Proyecto: Base de Datos TCG

En esta oportunidad planteamos al/la estudiante la creación de un pequeño modelo de bases de datos basado en la idea de un **TCG** (*Trading Card Game* o *Juego de Cartas Coleccionables* <JCC>). Este tipo de juegos de mesa, a diferencia de los juegos tradicionales con cartas inglesas o españolas, donde las reglas de cada juego definen para qué sirve cada carta y cómo se usa cada una, en un JCC las reglas del juego definen cómo funciona una partida pero luego cada carta particular tiene su propio uso particular, efectos y reglas, las cuales suelen estar escritas en dicha carta. Así, un JCC tiene cientos o miles de cartas disponibles con las cuales cada jugador arma su mazo (deck) con estrategias particulares, siendo, por tanto, cada partida prácticamente única. Los dos ejemplos, quizá, más conocidos son ***Yu Gi Oh!*** y ***Pokemon JCC***, aunque existen centenares de juegos con cartas coleccionables.

Este proyecto propondrá la creación de una base de datos SQL en SQLite y MySQL tomando como idea la existencia de los JCC, donde se proveerá de una estructura de persistencia para dar soporte a un supuesto juego de cartas coleccionables. Las reglas de dicho juego no nos interesan en absoluto, ya que nuestra labor es proveer de la estructura de datos que se solicitará en la descripción de este documento y nada más. La lógica del juego mismo debería ser programada en un lenguaje de programación que se encargaría de llevar a cabo todas las reglas del juego, eventos de los jugadores, etc., aplicación que guardaría todos sus datos en la base de datos que se propone crear aquí.

Objetivo del proyecto

El objetivo principal es que el/la estudiante aplique todo lo visto en el curso hasta el momento para crear su propia base de datos SQL siguiendo la solicitud que se describirá aquí, teniendo que repasar temas y asentar las bases que este curso pretende desarrollar en los/as estudiantes como capacidades. De este modo, se brindará aquí la descripción de un problema y el/la estudiante deberá modelar su propia base de datos para dar solución a dicho problema.

Descripción del problema

Supongamos que se va a desarrollar un videojuego del estilo JCC, en el cual existe un enorme listado de cartas disponibles para los jugadores, siendo factible que cada cierto tiempo se generen actualizaciones al juego agregando nuevas cartas que los jugadores podrían ganar y desbloquear dentro del juego.

Tipos de cartas

Dentro del juego, las cartas pueden ser de distinto tipo. En el momento de la creación de este documento existen 4 tipos posibles para una carta: **Magia**, **Herramienta**, **Arma** y **Guerrero**, pero podrían agregarse más. Los nombres de los tipos son únicos e irrepetibles.

Las cartas

Además del **tipo**, cada carta posee los siguientes atributos que deben poder guardarse:

- **Nombre**: un nombre único que identifica a la carta.
- **Número de serie**: un número único de 5 cifras que identifica a la carta.
- **Arte**: un dibujo ilustrado de la carta.
- **Descripción**: un texto descriptivo que explica lo que la carta hace dentro del juego.

A modo ilustrativo de muestra a continuación una carta del juego **Yu Gi Oh!**:

Esta carta contiene en su parte inferior un número identificatorio único, que en este caso es **26202165**. Su nombre es **Sangan**, tiene una ilustración representativa y sobre la esquina superior derecha un dibujo indica que el tipo es **Oscuridad**. También podemos leer, entre otras cosas, una descripción de lo que hace esta carta al ser usada en juego:

"Cuando esta carta se manda del Campo al Cementerio, selecciona 1 monstruo con ATK 1500 o menos de tu Deck, muéstralo a tu adversario y añádelo a tu mano. Luego baraja tu Deck."

¿Qué es Campo? ¿Qué es cementerio? ¿Qué son los atributos ***Demonio***, ***Efecto***, ***ATK*** o ***DEF***? No nos importa, son cuestiones que hacen parte de las reglas específicas del juego **Yu Gi Oh**. A los efectos de este proyecto, como diseñadores/as de la base de datos del juego que se nos describe, hemos de limitarnos a resolver lo que se nos pide y nada más. Veamos un ejemplo diferente usando una carta de ***Pokemon JCC***:

En este caso se tienen muchos atributos, pero a grandes rasgos y a los efectos del supuesto juego que estamos creando, la carta tiene un tipo, que en este caso es **Básico**, un nombre que es **Castform**, una ilustración, un número de serie que es **121/198** y una descripción (en realidad dos) que indica:

"Si tienes 8 cartas de Estado o más en tu pila de descartes, ignora todas las Energías en el coste de los ataques de este Pokémon."

Con estos dos ejemplos puedes hacerte una idea mejor de lo que se está modelando. Incluso, sin tú tener por qué saber algo sobre cómo se juega a cualquiera de estos dos juegos, con un detalle completo de la información de cada carta deberías poder modelar una base de datos, todo lo demás es problema de los programadores. En nuestro caso hipotético desconocemos totalmente las reglas del juego para el cual estamos diseñando la base de datos (de hecho no existe tal juego), pero debemos ser capaces de generar la estructura que permita dar solución a lo que se describe. Por tanto, con lo que tienes hasta ahora debes poder guardar los tipos de cartas y las cartas con los atributos mencionados más arriba. Nuestro juego nada tiene que ver con ***Yu Gi Oh!*** o ***Pokemon JCC***, simplemente se mostraron estos ejemplos para ayudarte a entrar en contexto.

Deck o mazo

Un ***deck*** simplemente es un listado de cartas conteniendo, para cada carta que lo conforma, la cantidad de la misma. Se ha de controlar que un *deck* no tenga menos de 50 cartas ni más de 60

en total. También se debe verificar que al menos 5 de las cartas sean de tipo **Guerrero**. Cada deck además estará identificado por un nombre único en la lista de decks de cada jugador.

El jugador

Representa a la persona que hace uso del juego, quién estará representada por un alias (nombre de usuario), un correo electrónico y una contraseña de 8 dígitos de largo como mínimo y 16 como máximo. La base de datos no hará un control de la cantidad mínima de caracteres ingresados, sino que limitará el máximo a 16, lo demás será gestionado por la interfaz gráfica del software del juego, por lo que no ha de preocuparnos.

Además de los datos básicos de identificación, un jugador tendrá en su cuenta la siguiente información:

- **Colección**: la lista completa de cartas en su poder. Esta lista contendrá, en cada registro, la carta en cuestión y la cantidad de la misma, ya que un jugador puede tener muchas copias de una misma carta.
- **Decks**: los jugadores pueden tener infinitos ***decks***, los cuales serán conformados con las cartas que tienen en su colección. La base de datos debe permitir guardar los decks de que los jugadores van creando.
- **Amigos**: cada jugador puede contener una lista de otros jugadores a los que llamará amigos y con los cuales podrá tener interacciones especiales. La base de datos debe permitir guardar dicho listado, luego será el software quién se encargue de dar funcionalidad a dicha lista.

Se pide

Crear una base de datos que modele el problema descrito anteriormente, tanto en **MySQL** como en **SQLite** de forma tal que:

- Se pueda tener una lista de cartas existentes en el juego así como una lista con los tipos disponibles. En la sección siguiente se describirá la información que se brindará al/la estudiante para que sea importada a la base de datos.
- Debe ser posible agregar tipos nuevos de cartas, tantos como se desee.
- El arte de las cartas (dibujo) no será de tipo BLOB, sino que será la ruta del archivo de imagen (extensión **PNG**). Esta ruta será, para todos los casos **/IMG/{SERIE}.png**. En este formato `{SERIE}` indica el número de serie de la carta en cuestión, por ejemplo, si dicho número es 45674 la ruta será **/IMG/45674.png**.
- Será posible eliminar de la base de datos tipos de cartas ya registrados siempre y cuando no exista vinculada ninguna carta a él. **El tipo Guerrero no puede ser eliminado**. Para lograr este comportamiento controlado sobre la eliminación, se debe registrar un disparador en la tabla de tipos sobre la sentencia DELETE de forma tal que:
  - Se verificará previamente si hay cartas asociadas al tipo que se quiere eliminar.
  - Si no hay cartas asociadas a él se podrá proceder con normalidad.
  - Si hay cartas asociadas a él se cancelará la eliminación para que no haya cambios:
    - Esto último requiere un poco de investigación: ¿cómo se aborta la eliminación? ¿Qué mecanismos provee MySQL y SQLite para esto?
- Se debe poder registrar jugadores/as nuevos/as. Cuando un/a jugador/a nuevo/a es registrado/a se generará para él/ella un **deck básico** cuya información será descrita en la sección siguiente. Esto se hará de forma diferente en MySQL que en SQLite:

    - **MySQL**: se creará un procedimiento almacenado llamado **registrarJugador**, el cual recibirá parámetros con los datos necesarios para crear un/a nuevo jugador/a desde cero. Se asumirá que el/la jugador/a a registrar no existe ya en la base de datos, por lo cual no hay que controlar esta situación. Este procedimiento deberá crear el registro del/la jugador/a y crear para él/ella un nuevo **deck** básico con los datos que se describirán en la sección siguiente.
    - **SQLite**: como no existen procedimientos almacenados se creará un disparador sobre la tabla de jugadores el cual se ejecutará luego de la inserción de un nuevo registro (no controlaremos la existencia previa del jugador). Este disparador se encargará de registrar el deck básico para el/la jugador/a.
- Debe ser posible que los/as jugadores/as tengan diferentes decks en su cuenta. Este registro se hará desde el software que usará la base de datos por lo cual con solo proveer la estructura es suficiente. Esto lo necesitarás, de hecho, para poder realizar el punto anterior al registrar un/a jugador/a con un deck básico.
- Se deben poder vincular y desvincular amigos.

Script de creación

Se deberá crear un *Script de Creación* de la base de datos desde cero el cual generará toda su estructura cumpliendo con todo lo solicitado anteriormente, y además insertará la información predeterminada en las tablas correspondientes:

- La lista inicial de tipos disponibles deberá estar incluida en dicho script.
- La lista inicial de cartas disponibles deberá estar incluida en dicho script.
- La información del mazo básico por defecto que será generado para cada jugador/a nuevo/a que se registra también deberá estar en dicho Script. El procedimiento MySQL y el disparador SQLite se basarán en esta información para ejecutarse.
- El código de creación también deberá encargarse de generar los disparadores y procedimientos almacenados.

De más está decir que se debe brindar un *script de creación* para **MySQL** y otro para **SQLite**.

Datos a importar

La información a importar será brindada al/la estudiante en archivos CSV, sin embargo será necesario que dicha información sea revisada y ajustada a fin de que pueda ser importada a la base de datos correctamente según la estructura que el/la estudiante decida implementar.

Tipos

Ya se mencionó anteriormente que los tipos existentes de forma predeterminada en la base de datos serán: **Magia**, **Herramienta**, **Arma** y **Guerrero**.

Lista de cartas existentes

Se brindará una lista de cartas existentes para precargar en la base de datos. Esta lista, al igual que los tipos, deberá ser importada para generar posteriormente el *Script de Creación* de la base de datos de forma tal que éste incluya estos registros en las tablas correspondientes.

Deck básico inicial

También se brindará la lista de cartas y la cantidad de cada una que será incluida en el deck básico inicial del/la jugador/a. Este deck tendrá por nombre ***Deck básico inicial***. El/La estudiante deberá procesar la información a fin de lograr que pueda ser importada a su base de datos tal como la haya implementado.

Lo que se debe entregar

Enviarás un correo electrónico a bedelia@kaedusoft.edu.uy indicando en el asunto tu **nombre**, **apellido**, seguido del texto **PROYECTO SQL TSG**. Por ejemplo, si el docente enviara su propio proyecto, el asunto del mail sería **Vladimir Rodríguez PROYECTO SQL TSG**.

Dicho correo deberá tener adjuntos dos archivos:

- Script de creación MySQL
- Script de creación SQLite

Ambos archivos deberán tener absolutamente todo lo necesario para que la base de datos sea generada con una estructura correcta, además de contener los códigos para añadir los disparadores y procedimientos almacenados necesarios.

Restricciones

En tanto se cumpla con todo lo anterior, no hay restricciones a lo que se puede usar, estando disponible todo lo que se ha visto en el curso y todo lo que el/la estudiante investigue por su cuenta si así lo desea.