Universidad Rey Juan Carlos

Escuela Técnica Superior Ingeniería Informática

Práctica 2 - XML

Gestión de Datos en Medios Digitales

Grupo 1

Blanca García Vera

Luis Antonio González Martínez

Mario López García

Juan Alessandro Vázquez Bustos

Alejandro Vargas Lugo

Índice

Introducción

Esta práctica sirve como continuación a la práctica realizada anteriormente, que consistía en la recreación de un sistema de bases de datos de un juego ya creado anteriormente. El juego seleccionado fue ***Paper Mario: La Puerta Milenaria*** (Nintendo, 2004), y se utilizó el sistema SQLite para realizar la entrada de datos y el cribado de tablas.

En esta ocasión la tarea es transformar la base de datos relacional definida en dicha práctica en un conjunto de ficheros XML bien construidos y validados a través del XSD, de acuerdo con la semántica definida en la misma.

Análisis

Al igual que en la primera práctica, el análisis gira en torno al sistema de combate donde se encuentran el personaje, las sinergias, ventajas y desventajas con los personajes y otros elementos del juego. Para representar esto, utilizamos clases como **"Jugador"**, donde almacenamos datos como los puntos de vida y los personajes aliados. Además, tenemos otras clases como **"Personaje", "Medalla", "Habilidad Estrella" e "Ítems"**. Estas clases nos permiten organizar y gestionar la información de manera eficiente.

Para cada tabla, se muestra una muestra del código en XML correspondiente a la misma:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Usuarios>
    <Usuario id="1">
        <Nombre>Alberto</Nombre>
    </Usuario>
    <Usuario id="2">
        <Nombre>Alejandro</Nombre>
    </Usuario>
```

Tabla Usuario

Esta tabla se puede utilizar para diferenciar a los usuarios que abren partidas en el juego, dado que cada jugador puede tener varias partidas asociadas, pero una partida solo puede haberla creado a un jugador.

La tabla "Usuario" tiene una primary key "ID_Usuario" y un atributo "Nombre" con el nombre de cada usuario.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Partidas>
    <Partida User_id="1" Jug_id="1">
        <Nombre>Alberto</Nombre>
        <Horas>19</Horas>
    </Partida>
    <Partida User_id="2" Jug_id="2">
        <Nombre>Alejandro</Nombre>
        <Horas>27</Horas>
    </Partida>
```

Tabla Partida

Cada partida tiene asociado a un jugador, que es el que la ha creado, la forma de que un usuario cree varias partidas es el identificador de jugador, si un mismo usuario genera múltiples partidas, todas ellas tendrán el mismo id de usuario, pero un id de jugador distinto.

La tabla "Partida" tiene una primary key "Nombre" (en este caso no utilizamos el ID para identificarla sino el nombre que pone el usuario al crear la partida), "Horas_jugadas" (que contiene el tiempo de juego), y 2 claves ajenas "Usuario_ID" y "Jugador_ID" que conforman la relación N:M entre ambas entidades.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Jugadores>
    <Jugador id="1" Personaje="Bombard">
        <PC>32</PC>
        <PE>0</PE>
        <PF>21</PF>
        <Monedas>104</Monedas>
        <Estado>4</Estado>
    </Jugador>
    <Jugador id="2" Personaje="Koops">
        <PC>34</PC>
        <PE>4</PE>
        <PF>30</PF>
        <Monedas>31</Monedas>
        <Estado>1</Estado>
    </Jugador>
```

Tabla Jugador

Esta tabla define al jugador controlable por el usuario (Mario). Contiene toda la información de sus estadísticas tales como salud ataque, habilidades disponibles, monedas…

La tabla "Jugador" se identifica por la primary key "Jugador_ID". Tiene 3 atributos que son diferentes tipos de puntos propios del jugador: "Puntos_corazon", "Puntos_estrella", y "Puntos_flor". El jugador también posee "Monedas" para contabilizar el dinero, y "Estado" para representar diferentes circunstancias como, por ejemplo, si está paralizado. Por último, la clave ajena "Personaje" para conformar la relación 1:N entre dichas entidades.

```xml
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<items>
    <item>
        <nombre>Seta</nombre>
        <descripcion>Recupera 5 PC.</descripcion>
    </item>
    <item>
        <nombre>Gelatina</nombre>
        <descripcion>Recupera 5 PF.</descripcion>
    </item>
    <item>
        <nombre>Ultra Seta</nombre>
        <descripcion>Recupera 15 PC.</descripcion>
    </item>
    <item>
        <nombre>Grandimiel</nombre>
        <descripcion>Recupera 15 PF.</descripcion>
    </item>
```

Tabla Ítems

Sirve para definir las características de cada ítem, identificado por su nombre en la tabla Jugador_items, de forma que cada ítem que posee el jugador cuenta con una descripción de su funcionamiento. Cada ítem (identificados por su nombre) que posee el jugador cuenta con una descripción específica de ese ítem, por eso es necesaria la creación de esta tabla.

La tabla "Items" consiste en únicamente dos atributos, el nombre del ítem y la descripción, que explica los efectos del ítem. Ambos son de tipo texto.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Jugadores_Items>
    <Jugador_Items Jugador_ID="1">
        <items_Nombre>Seta</items_Nombre>
        <num_Items>100</num_Items>
    </Jugador_Items>
    <Jugador_Items Jugador_ID="2">
        <items_Nombre>Espíritu</items_Nombre>
        <num_Items>59</num_Items>
    </Jugador_Items>
    <Jugador_Items Jugador_ID="3">
        <items_Nombre>Tónico</items_Nombre>
        <num_Items>81</num_Items>
    </Jugador_Items>
```

Tabla Jugador_items

Se utiliza esta tabla para que un jugador pueda poseer varios ítems distintos. Cada uno de ellos se distingue por su nombre (que es su PK) y contabiliza en un integer, todos los objetos que tengan el mismo nombre. Y por tanto las mismas propiedades por lo que si Mario consigue uno no es necesario crear una tabla de nuevo, sino añadir una unidad al objeto ya existente.

La tabla "Jugador_items" se identifica por la primary key de un jugador "Jugador_ID" y por el nombre del item que este posee "items_nombre" , guardando así para cada par de estos el número de ítems correspondiente a un jugador y un ítem concreto en "num_Items"

```xml
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<medallas>
    <medalla>
        <nombre>Salto Potente</nombre>
        <costePM>1</costePM>
        <descripcion>Permite realizar un Salto Potente. Gasta 2 PF.
Salta con mucho poder de ataque.</descripcion>
    </medalla>
    <medalla>
        <nombre>Multi-Rebote</nombre>
        <costePM>1</costePM>
        <descripcion>Permite realizar un Multi-Rebote. Gasta 2 PF.
Puedes saltar sobre todos los enemigos.</descripcion>
    </medalla>
    <medalla>
        <nombre>Golpe Potente</nombre>
        <costePM>1</costePM>
        <descripcion>Permite realizar un Golpe Potente. Gasta 2 PF.
Martillea con mucho poder de ataque.</descripcion>
    </medalla>
    <medalla>
```

Tabla Medalla

Sirve para identificar las medallas que puede tener el jugador, estas se identifican por su nombre, y pueden pertenecer a un solo jugador. De ahí que se utilice una FK que la asocia con el ID de un jugador concreto.

Es necesaria su creación porque estas cuentan con características propias como una descripción, atributos y un coste

La tabla "Medalla" tiene los siguientes atributos: "Nombre" es la primary key de esta tabla, usada para identificar cada elemento. Cada medalla tiene un coste de puntos medalla, este dato aparece reflejado en el atributo "Coste_PM". Cada medalla tiene asimismo una descripción detallada en "Descripcion". Con el objetivo de reflejar la relación 1:N entre "Medalla" y "Jugador" se usa la clave ajena "Jugador_ID".

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Habs_Estrella>
    <Hab_Estrella Jugador_ID="6">
        <Nombre>Recuperar</Nombre>
        <Coste_PM>1</Coste_PM>
        <Descripcion>Recupera PC y PF, curando a Mario y a su compañero.</Descripcion>
    </Hab_Estrella>
    <Hab_Estrella Jugador_ID="7">
        <Nombre>Terremoto</Nombre>
        <Coste_PM>2</Coste_PM>
        <Descripcion>Ataca a todos los enemigos haciendo temblar el suelo.</Descripcion>
    </Hab_Estrella>
    <Hab_Estrella Jugador_ID="4">
        <Nombre>Tiempo Muerto</Nombre>
        <Coste_PM>2</Coste_PM>
        <Descripcion>Detiene durante un rato a todos los enemigos.</Descripcion>
    </Hab_Estrella>
```

Tabla Hab_Estrella

Sirve para identificar las habilidades que puede tener el jugador, estas se identifican por su nombre, y pueden pertenecer a un solo jugador. De ahí que se utilice una FK que la asocia con el ID de un jugador concreto.

Es necesaria su creación porque estas habilidades cuentan con características propias como una descripción y un coste.

Esta tabla contiene como atributos un "Nombre", que permite identificar unívocamente la habilidad, siendo la Primary Key; un "Coste_PE", que indica el coste de la habilidad, lo que permite elegir la que más convenga según los puntos de los que disponga el jugador; una "Descripcion", que permite saber qué efecto tiene la habilidad; y el "Jugador_ID", que contiene el ID del jugador que posee la habilidad.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Personajes>
    <Personaje>
        <Nombre>Goomarina</Nombre>
        <PC>25</PC>
        <PF>15</PF>
        <Procedencia>Villa Viciosa</Procedencia>
        <Descripcion>Una Goomba universitaria que aspira a ser arqueóloga.</Descripcion>
    </Personaje>
    <Personaje>
        <Nombre>Koops</Nombre>
        <PC>15</PC>
        <PF>20</PF>
        <Procedencia>Villa Verde</Procedencia>
        <Descripcion>Un joven Koopa que desea volverse más fuerte.</Descripcion>
    </Personaje>
    <Personaje>
        <Nombre>Claudia</Nombre>
        <PC>35</PC>
        <PF>30</PF>
        <Procedencia>Bosque Maravilloso</Procedencia>
        <Descripcion>¡Un espíritu nebuloso con un cuerpo que es pura dinamita!</Descripcion>
    </Personaje>
```

Tabla Personaje

Hace referencia al compañero del jugador (Mario). Mario cuenta con un acompañante que cuenta con unas estadísticas propias, tales como vida, puntos de fuerza y su procedencia. Este personaje se asocia a la tabla lugar, que sirve para especificar que un este puede provenir de un solo lugar, mientras que de un lugar pueden provenir varios acompañantes.

Esta tabla se identifica por la primary key, que es el nombre del personaje "Nombre", contiene una foreign key que lo enlaza directamente el lugar del que procede, llamada "Procedencia". Contiene otros atributos como "PF": integer con los puntos de fuerza, "PC" integer con los puntos corazón, y "Descripción", con un string breve que indica aspectos a destacar del personaje.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Habilidades>
    <Habilidad Personaje_nom="Goomarina">
        <Nombre>Cabezazo</Nombre>
        <Coste_PF>0</Coste_PF>
        <Ind_atq>4</Ind_atq>
        <Ind_def>0</Ind_def>
        <Descripcion>Ataca a un enemigo de un cabezazo.</Descripcion>
    </Habilidad>
    <Habilidad Personaje_nom="Goomarina">
        <Nombre>Descripción</Nombre>
        <Coste_PF>0</Coste_PF>
        <Ind_atq>0</Ind_atq>
        <Ind_def>0</Ind_def>
        <Descripcion>Muestra la descripción del enemigo y sus PC.</Descripcion>
    </Habilidad>
```

Tabla Habilidad

Cada personaje acompañante de Mario puede tener varias habilidades. Esta tabla sirve para asociar a cada acompañante de Mario, cada una de las habilidades con las que cuenta. Para ello, cada habilidad se identifica por su nombre.

La entidad "Habilidad" tiene una primary key "Nombre". Para adquirir una habilidad es necesario gastar puntos flor, este coste aparece reflejado en el atributo "Coste_PF". "Índice_ataque" e "Índice_defensa" son el daño y protección asociados a la habilidad. A su vez, cada habilidad tiene un atributo "Descripcion" con una explicación de qué acción realiza. "Personaje_Nombre" sirve para constatar la relación 1:N entre "Personaje" y "Habilidad".

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Lugares>
    <Lugar>
        <Nombre>Villa Viciosa</Nombre>
        <Npc>15</Npc>
        <Enemigos>14</Enemigos>
    </Lugar>
    <Lugar>
        <Nombre>Subsuelo de Villa Viciosa</Nombre>
        <Npc>13</Npc>
        <Enemigos>8</Enemigos>
    </Lugar>
```

Tabla Lugar

Sirve para indicar el lugar en el que se encuentran enemigos o acompañantes de Mario. Gracias a la existencia de esta tabla, podemos clarificar que un lugar puede contener varios tipos de acompañantes que procedan del mismo, y varios tipos de enemigos que se encuentren en él.

La entidad "Lugar" se identifica por una primary key "Nombre" y cuenta con dos atributos "NPCs" que indica el numero de NPCs del lugar (personajes interactuables pero que no batallan) y "Enemigo" que se refiere al numero de enemigos que se encuentran en el lugar.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Lugar_Enemigos>
    <Lugar_Enemigo Nombre_Lugar="Templo Lúgubre" Enemigo_ID="31" />
    <Lugar_Enemigo Nombre_Lugar="Llanura Estelar" Enemigo_ID="10" />
    <Lugar_Enemigo Nombre_Lugar="Isla Trópico" Enemigo_ID="52" />
    <Lugar_Enemigo Nombre_Lugar="Fortaleza de Piedra" Enemigo_ID="50" />
    <Lugar_Enemigo Nombre_Lugar="Templo Lúgubre" Enemigo_ID="64" />
    <Lugar_Enemigo Nombre_Lugar="Bomburgo" Enemigo_ID="69" />
    <Lugar_Enemigo Nombre_Lugar="Bomburgo" Enemigo_ID="58" />
    <Lugar_Enemigo Nombre_Lugar="Villa Viciosa" Enemigo_ID="42" />
    <Lugar_Enemigo Nombre_Lugar="Cien Mazmorras" Enemigo_ID="30" />
    <Lugar_Enemigo Nombre_Lugar="Subsuelo de Villa Viciosa" Enemigo_ID="18" />
    <Lugar_Enemigo Nombre_Lugar="Llanura Estelar" Enemigo_ID="2" />
    <Lugar_Enemigo Nombre_Lugar="Castillo de Goombaba" Enemigo_ID="20" />
    <Lugar_Enemigo Nombre_Lugar="Camino Sombrío" Enemigo_ID="61" />
    <Lugar_Enemigo Nombre_Lugar="Ciudad Dojo" Enemigo_ID="38" />
    <Lugar_Enemigo Nombre_Lugar="Gran Árbol" Enemigo_ID="62" />
```

Tabla Lugar_Enemigo

Esta tabla existe meramente para especificar el tipo de conexión que tienen los enemigos con los lugares. Ya que en el caso de los enemigos, al contrario que con el lugar de procedencia de los acompañantes de Mario, que solo puede ser uno, estos se pueden encontrar en varios

lugares (ej. tanto en “Llanura Estelar” como “Cien Mazmorras” pueden aparecer goombas)

Esta tabla intermedia contiene como atributos: “Nombre_Lugar”, que almacena el nombre de las distintas áreas del juego y “Enemigo_ID”, que contiene el ID de los enemigos que se encuentran en cada una de las zonas. Estos dos atributos sirven juntos como Primary key.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Enemigos>
  <Enemigo Enemigo_ID="1" Vuela="0" Tiene_pincho="0" Tiene_caparazon="0" Es_Boss="0">
    <Nombre>Goomba</Nombre>
    <Region>Diversos lugares</Region>
    <Puntos_ataque>1</Puntos_ataque>
    <Puntos_defensa>0</Puntos_defensa>
    <Puntos_corazon>2</Puntos_corazon>
  </Enemigo>
  <Enemigo Enemigo_ID="2" Vuela="1" Tiene_pincho="0" Tiene_caparazon="0" Es_Boss="0">
    <Nombre>Paragoomba</Nombre>
    <Region>Diversos lugares</Region>
    <Puntos_ataque>1</Puntos_ataque>
    <Puntos_defensa>0</Puntos_defensa>
    <Puntos_corazon>2</Puntos_corazon>
  </Enemigo>
  <Enemigo Enemigo_ID="3" Vuela="0" Tiene_pincho="1" Tiene_caparazon="0" Es_Boss="0">
    <Nombre>Goompincho</Nombre>
    <Region>Diversos lugares</Region>
    <Puntos_ataque>2</Puntos_ataque>
    <Puntos_defensa>0</Puntos_defensa>
    <Puntos_corazon>2</Puntos_corazon>
  </Enemigo>
```

Tabla Enemigo

Define todas las estadísticas de cada enemigo, y asocia un id para cada uno.

Esta entidad tiene un total de 10 atributos. La tabla define todos los parámetros de cada enemigo, que incluyen los puntos corazón (vida) (“Puntos_corazon”), puntos ataque (“Puntos_ataque”) y defensa (“Puntos_defensa”), y cuatro booleanos para determinar si el enemigo (1) es capaz de volar (“Vuela”) (no se puede atacar en suelo), (2) posee la característica de que tiene pinchos en su cuerpo (“Tiene_pincho”) (hace daño al saltar sobre él), (3) tiene un caparazón sobre la piel (“Tiene_caparazon”) (inmune a ciertos ataques) o (4) es boss (jefe) (“Es_Boss”); además de un ID (“Enemigo_ID”) que los identifica (su primary key).

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Batallas>
  <Batalla Enemigo_ID="59" Jugador_ID="1">
    <num_Turnos>3</num_Turnos>
    <num_Ayudantes>0</num_Ayudantes>
  </Batalla>
  <Batalla Enemigo_ID="12" Jugador_ID="2">
    <num_Turnos>2</num_Turnos>
    <num_Ayudantes>3</num_Ayudantes>
  </Batalla>
  <Batalla Enemigo_ID="32" Jugador_ID="3">
    <num_Turnos>10</num_Turnos>
    <num_Ayudantes>0</num_Ayudantes>
  </Batalla>
  <Batalla Enemigo_ID="2" Jugador_ID="4">
    <num_Turnos>9</num_Turnos>
    <num_Ayudantes>2</num_Ayudantes>
  </Batalla>
```

Tabla Batalla

Esta tabla se genera para cada enfrentamiento entre jugador (Mario) y un enemigo. Dicho enemigo puede venir acompañado por otros, pero la batalla se define por Mario y el enemigo “principal” con el que ha iniciado batalla.

La creación de esta tabla es necesaria para especificar que tanto el jugador como enemigos pueden competir en múltiples batallas (relación N a N)

Cada tabla se identifica por el id del jugador y el id del enemigo principal. No obstante, el enemigo se verá acompañado por más enemigos de características similares especificados en el nº de ayudantes.

Se guarda también el nº de turnos que va durando la batalla.

Esta entidad se identifica por dos primary keys, que son “Enemigo_ID” y “Jugador_ID”. De esta forma cada batalla se identifica por los dos involucrados en esta, el id de enemigo y el de un jugador.

Adicionalmente contiene dos atributos para manejar el numero de turnos: “núm Turns” y otro que indica la ayuda que el jugador puede solicitar durante el combate.

Relaciones

Las relaciones establecidas son las necesarias para que la base de datos funcione correctamente.

Relación 1, Jugador - Batalla - Enemigo (N-N):

La relación existente entre "Jugador" y "Enemigo" se hace a través de la tabla intermedia "Batalla", lo que nos permite tener toda la información acerca de las batallas en el juego: qué jugador y enemigo participan, el número de turnos y el número de ayudantes que tiene el enemigo. Sería la relación en la que se sustenta la información relacionada con el sistema de combate.

Un jugador puede pelear contra varios enemigos y un enemigo puede pelear contra varios jugadores, al ser la relación de muchos a muchos, se debe usar una relación externa que sea capaz de soportar estos requisitos:

Un jugador puede participar en varias batallas pero en una batalla puede participar un solo jugador.

En la batalla puede participar un solo enemigo pero un tipo de enemigo puede pelear en varias batallas.

Relación 2, Enemigo - Lugar_Enemigo - Lugar (N-N):

Un enemigo puede aparecer en varios lugares y en cada lugar pueden aparecer varios enemigos. Esto requiere de una entidad externa (entidad Lugar_Enemigo como "zona").

Un enemigo puede aparecer en varias zonas y una zona puede tener varios enemigos.

Un lugar puede tener varias zonas pero una zona solo puede estar en un solo lugar.

Relación 3, Lugar - Personaje (1-N):

Luego está la relación existente entre "Jugador" y "Personaje", que permite almacenar todos los personajes que acompañan al jugador en su aventura; además de la relación que asocia "Personaje" y "Lugar", que permite saber la ubicación del personaje y, por tanto, donde lo puede obtener el jugador como acompañante.

Un lugar puede tener varios personajes y un personaje solo está en un lugar.

Relación 4, Personaje - Habilidad (1-N):

Un personaje puede tener varias habilidades pero una habilidad solo puede pertenecer a un personaje.

Relación 5, Personaje - Jugador (1-N):

Un personaje puede ser el personaje activo de un jugador, pero el jugador Pero el jugador solo puede tener un personaje activo a la vez.

Por último, cabe destacar las relaciones entre "Jugador" y las tablas "Hab_Estrella" y "Medalla", que permiten conocer las habilidades y las medallas obtenidas por el jugador, con su descripción y su coste, que ayudarán al jugador en las batallas.

Relación 6, Medalla - Jugador (1-N):

Un jugador puede llevar varias medallas pero una medalla solo se la puede poner un jugador.

Relación 7, Hab_Estrella - Jugador (1-N):

El jugador puede aprender varias habilidades estrellas (tener habilidad estrella) pero solo puede tener activa una habilidad estrella.

Relación 8, Jugador - Partida - Usuario (N-N):

La idea inicial es que cada usuario pueda manejar a uno o varios jugadores, lo que equivaldría a distintas "ranuras de guardado", lo que se refleja en la tabla intermedia "Partidas". Esta relación es la base del guardado de partidas del juego.

Un jugador puede ser jugado por varios usuarios y un usuario puede jugar varios jugadores generando la tabla partida.

Un usuario puede jugar varias partidas y una partida solo puede ser jugada por un usuario.

Un jugador puede asociarse a varias partidas pero una partida solo se usa un jugador.

Relación 9, Items - Jugador_Items - Jugador (N-N):

Otra relación clave sería la que une "Jugador" con "Ítems", mediante la tabla intermedia "jugador_items", que permite mantener el número de ítems de cada tipo por jugador, lo que de otra forma no sería posible, ya que se omitiría información.

Un jugador puede tener varios ítems pero un ítem puede pertenecer a varios jugadores. creando una dependencia (entidad "jugador_Items" como inventario)

Un jugador puede tener varios inventarios, pero un inventarios solo es de un jugador.

Un ítem puede estar en varios inventarios pero un inventarios solo tiene ese ítem.

Implementación y consultas

Consultas simples

Todos los jugadores con más de 50 monedas y 30 puntos flor.

```
for $Jugador in doc("../Tablas en XML/jugador.xml")//Jugador
where $Jugador/number(Monedas) > 50 and $Jugador/number(PF) > 20
return $Jugador
```

Devuelve todas las filas de la tabla "Jugador" donde el número de monedas sean superior a 50 y los puntos de flor, a 15.

El nombre de las habilidades estrella que consuman más de 4 puntos estrella.

```
for $nombre in doc("hab_estrella.xml")/Habs_Estrella/Hab_Estrella
where $nombre/Coste_PM>4
return $nombre/Nombre
```

Recupera los nombres de "Hab_Estrella" donde el valor en la columna "Coste_PE" sea mayor que 4.

El nombre de los lugares donde haya más de 10 enemigos y 5 NPCs.

```
for $Lugar in doc("lugar.xml")//Lugar
where $Lugar/number(Enemigos) > 10 and $Lugar/number(Npc) > 5
return $Lugar/Nombre/text()
```

Muestra los nombres de "Lugar" donde el valor en la columna "Enemigos" sea mayor que 10 y el valor en la columna "NPCs" es mayor que 5.

Consultas compuestas

El nombre de los enemigos que vuelan que vivan en X región.

```
for $enemigo in doc("enemigos.xml")//Enemigo
let $lugarEnemigo := doc("lugar_enemigo.xml")//Lugar_Enemigo[@Enemigo_ID = $enemigo/@Enemigo_ID]
where $enemigo/@Vuela = "1"
and $lugarEnemigo/@Nombre_Lugar = "Llanura Estelar"
return $enemigo/Nombre/text()
```

Une las tablas "Enemigo" y "Lugar" mediante un nombre de enemigo común.

Luego obtiene los nombres de la tabla "Enemigo" donde el valor en la columna "Vuela" es igual a 1 (es decir, puede volar) y el valor de "Nombre_Lugar" es igual a "Llanura Estelar" (este valor puede ser sustituido por cualquier otra región del juego donde haya enemigos).

La descripción de un ítem cuyo nombre sea X del jugador con id X.

```
for $items in doc("items.xml")//item
for $jugador_items in doc("jugador_items.xml")//Jugador_Items[items_Nombre = $items/nombre]
where $items/nombre = "Concha Protectora"
and $jugador_items/@Jugador_ID = "4"
return $items/descripcion
```

Une la tabla "Items" y "Jugador_Items" mediante un nombre de ítem común.

Posteriormente muestra los ítems correspondientes a "Concha Protectora" y pertenezcan al jugador identificado como "04".

Mostrar el nombre de los enemigos que se han enfrentado al jugador con id X agrupados según el número de ayudantes que han tenido en la batalla

```
for $enemigo in doc("../Tablas en XML/enemigos.xml")//Enemigo
let $batalla := doc("../Tablas en XML/batallas.xml")//Batalla[$enemigo/@Enemigo_ID=@Enemigo_ID]
where $batalla and $batalla/@Jugador_ID="1"
return $enemigo/Nombre/text()
```

Une las tablas "Enemigo" y "Batalla" mediante un identificador de enemigo común.

Después, selecciona las batallas donde el jugador sea "01" y los agrupa por el número de ayudantes.

Consultas agrupadas

Agrupar las habilidades estrella según su coste y mostrarlas.

```
for $coste in doc("hab_estrella.xml")//Hab_Estrella
group by $c:= $coste/Coste_PM
let $appareances := count($coste/Coste_PM)
return element Hab_Estrella{$appareances, distinct-values($coste/Coste_PM)}
```

Busca obtener las descripciones de la tabla "Hab_Estrella" agrupadas por el valor de "Coste_PE".

Calcular la media de PF de los personajes según su procedencia

```
let $procedenciasUnicas := distinct-values(doc("personaje.xml")//Personaje/Procedencia/text())
for $procedencia in $procedenciasUnicas
return fn:concat($procedencia," ",avg(doc("personaje.xml")//Personaje[Procedencia = $procedencia]/PF))
```

Hace el promedio de puntos flor (PF) de los personajes, y luego los agrupa por procedencia.

Mostrar nombre, puntos de fuerza y ataque total de todas las habilidades de un personaje agrupadas por el nombre

```
for $p in doc("../Tablas en XML/personaje.xml")//Personaje
let $h := doc("../Tablas en XML/habilidad.xml")//Habilidad[$p/Nombre = @Personaje_nom]
order by $p/Nombre
return fn:concat($p/Nombre/text()," ",$p/PF/text()," ",sum($h/Ind_atq))
```

Calcular el número total de ítems de los jugadores

```
for $jugi in doc("..//Tablas en XML/jugador_items.xml")//Jugador_Items
order by xs:integer($jugi/@Jugador_ID)
return fn:concat($jugi/@Jugador_ID," ",$jugi/num_Items)
```

Bibliografía

https://www.mariowiki.com/Paper_Mario:_The_Thousand-Year_Door_bestiary

- Bestiario de Paper Mario: La Puerta Milenaria (en inglés)

https://www.mariowiki.com/List_of_Paper_Mario:_The_Thousand-Year_Door_enemy_formations

- Formaciones de enemigos en Paper Mario (en inglés), imagen de las batallas.

https://www.freeformatter.com/xml-formatter.html

- XML-formatter: herramienta online para dar formato y validar archivos XML.