|  |  |  |
| --- | --- | --- |
| Proyecto final base de datos | | Aula Campus |
| Download Counter Strike 2 Steam | | |
|  |  | |
| YEHOR BURLACHENKO | yebur@campusaula.com  github.com/krypton1312 | |

Indice

1. **Introduccion.**

El objetivo de este proyecto es crear y utilizar una base de datos para gestionar y analizar las partidas de Counter-Strike 2 (CS2), un juego muy popular en la escena de los eSports. La base de datos permitirá almacenar datos específicos sobre jugadores, equipos, armas, mapas, competiciones y estadísticas de rendimiento. Con esta base de datos, será posible realizar consultas complejas que faciliten el análisis del rendimiento de los jugadores y equipos, conocer las armas utilizadas y observar cómo afecta la economía del juego a los resultados de las partidas.

A nivel académico, este proyecto ofrece la oportunidad de aplicar conceptos avanzados de bases de datos, como normalización, modelado relacional y la creación de consultas SQL complejas. Además, permite trabajar con conceptos más avanzados, como relaciones ternarias y la optimización de esquemas de bases de datos.

Este tipo de trabajo también proporciona una comprensión más profunda de la gestión de datos en situaciones reales, lo que es esencial para quienes buscan desarrollarse en áreas como el desarrollo de software y la administración de bases de datos.

Desde un punto de vista práctico, bases de datos como la que se va a desarrollar son clave en la industria de los videojuegos y los eSports, donde se requieren herramientas eficaces para analizar y gestionar grandes volúmenes de datos generados durante los torneos y partidas. Este sistema podría usarse en plataformas de torneos, en el análisis de jugadores y equipos, y en sistemas de emparejamiento.

Al desarrollar este proyecto, no solo he adquirido habilidades técnicas en la gestión de bases de datos, sino también experiencia en la creación de soluciones aplicables a problemas reales en el mundo de los videojuegos competitivos. Esto prepara a los estudiantes para trabajar en empresas relacionadas con los eSports, el análisis de datos y el desarrollo de software.

**2. Desarrollo.**

2.1 Texto Explicativo.

Una empresa bastante grande Valve, nos pide que le hagamos una base de datos, cuya puedan utilizar para gestionar los datos de su juego Counter-Strike 2.

De los jugadores casuales les interesa guardar id, nombre, apellido, nickname, edad, país de nacimiento, ELO, y en caso si es jugador profesional su equipo a que pertenece y cuál es su rol. Los entrenadores, que tendrán id, nombre, apellido, nickname, edad, país de nacimiento, ELO, equipo que entrenan y que es un jugador retirado. Entrenador puede entrenar un equipo único, o ninguno.

Los equipos tienen su id, nombre, país y fecha de creación. Al mismo tiempo un jugador puede estar en un equipo, pero equipo puede tener varios jugadores. Nos interesa guardar la historia de los jugadores, o sea en que equipos ha estado y que cobraba en cada uno. Un equipo puede ser entrenado por un entrenador, o por ninguno.

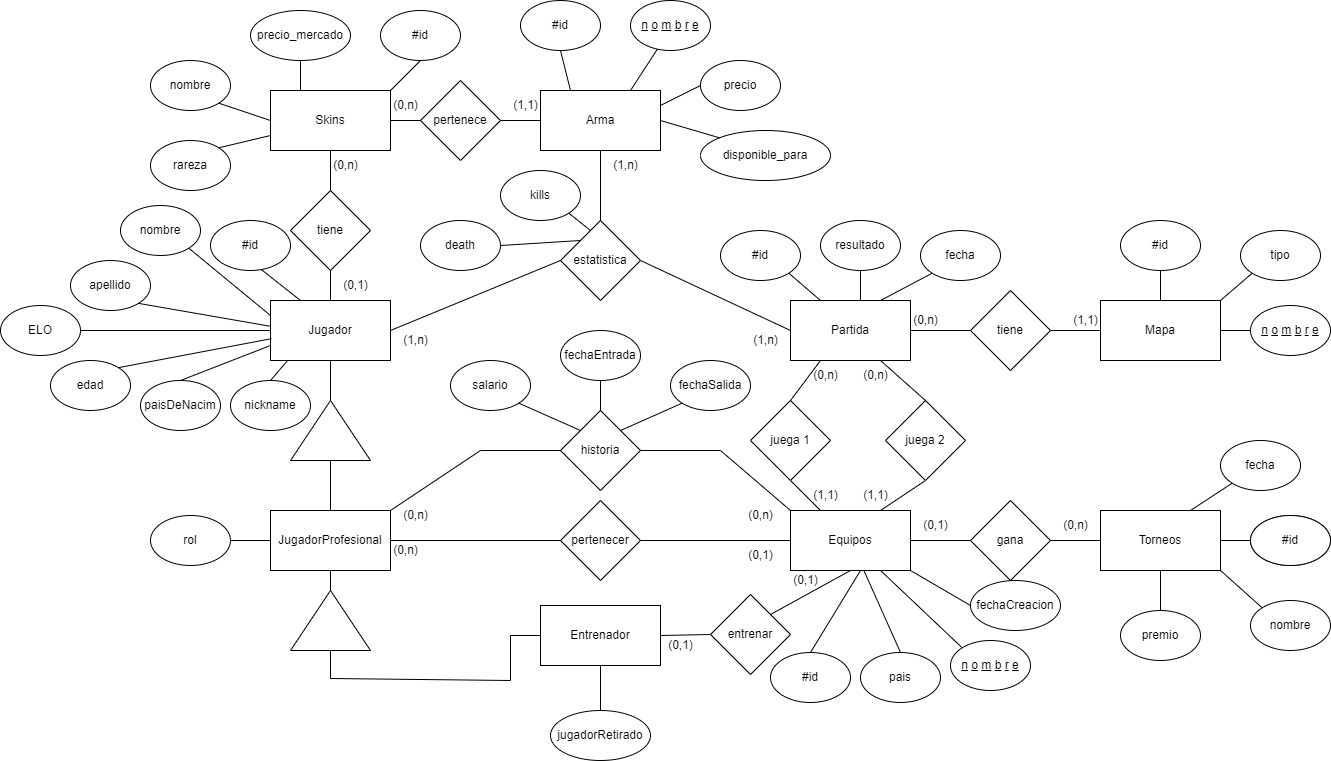
Las partidas guardan id, equipos que han participado, los resultados, fecha y mapa. En su lugar el mapa tiene su id, nombre, tipo. Una partida puede tener un mapa, y el mapa puede usarse en múltiples partidas.

Las arman tienen su id, un único nombre, precio, daño, quien la puede comprar. Cada arma tiene varios skins, de ellos tenemos que saber su id, precio en mercado y rareza. Skin solo pertenece a un arma único. Cada skin es propiedad de un jugador, pero un jugador puede tener muchos skins.

Es importante almacenar estadística de los jugadores de cada de sus partidas, se refiere a cuantas veces ha matado y ha muerto, y su arma favorita en esta partida

También organizan unos torneos y de ellos les interesa guardar su nombre, equipo que ha ganado, fecha y su premio. Un equipo puede ganar varios torneros.

2.2 Diagrama Entidad/Relacion.



2.3. Modelo relacional.

2.3.1Entidades

Jugador(id\_jugador, nickname, nombre, apellido, edad, paisNacim, ELO)

CP{id\_jugador}

JugadorProfesional(rol, id\_jugador, id\_equipo)

CP{id\_jugador}

CAj{id\_jugador} hace referencia a Jugador

CAj{id\_equipo} referencia a Equipos

Jx: JugadorProfesional, Ex:Estatistica

Ɐ Jx(JugadorProfesional(Jx) -> Ǝ Ex(Estatistica(Ex) ↑ Jx.id\_jugador = Ex.id\_jugador))

Entenador(jugadorRetirado, id\_jugador)

CP{id\_jugador}

CAj{id\_jugador} hace referencia a JugadorProfesional

Skins(id\_skin, nombre, rareza, precio\_mercado, id\_jugador, id\_arma)

CP{id\_skin}

CAj{id\_jugador} referencia a Jugador

CAj{id\_arma} referencia a Skins

UNI{id\_jugador}

UNI{id\_arma}

VNN{id\_arma}

Arma(id\_arma, nombre, precio, disponible\_para)

CP{id\_arma}

UNI{nombre}

Ax: Arma, Ex:Estatistica

Ɐ Ax(Arma(Ax) -> Ǝ Ex(Estatistica(Ex) ↑ Ax.id\_arma = Ex.id\_jugador))

Partida(id\_partida, resultado, fecha, id\_mapa, id\_equipo1, id\_equipo2)

CP{id\_partida}

CAj{id\_mapa} referencia a Mapa

CAj{id\_equipo1} referencia a Equipos

CAj{id\_equipo2} referencia a Equipos

UNI{id\_mapa}

VNN{id\_mapa}

UNI{id\_equipo1, id\_equipo2}

Px: Partida, Ex:Estatistica

Ɐ Px(Partida(Px) -> Ǝ Ex(Estatistica(Ex) ↑ Ax.id\_partida = Ex.id\_partida))

Equipos(id\_equipo, nombre, pais, fechaCreacion, id\_jugador)

CP{id\_equipo}

CAj{id\_jugador} referencia a Entenador

UNI{id\_jugador}

UNI{nombre}

Mapa(id\_mapa, nombre, tipo)

CP{id\_mapa}

UNI{nombre}

Torneos(id\_torneo, nombre, premio, fecha, id\_equipo)

CP{id\_torneo}

CAj{id\_equipo} referencia a Equipos

2.3.2 Relaciones

Historia(id\_jugador, id\_equipo, salario, fechaEntrada, fechaSalida)

CP{id\_jugador, id\_equipo}

CAj{id\_jugador} referencia a JugadorProfesional

CAj{id\_equipo} referencia a Equipos

Estatistica(id\_jugador, id\_arma, id\_partida, death, kills)

CP{id\_jugador, id\_arma, id\_partida}

CAj{id\_jugador} referencia a JugadorProfesional

CAj{id\_arma} referencia a Arma

CAj{id\_partida} referencia a Partida

VNN{id\_jugador, id\_arma, id\_partida}