Proyecto 3º Trimestre

Jairo Rastrojo Rodríguez

1º ASIR: Gestión de Base de Datos (GBD)

Adolfo Salto Sánchez del Corral

Introducción

La base de datos que trataremos en este proyecto está basada en la liga española de League Of Legends, para ello dispone de 3 colecciones diferentes

1. Colecciones

1.1 Jugadores

Dicha colección dispone de un total de 56 jugadores pertenecientes a diferentes equipos, los cuales trataremos en la siguiente colección, pero centrándonos en esta, vamos a echar un vistazo a su estructura:

_id: (Number) → Número de identificación del jugador en cuestión.

NickName: (String) → Nombre de invocador o jugador que ha elegido dicho jugador para su cuenta.

Nombre: (String) → Nombre real del jugadores

Apellido: (String) → Primer apellido del jugador. (solo el primero ya que algunos no tienen segundo reflejado en su información, así evitamos datos nulos)

Edad: (Number) → Edad del jugadores

País: (String) → País de origen del jugadores

Posición: (String)→ Nombre de la posición que juega en el equipo dicho jugador

KDA: (Number) → Número que determina la calidad del jugador refiriéndose a sus resultados de partidas.

Total: (Array) → Es un array de documentos, el cual posee 3 documentos:

- Asesinatos: (Number) → Asesinatos que ha llevado a cabo el jugador.

-Muertes: (Number) → Número total de veces que el jugador ha sido asesinado por el equipo rival.

-Asistencias: (Number) → Número total de ayudas que ha brindado el jugador a sus compañeros.

Campeones_destacados: (Array) → Es un array de String que determina los campeones o personajes preferidos de ese jugador

Equipo: (String) → Nombre del equipo al que pertenece el jugador.

1.2 Equipos

_id: (Number) → Número identificador del Equipo

Nombre: (String) → Nombre del equipo

Entrenador: (Arrya) → Es un array de documentos referentes a la información personal del entrenador del equipo:

-NickName: (String) → Nombre de invocador o jugador del entrenador

-Nombre: (String) → Nombre Real del Entrenador

-Apellido: (String) → Apellido del entrenador

-Edad: (Number) → Edad del entrenador

-País: (String) → País de origen del entrenador

Patrocinado: (Boolean) → Es un campo de respuesta true o false, dependiendo de si es o no patrocinado el equipo.

Patrocinador: (Array) → Puede ser un array de Strings o un solo string, dependerá del número de patrocinadores que tenga el equipo.

1.3 Partidas

_id: (Number) → Número identificador de la partidas

Jornada (String) → Se trata o bien del número de la jornada o bien de la fase del torneo

Equipo1: (String) → Nombre del equipo que juega del lado izquierdo del mapa.

Equipo2: (String) → Nombre del equipo que juega del lado derecho del mapa

Resultado: (String) → Resultado de las partidas

Fecha: (Date) → Fecha del encuentro (Todas las partidas son de 2021, pero son desde enero hasta marzo)

Ganador: (String) → Nombre del equipo que sale victorioso del encuentro.

2. Operadores

$lookup: realiza una unión entre colecciones para poder alcanzar resultados más completos y complejos a la hora de realizar consultas que requieran información de varias colecciones.

Estructura:
{
$lookup: {
    from: “Nombre de colección a la que se quiere unir”
    localField: “Nombre del campo local que queremos unir”
    foreignField: “Nombre del campo de la colección que se ha unido que queremos enlazar”
    as: “Nombre se se le da a la salida de la unión”
}
}

$project: Pasa los documentos con los campos solicitados a la siguiente etapa del proceso, se usa principalmente para editar y gestionar la salida final de la consulta.

Estructura:

{$project: { <specification (s)>}}

$group: Permite agrupar los documentos entrantes, por la _id especificada y para cada agrupación genera documentos, suele utilizarse para realizar operaciones matemáticas en conjunto.(Si no se hacen operaciones después de un grup, es una agrupación sin sentido)

Estructura:

```
{
    $group:
        {
            _id: <expression>,
            <field1>: {<accumulator1> : <expression1>},
            ...
        }
}
```

$sort: Permite ordenar la salida de la consulta, 1 ascendente, -1 descendente

Estructura:

{$sort: {<field1>: <sort order>, <field2>: <sort order> …}}

$unwind: Deconstruye un array dado, para poder de esa forma realizar operaciones individuales con el contenido de ese array

Estructura:

{$unwind: <field path> }

3. Consultas

3.1 Fuera de replit

3.1.1 Consulta 1

Queremos que se nos muestre por pantalla el NickName de cada jugador junto al equipo que pertenece y el NickName del entrenador que tienen asignado:

```
db.jugadores.aggregate([
{
$lookup:
{
from: "equipos",
localField: "Equipo",
foreignField: "Nombre",
as: "Entrenador"
}
},
{
$project: {
Invocador: "$NickName",
Equipo: "$Equipo",
Entrenador: "$Entrenador.Entrenador.NickName"
}
}
]).pretty()
```

3.1.2 Consulta 2

¿Cuantas victorias ha tenido cada equipo en total en lo que llevamos de temporada?

```
db.partidas.aggregate([
{
$group: {
_id:"$Ganador",
Total_Victorias: {$sum:1}
}
},
{
$sort: {
Total_Victorias: -1
}
}
]).pretty()
```

3.1.3 Consulta 3

Para Valorar y comparar las estadísticas generales de los equipos, queremos la media del kda de cada equipo y el acumulado de los asesinatos, muertes y asistencias, ademas nos interesa saber que la edad del jugador más viejo y el más joven.

```
db.jugadores.aggregate([
{
$unwind: "$Total"
},
{
$group: {
_id: "$Equipo",
Media_Kda: {$avg: "$KDA"},
Mas_Viejo: {$max: "$Edad"},
Mas_Joven: {$min: "$Edad"},
T_Asesinatos: {$sum: "$Total.Asesinatos"},
T_Muertes: {$sum: "$Total.Muertes"},
T_Asistencias: {$sum: "$Total.Asistencias"}
}
},
{
$project: {
_id:0,
Equipo: "$_id",
Media_KDA: {$round: ["$Media_Kda", 2]},
Jugador_Veterano: "$Mas_Viejo",
Jugador_Novel: "$Mas_Joven",
Total_Asesinatos: "$T_Asesinatos",
Total_Muertes: "$T_Muertes",
Total_Asistencias: "$T_Asistencias"
}
}
]).pretty()
```

3.1.4 Consulta 4

Cada jugador tiene sus campeones preferidos a la hora de jugar, pero segun dichos gustos y preferencias, ¿qué campeones son los más jugados o elegidos como mejores opciones?

```
db.jugadores.aggregate([
{
$unwind: "$Campeones_destacados"
},
{
$group: {
_id: "$Campeones_destacados",
Veces_jugadas: {$sum: 1}
}
},
{
$sort: {
Veces_jugadas: -1
}
}
]).pretty()*/
```

3.2 Dentro de replit

3.2.1 Consulta 1

La cabecera será la página primera que se vera y a partir de la cual se podrá acceder a las consultas. (**no es una consulta pero se ha metido en esta sección ya que ha sido código introducido en el index.ts)**

```
const cabecera = async (req: Request, res: Response) => {
res.send("Proyecto 3º Trimestre: Liga Española de League Of Legends")
}
```

3.2.2 Consulta 2

Esta primera consulta llamada "primera", nos mostrará la media de asesinatos, muertes y asistencia de los ADC de los equipos, para saber cual equipo tiene las mejores estadisticas en la posición de ADC

```
const primera = async (req: Request, res: Response) => {
await db.conectarBD()
.then(
async (mensaje) => {
console.log(mensaje)
const query = await Jugadores.aggregate([
{
$match: {"Posición":"ADC"}
},
{
$unwind: "$Total"
},
{
$group: {
_id: "$Equipo",
Media_Asesinatos: {$avg: "$Total.Asesinatos"},
Media_Muertes: {$avg: "$Total.Muertes"},
Media_Asistencias: {$avg: "$Total.Asistencias"}
}
}
])
res.json(query)
})
.catch(
(mensaje) => {
res.send(mensaje)
console.log(mensaje)
})
db.desconectarBD()
}
```

3.2.3 Consulta 3

En la segunda consulta queremos saber que equipo va primero en la liga, es decir que equipo ha ganado más partidas, sin contar el torneo

```
const segunda = async (req: Request, res: Response) => {
await db.conectarBD()
.then(
async (mensaje) => {
console.log(mensaje)
const query = await Partidas.aggregate([
{
$match:{
$expr: { $lt: [ { $month: "$Fecha" }, 3 ] } }
},
{
$group: {
_id:"$Ganador",
Total_Victorias: {$sum:1}
}
},
{
$sort: {
Total_Victorias: -1
}
}
])
res.json(query)
})
.catch(
(mensaje) => {
res.send(mensaje)
console.log(mensaje)
})
db.desconectarBD()
}
```

3.2.4 Consulta 4

Lo siguiente tampoco es una consulta pero es código que es necesario para que se ejecute la aplicación y funcionen los apartados anteriores de las consultas en replit

```
app.get('/', cabecera)
app.get('/primera', primera)
app.get('/segunda', segunda)
app.listen(3000)
```

4. Páginas de interes y usadas para el proyecto

Página de la superliga de League Of Legends: https://superliga.lvp.global

Enlace directo al proyecto en replit: https://replit.com/@JairoRastrojo1/ASIRrastrojorodriguezjairo

Enlaces para conectarse a la base de datos en mongo Atlas:

- Por replit:

Para el usuario Jairo con permisos de lectura y escritura:

'mongodb+srv://jairo:proyecto@proyecto.znlpy.mongodb.net/liga?retryWrites=true&w=majority'

Para el usuario Kurumi con permisos de Solo lectura:

‘mongodb+srv://kurumi:proyecto@proyecto.znlpy.mongodb.net/liga?
retryWrites=true&w=majority’

-Por Shell/Terminal:

Para el usuario Jairo con permisos de lectura y escritura:

mongo "mongodb+srv://proyecto.znlpy.mongodb.net/myFirstDatabase" --username jairo

Para el usuario Kurumi con permisos de Solo lectura:

mongo "mongodb+srv://proyecto.znlpy.mongodb.net/myFirstDatabase" --username kurumi

-Por Mongo Compass:

Para el usuario Jairo con permisos de lectura y escritura:

mongodb+srv://jairo:proyecto@proyecto.znlpy.mongodb.net/test

Para el usuario Kurumi con permisos de Solo lectura:

mongodb+srv://kurumi:proyecto@proyecto.znlpy.mongodb.net/test