking de clientes basado en sus ventas acumuladas.
* Descripción: Esta vista combina las tablas Customer y Order para mostrar un resumen de las ventas totales por cliente cuyo estado de pedido es "Delivered". Agrupa a los clientes por su ID y nombre completo, mostrando el total de ventas acumuladas en orden descendente.
* Tablas involucradas: Customer, Order.

vw\_CustomerPayment

* Objetivo: Visualizar los clientes junto con sus métodos de pago, útil para informes de facturación y ventas.
* Descripción: Esta vista muestra detalles de los clientes, como nombre, apellido y correo electrónico, junto con el tipo de método de pago que usan. Utiliza una unión entre Customer y PaymentMethod.
* Tablas involucradas: Customer, PaymentMethod.

vw\_PendingOrders

* Objetivo: Listar todos los pedidos que aún están en estado "Pending".
* Descripción: Muestra los pedidos en estado pendiente, junto con su fecha de creación, monto total y el nombre del cliente que realizó el pedido.
* Tablas involucradas: Order, Customer.

vw\_CostsCustomer\_and\_PaymentMethod

* Objetivo: Proporcionar un informe de costos totales por cliente y método de pago, agrupado por mes.
* Descripción: Muestra un resumen mensual de los pedidos realizados por cada cliente, el método de pago utilizado, el costo total de los pedidos y la cantidad de pedidos realizados por cada cliente.
* Tablas involucradas: Order, Customer, PaymentMethod.

## **Funciones**

fn\_TotalSalesByCustomer

* Objetivo: Calcular el total de ventas realizadas por un cliente específico.
* Descripción: Recibe el ID del cliente como parámetro y devuelve el monto total de las ventas que ha realizado.
* Tablas involucradas: Order.

fn\_CalculateVIPDiscount

* Objetivo: Aplicar un descuento del 10% en el monto total de un pedido si el cliente tiene más de 5 pedidos.
* Descripción: Recibe el ID del cliente y el monto total del pedido. Si el cliente ha realizado más de 5 pedidos, aplica un descuento del 10% al monto total, de lo contrario, no aplica ningún descuento.
* Tablas involucradas: Order.

fn\_ValidateEmail

* Objetivo: Validar el formato del correo electrónico de un cliente.
* Descripción: Si el correo electrónico contiene el dominio "@example", lo reemplaza automáticamente por "@gmail" para asegurarse de que el formato del correo sea válido.
* Tablas involucradas: Ninguna (es una validación de formato).

## **Stored-Procedure (SP):**

sp\_UpdateOrderState

* Objetivo: Actualizar el estado de un pedido.
* Descripción: Permite actualizar el estado de un pedido a "Pending" o "Delivered". Si el nuevo estado es inválido, arroja un error.
* Tablas involucradas: Order.

sp\_UpdateVideoGamePrice

* Objetivo: Actualizar el precio de un videojuego.
* Descripción: Recibe el ID del videojuego y el nuevo precio. Verifica que el nuevo precio sea mayor que el precio actual y mayor a cero, de lo contrario, arroja un error.
* Tablas involucradas: VideoGame.

## **Triggers**

after\_update\_AuditStockChange

* Objetivo: Registrar los cambios de stock en la tabla StockAudit.
* Descripción: Cada vez que el stock de un videojuego es actualizado, se registra el cambio en una tabla de auditoría.
* Tablas involucradas: VideoGame, StockAudit.

after\_insert\_ReduceStockAfterOrderDetail

* Objetivo: Reducir el stock de un videojuego cuando se inserte un nuevo OrderDetail.
* Descripción: Cada vez que se inserta un detalle de pedido, el stock del videojuego correspondiente se reduce en función de la cantidad solicitada.
* Tablas involucradas: OrderDetail, VideoGame.

after\_delete\_RestoreStockAfterOrderDeletion

* Objetivo: Restaurar el stock de un videojuego cuando se elimina un detalle de pedido.
* Descripción: Cuando se elimina un registro de OrderDetail, el stock del videojuego se incrementa de acuerdo con la cantidad eliminada.
* Tablas involucradas: OrderDetail, VideoGame.

before\_insert\_CustomerEmail

* Objetivo: Validar el formato del correo electrónico antes de insertar un nuevo cliente.
* Descripción: Antes de insertar un nuevo cliente, verifica si el correo electrónico contiene "@example" y lo reemplaza por "@gmail".
* Tablas involucradas: Customer.sss

# **Usuarios y Comandos TCL**

## **En ‘users-DCL.sql’**

Este script configura los usuarios de la base de datos y administra el acceso:

* **master**: El usuario master tiene acceso completo sobre todas las tablas, vistas y funciones de la base de datos GameHub. Además, se le otorga la opción WITH GRANT OPTION para que pueda delegar permisos a otros usuarios.
* **user1**: Fue configurado con permisos específicos de lectura en tablas seleccionadas, vistas y funciones que permiten el análisis de datos de clientes y el monitoreo de órdenes. Posteriormente, se realizan las revocaciones de estos permisos y se elimina el usuario:

Por ejemplo, para crear el usuario **master** con todos los permisos se realizó así:

CREATE USER 'master'@'localhost' IDENTIFIED BY 'passmaster123';

GRANT ALL PRIVILEGES ON gameHub.\* TO 'master'@'localhost' WITH GRANT OPTION;

## **En ‘user1sql’**

Este archivo realiza pruebas en transacciones mediante el uso de puntos de control (SAVEPOINT) y consulta de vistas y funciones específicas.

Cada sección utiliza SAVEPOINT, COMMIT y ROLLBACK para definir una operación de consulta.

Esta implementación asegura la prueba de seguridad y control de datos en la base de datos GameHub.