Universidad Rey Juan Carlos

Escuela Técnica Superior Ingeniería Informática

Práctica 3 - MongoDB

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

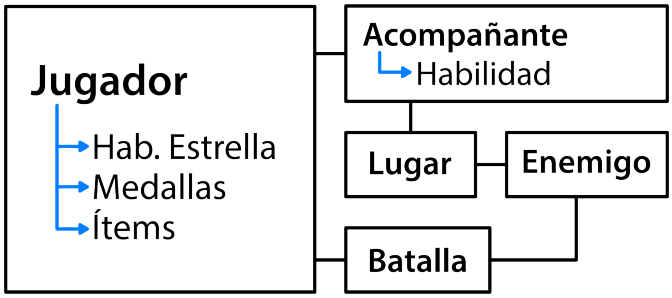
Índice

Introducción

Esta tercera y última práctica de la asignatura tiene como objetivo la creación de una base de datos completa, junto con varias consultas correspondientes, usando el sistema de bases de datos noSQL MongoDB (https://www.mongodb.com/) y tomando como base aquellas realizadas en las dos prácticas anteriores, que se habían realizado con tecnologías basadas en SQL y XML respectivamente. El juego seleccionado fue ***Paper Mario: La Puerta Milenaria*** (Nintendo, 2004).

Análisis

MongoDB presenta numerosas diferencias estructurales frente a SQL, siendo una de ellas el uso de **colecciones** en vez de **tablas**, las cuales son más flexibles y permiten el uso de más tipos de datos. Esta flexibilidad nos ha permitido consolidar las once tablas que teníamos en la base de datos original en cinco colecciones en MongoDB.



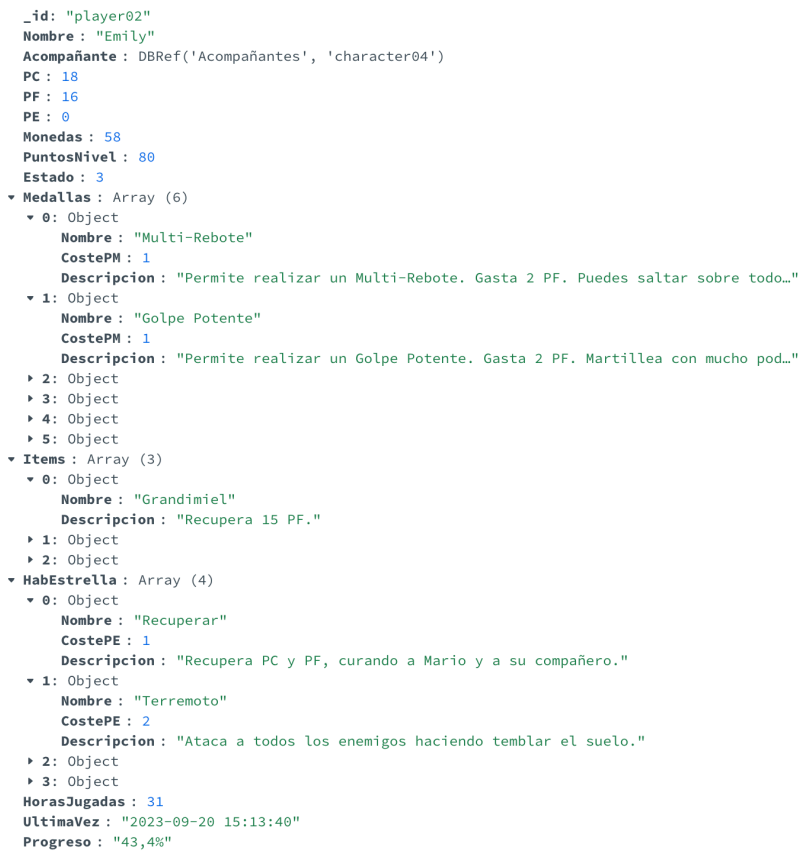
El diagrama superior muestra las nueve tablas (de 11) que teníamos originalmente y cómo se han consolidado en cinco colecciones, siendo estas **jugador, acompañante, lugar, enemigo y batalla.**

Inserción de datos

Para la inserción de la información de la base de datos (todos los jugadores con su información correspondiente, acompañantes, enemigos, etc.) se ha seguido un procedimiento similar al de la primera práctica, en la que se han generado tablas de Microsoft Excel con muchos elementos con datos aleatorios (dentro de rangos coherentes) usando fórmulas elaboradas, tras lo cual se han exportado a CSV y se han pasado por un conversor CSV a JSON para su posterior adaptación e inserción en MongoDB.

_id	Nombre	Region	PuntosAtaque	PuntosDefensa	PuntosCorazon	Vuela	TienePincho	TieneCaparazon	EsBoss
enemy001	Goomba	{ "$ref": "Lugares", "$id": "place22" }	1	0	2	false	false	false	false
enemy002	Paragoomba	{ "$ref": "Lugares", "$id": "place22" }	1	0	2	true	false	false	false
enemy003	Goompincho	{ "$ref": "Lugares", "$id": "place22" }	2	0	2	false	true	false	false
enemy004	Híper Goomba	{ "$ref": "Lugares", "$id": "place12" }	2	0	8	false	false	false	false
enemy005	Híper Paragoomba	{ "$ref": "Lugares", "$id": "place12" }	2	0	8	true	false	false	false
enemy006	Híper Goompincho	{ "$ref": "Lugares", "$id": "place12" }	3	0	8	false	true	false	false
enemy007	Gloomba	{ "$ref": "Lugares", "$id": "place21" }	3	0	7	false	false	false	false
enemy008	Paragloomba	{ "$ref": "Lugares", "$id": "place21" }	3	0	7	true	false	false	false

Colección Jugadores



Esta colección es con diferencia la más compleja de las cinco. Contiene tanto la información del jugador como la información de la partida en la que está. A diferencia de la práctica anterior, hemos asumido que la relación entre ambos es de 1:1, y de esta manera se han consolidado las tablas jugador, partida y usuario en una sola.

El _id de cada elemento de la colección tiene el formato “playerXXX” y los atributos incluyen los siguientes:

- Nombre: nombre o apodo del jugador.
- Acompañante: en las prácticas anteriores recibía el nombre de personaje, se ha cambiado para que no preste a confusión. Contiene una referencia a la colección Acompañantes (ver más abajo) que indica el acompañante activo en ese momento.
- PC, PF y PE (Puntos Corazón, Flor y Estrella), que son los puntos de vida, puntos para habilidades y ataques especiales y puntos para habilidades estrella, respectivamente.
- Monedas, que son necesarias para comprar ítems y medallas, entre otros.
- Puntos de nivel: cuando el jugador gana una batalla, consigue una cantidad de puntos de nivel* que varía en función del número de enemigos y su dificultad relativa al jugador. Cuando este consigue 100 puntos, sube de nivel y puede aumentar sus PC, PF o PM (Puntos Medalla).
- Estado: estado interno del jugador (Mario)

— Arrays —

- Medallas: contiene todas las medallas que porta el jugador. Cada medalla tiene los atributos nombre, coste (en PM) y descripción.
- Ítems: contiene todos los ítems que porta el jugador. Cada ítem se identifica por un nombre y una descripción.
- Hab. Estrella: contiene todas las habilidades estrella que el jugador puede ejecutar durante las batallas. Cada habilidad estrella se identifica por su nombre, su coste (en PE) y una descripción.

— Variables de la partida —

- Horas jugadas
- Última vez: contiene la fecha y hora de la última vez que se guardó la partida en formato ISO 8601 (año-mes-día hora-minutos-segundos)
- Progreso: porcentaje de todo el juego que ha superado el jugador.

*En el juego real, los puntos de nivel, peculiarmente, también se llaman "Puntos Estrella", por lo que para no confundirlos con los PE usados en las Habilidades Estrella, aquí los hemos llamado "puntos de nivel".

Colección Acompañantes


```
_id: "character01"
Nombre : "Goomarina"
PC : 25
PF : 15
Procedencia : DBRef('Lugares', 'place01')
Descripcion : "Una Goomba universitaria que aspira a ser arqueóloga."
▾ Habilidades : Array (4)
  ▾ 0: Object
      Nombre : "Cabezazo"
      Coste_PF : 0
      Indice_ataque : 2
      Indice_defensa : 0
      Descripcion : "Ataca a un enemigo de un cabezazo."
  ▾ 1: Object
      Nombre : "Descripción"
      Coste_PF : 0
      Indice_ataque : 0
      Indice_defensa : 0
      Descripcion : "Muestra la descripción del enemigo y sus PC."
  ▸ 2: Object
  ▸ 3: Object
```


La colección Acompañantes contiene a todos los acompañantes presentes en el juego y sus atributos. Es la colección que menos elementos tiene, ya que solo hay 7 acompañantes en total. El _id de cada uno tiene la estructura "characterXX" y los atributos son los siguientes:

- Nombre del acompañante.
- Puntos Corazón y Puntos Flor (PC y PF).

- Procedencia: Hace referencia a la colección Lugares (ver más abajo), e indica el lugar de donde proviene el acompañante (o lo que es lo mismo: dónde puede el jugador encontrar al acompañante).
- Descripción.
- Array de habilidades, que contiene todas las habilidades que puede ejecutar el acompañante durante las batallas. Cada habilidad está definida por los atributos nombre, coste (en PF), un índice de ataque y otro de defensa, y una descripción.

Colección Lugares

```
_id: "place01"
Nombre : "Villa Viciosa"
NPCs : 10
Enemigos : 17
```

La colección Lugares contiene todos los lugares dentro del juego. El _id de cada uno tiene la estructura "placeXX" y contiene los siguientes atributos:

- Nombre del lugar.
- Número de NPCs (Personajes No Jugables) del lugar.
- Número de enemigos del lugar.

Colección Enemigos

```
_id: "enemy035"
Nombre : "Punzón Rojo"
Region : DBRef('Lugares', 'place10')
PuntosAtaque : 3
PuntosDefensa : 4
PuntosCorazon : 5
Vuela : false
TienePincho : true
TieneCaparazon : true
EsBoss : false
```

Esta colección es la que más elementos contiene de todas. Define los enemigos que se encuentran en el juego, junto a todas sus características relevantes. El _id de cada uno tiene la estructura "enemyXXX" y contiene los siguientes atributos:

- Nombre del enemigo, tal y como aparece en el juego (al mostrarse su descripción).
- Región donde puede hallarse. Es una referencia a la colección Lugar (ver arriba). Nótese que un enemigo puede hallarse en múltiples lugares, por lo que se ha definido un lugar comodín llamado "Diversos Lugares" con _id "place22" (que es como aparece en el juego).
- Valores numéricos para puntos de ataque y defensa, y puntos corazón (vida).

- Cuatro atributos booleanos, que definen si el enemigo:
    1. Puede volar (por lo que no le afectan ataques a nivel de suelo, como el martillo o la habilidad terremoto).
    2. Tiene pincho (por lo que no recibe daño con ataques de pisotón, salto).
    3. Tiene caparazón (por lo que no recibe daño hasta ser colocado boca arriba).
    4. Es boss (por lo que no se podría huir de la batalla contra el mismo).

Colección Batallas

```
_id: "battle013"
Enemigo : DBRef('Enemigos', 'enemy054')
Jugador : DBRef('Jugadores', 'player13')
NumTurnos : 2
NumAyuda : 2
```

La colección recopila las batallas que hay presentes en el juego. El _id de cada una sigue la estructura "battleXXX" y tiene los siguientes atributos:

- Enemigo contra el que se está combatiendo. Es una referencia a la colección Enemigos (ver arriba).
- Jugador que se está enfrentando al enemigo. Es una referencia a la colección Jugadores (ver arriba).
- Número de turnos que ha durado la batalla.
- Número de ayudas que se han recibido durante la batalla.

Consultas CRUD (simples)

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

```
db.Jugadores.find({Monedas:{$gt:50}, PF:{$gt:30}})
```

Todas las habilidades estrella que consuman más de 4 puntos estrella.

```
db.Jugadores.aggregate([
  {
    $unwind: "$HabEstrella" // Descomponer el array "HabEstrella" en documentos individuales
  },
  {
    $match: {
      "HabEstrella.CostePE": { $gt: 4 } // Filtrar habilidades estrella con un coste mayor a 1
    }
  },
  {
    $project: {
      _id: 0,
      Nombre: "$HabEstrella.Nombre",
      CostePE: "$HabEstrella.CostePE",
      Descripcion: "$HabEstrella.Descripcion"
    }
  }
])
```

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

```
db.Lugares.find({$and: [{"Enemigos":{$gt:10}}, {"NPCs":{$gt:5}}]}, {"_id": 0, "Nombre": 1})
```

Consultas CRUD (compuestas)

Todos los enemigos que vuelan que vivan en X región.

En este caso, se ha seleccionado como ejemplo la región "Llanura Estelar".

```
db.Enemigos.aggregate([
  {
    $lookup: {
    from: "Lugares",
    localField: "Region.$id",
    foreignField: "_id",
    as: "Enemigos_Lugares"
  }},
    {$unwind:"$Enemigos_Lugares"},
  {
    $match: { "Enemigos_Lugares.Nombre": "Llanura Estelar"
  }},
  {
    $match: { "Vuela": true
  }},
  {
    $project:{_id:0,Nombre:1
  }}
])
```

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

En este caso, obtenemos la descripción del ítem "Seta" que pertenece al jugador identificado por el ID "player01".

```
db.Jugadores.aggregate([
  {
    $match: { "_id": "player01" } // Seleccionar ID jugador
  },
  {
    $unwind: "$Items" // Descomponer array
  },
  {
    $match: { "Items.Nombre": "Seta" } // Filtrar por nombre
  },
  {
    $project: {
      _id: 0,
      Descripcion: "$Items.Descripcion"
    }
  }
])
```

Nombre de los enemigos que se han enfrentado al jugador con id X.

```
db.Batallas.aggregate({$match:{"Jugador.$id":'player02'}},
{$lookup:{from:'Enemigos',localField:'Enemigo.$id',foreignField:'_id',as:'enemigo_info'}},{$unwind:'$enemigo_info'},
{$project:{_id:0,Nombre:'$enemigo_info.Nombre'}});
```

Consultas CRUD (agrupadas)

Agrupar las habilidades estrella según su coste y mostrar cuántas habilidades hay de cada coste.

```
db.Jugadores.aggregate([
    { $unwind: "$HabEstrella" },
    {
        $group: {
            _id: "$HabEstrella.CostePE",
            count: { $sum: 1 }
        }
    }
])
```

Calcular la media de PC de los personajes según su procedencia.

```
db.Acompañantes.aggregate([{$group:{_id:'$Procedencia',avgPC:{$avg:'$PC'}}},{$project:{Procedencia:'$_id',avgPC:{$round:['$avgPC',2]}}}]);
```

Calcular el número total de ítems de los jugadores.

```
db.Jugadores.aggregate([
  {
    $project: {
      itemCount: { $size: "$Items" }
    }
  },
  {
    $group: {
      _id: null,
      totalItems: { $sum: "$itemCount" }
    }
  }
])
```

Índices

Jugadores que pueden subir de nivel

```
db.Jugadores.createIndex({"PuntosNivel":1}, {name:"SubirNivel", partialFilterExpresion:{PuntosNivel:{$gt:100}}});
```

Jugadores que pueden abrir la Puerta Milenaria (todas las estrellas)

```
db.Jugadores.createIndex({"PE":1}, {name:"AbrirPuertaMilenaria", partialFilterExpresion:{PE:{$lt:8}}});
```

Jugadores que tienen un estado concreto

```
db.Jugadores.createIndex({"_id":1, "Estado":1});
```

Jugadores que tienen un gran número de puntos flor y de monedas

```
db.Jugadores.createIndex({"Monedas": 1, "PF": 1},{name:"ConsultaSimple1", partialFilterExpresion:{Monedas:{$gt:50}, PF:{$gt:30}}});
```

Jugadores que tienen habilidad estrella 1

```
db.Jugadores.createIndex({"HabEstrella.CostePE":1});
```

Reduce la búsqueda a los enemigos que son débiles

```
db.Enemigos.createIndex({"PuntosAtaque":1, "PuntosDefensa":1})
```

Reduce la búsqueda a los enemigos que son boss

```
db.Enemigos.createIndex({"Nombre": 1},{name:"EsBoss", partialFilterExpression:{EsBoss:true}})
```

Evita nombres duplicados

```
db.Enemigos.createIndex({"Nombre":1}, {"unique":true})
```

Mejorar eficiencia de las búsquedas por procedencia

```
db.Acompañantes.createIndex({ "Procedencia": 1 });
```

Duelos de gran duración

```
db.Batallas.createIndex({NumTurnos:1,NumAyuda:1},{name:"Duelo_LargaDuracion",partialFilterExpression:{NumTurnos:{$gt:5},NumAyuda:0}});
```

Grandes enfrentamientos de duración mínima

```
db.Batallas.createIndex({NumTurnos:1,NumAyuda:1},{name:"GranBatalla",partialFilterExpression:{NumTurnos:{$gt:4},NumAyuda:{$gt:4}}})
```

Mostrar solo lugares pacíficos

```
db.Lugares.createIndex({"Enemigos":1},{name:"Lugares pacificos",partialFilterExpression:{"Enemigos":0}})
```

Mostrar lugares peligrosos (con muchos enemigos)

```
db.Lugares.createIndex({"Enemigos":1},{name:"Lugares peligrosos",partialFilterExpression:{"Enemigos":{$gt:15}}})
```

Consultas frecuentes

1 - Las descripciones de los ítems del jugador con id X

Esta consulta se utiliza cuando se abre el inventario, ya que deben deben aparecer todos los ítems en el menú.

```
db.Jugadores.find({ "_id": 'player01' }, {Items.Descripcion: 1, _id: 0 });
```

2- Número de puntos corazón de cada personaje

Aparecen en la interfaz de usuario del juego, por lo que debe accederse al dato constantemente.

```
db.Jugadores.find({ "_id": 'player01' }, {PC: 1, _id: 0});
```

3 - Número de monedas del personaje

Igual que en la anterior, aparece de forma constante en la interfaz de usuario.

```
db.Jugadores.find({ "_id": 'player01' }, {Monedas: 1, _id: 0});
```

4 - Número de puntos estrella del jugador con id X

Durante la batalla, los puntos estrella aparecen en la interfaz superior como varios círculos pequeños que se van llenando gracias a las reacciones del público, y se van gastando al usar habilidades estrella.

```
db.Jugadores.find({"_id":"player03"}, {"_id":0, "PE": 1})
```

5 - Índice de defensa y ataque de cada enemigo

Esta información se obtiene justo antes de comenzar una batalla. Esta información es invisible al jugador hasta que se usa la habilidad Descripción del acompañante Goomarina.

```
db.Enemigos.find({}, {"_id": 0, "PuntosAtaque": 1, "PuntosDefensa": 1})
```

6 - Salud del enemigo con nombre X

En la batalla se irá comprobando continuamente.

```
db.Enemigos.find({"Nombre":"Koopa Troopa"}, {"_id": 0, "PuntosCorazon": 1})
```

7 - El número de ayudantes que tiene una batalla

Cada vez que haya una batalla se accederá al dato para generar los enemigos secundarios.

```
db.Batallas.find({"_id":"battle001"},{"_id":0, "NumAyuda":1})
```

8 - Número de puntos flor de los que dispone el jugador.

Es necesario mostrarlo durante las batallas para que el jugador sepa qué habilidades puede realizar.

```
db.Jugadores.find({"_id":"player01"},{_id:0,"PF":1})
```

9 - Ordenar las habilidades estrella por coste de menor a mayor.

Al usar una habilidad estrella, se deben mostrar en una lista según su coste de menor a mayor.

```
db.Jugadores.aggregate([
  {
    $unwind: "$HabEstrella"
  },
  {
    $project: {
      _id: 0,
      Nombre: "$HabEstrella.Nombre",
      CostePE: "$HabEstrella.CostePE",
      Descripcion: "$HabEstrella.Descripcion"
    }
  },
  {
    $sort: {
      CostePE: 1
    }
  }
])
```

10 - Comprobar si el número de PE que tiene el jugador es inferior a 8

Ya que el número máximo de PE que puede tener el jugador son 8 (7 + 1, la que dan por defecto), cuando llega a 8 y el jugador va a la puerta milenaria, esta se abre.

```
db.Jugadores.find({PE:{$lt:7}},{_id:0,PE:1})
```

11 - Comprobar si el número de puntos nivel es igual o superior a 100

Cuando el jugador llega a 100 puntos de nivel, tras el final de una batalla, este sube de nivel y los puntos se vuelven a dejar a 0.

```
db.Jugadores.find({$PuntosNivel:{$gt:100}},{_id:1})
```

12 - Las habilidades que tiene el personaje acompañante del jugador

Durante el turno del acompañante en las batallas, se deben mostrar las habilidades que este puede realizar.

```
db.Acompañantes.find({},{_id:0,Habilidades:1})
```

13 - Saber si un enemigo es jefe o no

Ya que el jugador no podrá huir de una batalla en la que el enemigo sea un boss, es necesario hacer una comprobación justo antes del comienzo de la batalla.

```
db.Enemigos.find({},{_id:0,EsBoss:1})
```

14 - Consultar el lugar de procedencia de un personaje acompañante

Es importante para saber dónde se puede conseguir el personaje.

```
db.Acompañantes.find({},{_id:0,Procedencia:1})
```

15 - Consultar el estado del jugador

Antes de comenzar (reanudar) una partida ya existente, es necesario comprobar el estado en el que se encontraba el jugador antes de cerrar la partida.

```
db.Jugadores.find({},{_id:0,Estado:1})
```

¿Por qué estas consultas?

Hemos elegido estas consultas como las más frecuentes ya que permiten conocer datos cruciales de cada uno de los elementos del juego. Se han incluido consultas en las que se quiere conocer estadísticas del jugador, como ver los puntos estrella de los dispone el jugador, para saber qué habilidades estrella puede usar y cuáles no, o consultar las monedas que tiene y saber qué puede comprar en la tienda. También hay consultas que permiten saber la situación de la batalla, como conocer el número de ayudantes que tiene el enemigo durante la misma; o consultas que nos permiten conocer datos del enemigo al que nos enfrentamos, como saber si es un boss o no, para determinar si el jugador puede huir de la batalla o no, así como índice de ataque y defensa.

Estas consultas son necesarias ya que son las que sustentan el sistema de combate del juego, además de los datos del jugador e incluso información de la propia partida.

Recursos y 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://csvjson.com/csv2json

- Conversor en línea de archivos CSV a JSON