Presentan:
Nicole Dávila Hernández – A01784217
Miguel Enrique Soria- A01028033
Fabrizio Barrio Blanco- A01784901

TC2005B - Construcción de software y toma de decisiones (Gpo 501)
Profesores:
Esteban Castillo Juarez
Gilberto Echeverría Furió
Octavio Navarro Hinojosa

2. Ejercicio de Modelación de Base de Datos del reto

Para el juego estilo TCG ‘Sublim’, se proponen las siguientes tablas para un modelado Entidad-Relación de UML:

- *Player*
- *Match*
- *Cards*
- *Deck*
- *Deck_Card*
- *Effect_Activation*
- *Effect*
- *Type*

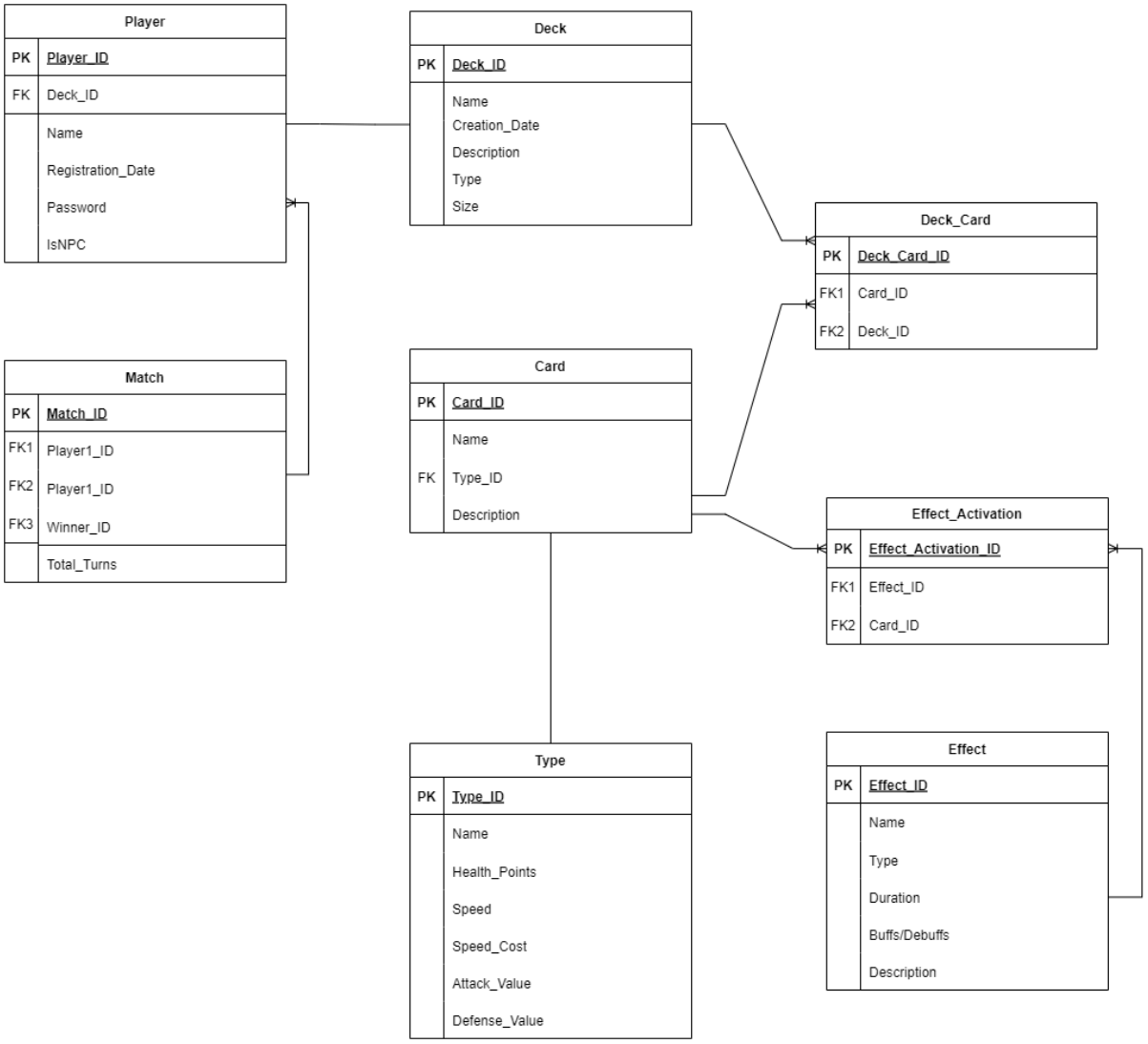

Figura 1: Diagrama del modelado ER de la base de datos.

Cabe mencionar que algunas mecánicas del juego, como lo pueden ser aquellos momentos donde se requiera tirar una moneda al azar, cómo ganar / perder el juego, etcétera, se podrían programar directamente y sin necesidad de asignarles una tabla.

Tomando esto en cuenta, comenzamos con la tabla *Player*.

Esta tabla contiene los siguientes atributos, donde *'PK'* denota que esta es la restricción de llave primaria de la tabla:

- *Player_ID (PK)*
- *Name*
- *Registration_Date*
- *Password*
- *IsNPC*

Esta tabla incluye atributos clave para identificar correctamente a los jugadores, así como si el jugador es un Non-Playable Character o el jugador como tal. El *Player_ID* será único y exclusivo para cada jugador. La información del jugador también podría ser útil para crear un leaderboard con los puntajes más altos.

La siguiente tabla que tenemos es *Match*.

Esta tabla principalmente funciona para denominar los jugadores, el juego en el que se encuentran y llevar cuenta del turno de cada jugador. Su llave primaria es un ID del juego y sus llaves secundarias se componen de:

- *user1_id* – en la cual se encontrará un *user_id* de quien se haya puesto como jugador 1.
- *user2_id* – en la cual se encontrará el *user_id* de quien se haya puesto como jugador 2.
- *winner_id* – representa el jugador que ganó la partida.
- *total_turns* – representa la cantidad de turnos que duró la partida.

Sublim no contará con una banca, por lo que esta tabla no existe en nuestra propuesta de diagrama UML. En vez de esto, consideramos más eficiente que Unity se encargue de la actividad o inactividad de las cartas. Tocando el tema de las cartas, la tabla *Cards* contiene lo siguiente:

- *Card_ID (PK)*
- *Name (FK)*
- *Type_ID (FK)*
- *Description (FK)*

Sin embargo, los aspectos más importantes de las cartas entran en las tablas de *effect_activation*, *type* y *effect*.

effect_activation es una tabla intermedia para conectar la relación muchos a muchos entre *card* y *effect*. Incluye únicamente su llave primaria y las llaves foráneas a las tablas anteriormente mencionadas.

effect representa los distintos efectos de las cartas objeto del juego, como lo son las cartas de aumento de daño, defensa o penetración. Esta tabla también describe los efectos de las cartas robo.

deck_card es una tabla intermedia que conecta la relación muchos a muchos de las entidades *deck* y *card*, teniendo de manera similar a *effect_activation* su llave primaria y las dos llaves foráneas de las tablas mencionadas.

type es una tabla derivada de card para representar con menos duplicados el tipo de carta del que se está hablando, al igual que sus atributos. En esta tabla varios de los valores como *speed, speed cost, health, attack_value, defense_value,* etcétera, pueden ser nulos.

Deck es la tabla que dará las características que tendrá el deck de cada jugador. Tendremos como (*PK*) el *Deck_Id*, el nombre que se asignará a cada mazo, datos de creación, descripción del mazo y el tipo de mazo que va a depender de la estrategia de cada jugador.

Por supuesto, estas cartas pueden pertenecer a un mazo, por lo que tenemos la tabla *deck*, la cual incluye el ID del mazo como llave principal, el ID del usuario como llave foránea y el nombre del mazo. Esta tabla entonces se relaciona con *Cards_Deck*, y define las cartas que contiene el mazo así como la cantidad de cartas que existen en este.