Pau Riera Guardia DNI: 53866841J´

TAREA FINAL- BASES DE DATOS NO SQL

1. Dataset Club Football Match Data (2000-2025)

Descripción: Este conjunto de datos ofrece información detallada sobre partidos de fútbol de 27 países y 42 ligas, incluyendo estadísticas como goles, posesión, tarjetas, cuotas de apuestas y calificaciones Elo. Los datos abarcan desde la temporada 2000/01 hasta la 2024/25.

Contexto y estructura del Dataset:

El dataset utilizado para esta práctica corresponde a una colección completa de datos históricos relacionados con partidos de fútbol de clubes de diversas ligar internacionales y sus respectivos rating Elo. Esta dividido en dos tablas principales:

Tabla 1: ELO_RATINGS.csv

Esta tabla contiene valores de rating Elo para clubes de fútbol, extraídos del sitio especializado ClubElo. Las mediciones se toman dos veces al mes (el 1 y el 15) y se almacenan en forma de snapshots (instantáneas). Algunos nombres de clubes han sido adaptados para mantener consistencia con la tabla de partidos.

Columna	Tipo de dato	Descripción
Date	Date	Fecha del snapshot
Club	String	Nombre del Club
Country	Enum	Código de país
Elo	Float	Rating Elo actual del club

Tabla 2: MATCHES.csv

Esta tabla recoge resultados de partidos y estadísticas clave obtenidas del sitio Football-Data.co.uk, desde el inicio de la temporada 2000/01 hasta diciembre de 2024. A pesar de que hay diferencias en los datos disponibles según cada liga, se proporciona información suficiente para diversos análisis y consultas.

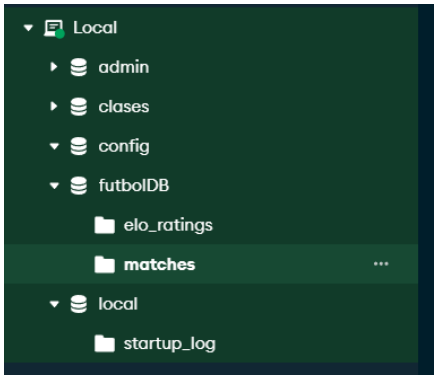
Columna	Tipo de dato	Descripción
Division	Enum	Código de liga
MatchDate	Date	Fecha del partido
MatchTime	Time	Hora del partido
HomeTeam	String	Nombre del equipo local
AwayTeam	String	Nombre del equipo visitante
HomeElo	Float	Rating Elo más reciente del equipo local
AwayElo	Float	Rating Elo más reciente del equipo visitante
Form3Home	Int	Puntos en los últimos 3 partidos del equipo local
Form5Home	Int	Puntos en los últimos 5 partidos del equipo local
Form3Away	Int	Puntos en los últimos 3 partidos del equipo visitante
Form5Away	Int	Puntos en los últimos 5 partidos del equipo visitante

FTHome	Int	Goles anotados por el equipo local
FTAway	Int	Goles anotados por el equipo visitante
FTResult	Enum	Resultado final del partido
HTHome	Int	Goles anotados por el local en el primer tiempo
HTAway	Int	Goles anotados por el visitante en el primer tiempo
HTResult	Enum	Resultado del primer tiempo
HomeShots	Int	Total tiros equipo local
AwayShots	Int	Total tiros equipo visitante
HomeTarget	Int	Tiros a la portería equipo local
AwayTarget	Int	Tiros a la portería equipo visitante
HomeFouls	Int	Faltas cometidas local
AwayFouls	Int	Faltas cometidas visitante
HomeCorners	Int	Corners local
AwayCorners	Int	Corners Visitante
HomeYellow	Int	Tarjetas amarillas local
AwayYellow	Int	Tarjetas amarillas visitante
HomeRed	Int	Tarjetas rojas local
AwayRed	Int	Tarjetas rojas visitante
OddHome	Float	Cuota victoria local Bet365
OddDraw	Float	Cuota empate Bet365
OddAway	Float	Cuota victoria visitante Bet365
MaxHome	Float	Cuota máxima victoria local
MaxDraw	Float	Cuota máxima empate
MaxAway	Float	Cuota máxima victoria visitante
Over25	Float	Cuota de +2,5 goles totales Bet365
Under25	Float	Cuota de -2,5 goles totales Bet365
MaxOver25	Float	Cuota máxima +2,5 goles totales
MaxUnder25	Float	Cuota máxima -2,5 goles totales
HandiSize	Float	Tamaño del hándicap asiático
HandiHome	Float	Cuota de vitoria local con hándicap
HandiAway	Float	Cuota victoria visitante con hándicap

2. Código de las consultas y Resultados

2.1. Carga e Importación del dataset

Para la importación de los dataset descargados a través de Github y comentados con anterioridad, utilizaremos la herramienta MongoDB Compass. Para ello, contactándonos en el local de nuestro ordenador, deberemos crear una nueva base de datos que llamaremos futbolDB. Dentro de ella crearemos dos colecciones, una llamada elo_ratings y la otra matches, donde cargaremos los datos de nuestros respectivos csv.



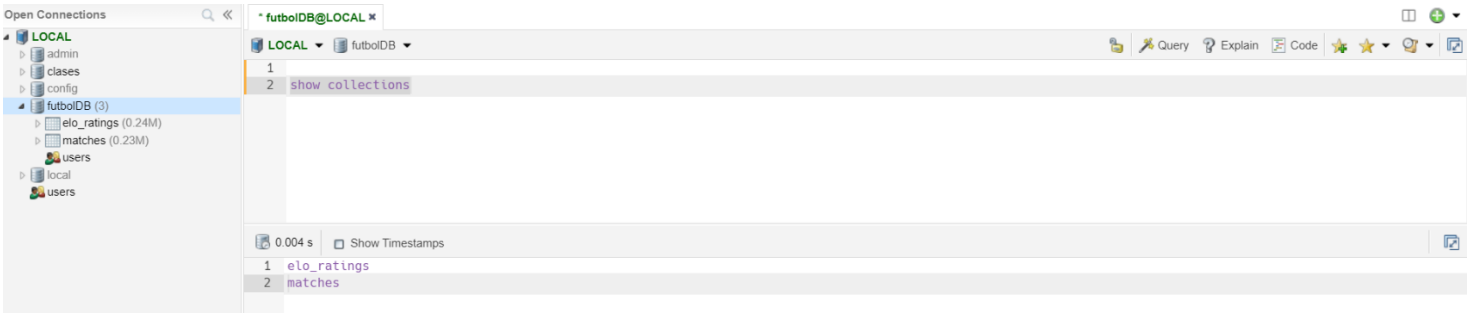
El output final será el siguiente:

Una vez cargados los datos, podemos proseguir con la práctica

2.2. *Ejercicios sobre inserción, actualización, proyección y filtrado*

Para realizar este apartado, haremos uso de la herramienta NOSQLBooster. En ella escribiremos nuestras consultas JavaScript estilo mongo.

Antes de empezar con los distintos ejercicios, vayamos a asegurarnos de que nuestras bases de datos se han generado correctamente en nuestro local. Para ello, hacemos uso de la función *show collections* manteniéndonos en nuestra base de datos futbolDB como se muestra en el siguiente pantallazo.



Efectivamente, nuestros datos se han cargado correctamente en las colecciones *elo_ratings* y *matches*.

Seguidamente deberemos crear los respectivos índices para facilitar nuestra búsqueda y filtrado en las siguientes consultas. Par ello usaremos el siguiente código:

```
// Creación de indices para cada colección

print("Creando índices para elo_ratings...");
db.elo_ratings.createIndex({ Country: 1 });
db.elo_ratings.createIndex({ Club: 1 });
db.elo_ratings.createIndex({ Date: 1 });
db.elo_ratings.createIndex({ Elo: -1 });
```

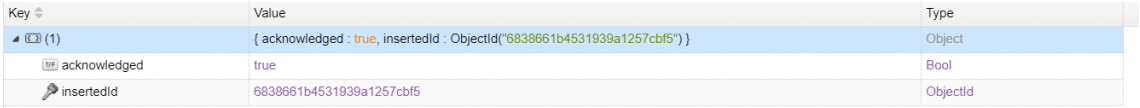
```
print("Creando índices para matches...");
db.matches.createIndex({ MatchDate: 1 });
db.matches.createIndex({ HomeTeam: 1 });
db.matches.createIndex({ AwayTeam: 1 });
db.matches.createIndex({ FTResult: 1 });
db.matches.createIndex({ HTResult: 1 });
db.matches.createIndex({ FTHome: 1 });
db.matches.createIndex({ FTAway: 1 });
db.matches.createIndex({ HomeShots: 1 });
db.matches.createIndex({ AwayShots: 1 });
db.matches.createIndex({ HomeTarget: 1 });
db.matches.createIndex({ AwayTarget: 1 });
db.matches.createIndex({ HomeFouls: 1 });
db.matches.createIndex({ AwayFouls: 1 });
db.matches.createIndex({ HomeCorners: 1 });
db.matches.createIndex({ AwayCorners: 1 });
db.matches.createIndex({ HomeYellow: 1 });
db.matches.createIndex({ AwayYellow: 1 });
db.matches.createIndex({ HomeRed: 1 });
db.matches.createIndex({ AwayRed: 1 });
```

2.2.1. *Inserción de un nuevo documento*

Para realizar la inserción de nuevos datos en nuestras bases de datos, debemos utilizar la función *insertOne()*. En nuestro caso, insertaremos nuevos datos en la colección *elo_ratings* insertándole una nueva fecha (*Date*), un club, un país (*Country*), y un Elo rating. Para ello correremos las siguientes líneas de código.

```
// 2.2.1 Inserción de un nuevo documento
db.elo_ratings.insertOne({
  Date: new Date("2025-03-01"),
  Club: "Valencia CF",
  Country: "ESP",
  Elo: 1470.33,
})
```

El output del código es el siguiente:



Key	Value	Type
(1)	{ acknowledged : true, insertedId : ObjectId("6838661b4531939a1257cbf5") }	Object
acknowledged	true	Bool
insertedId	6838661b4531939a1257cbf5	ObjectId

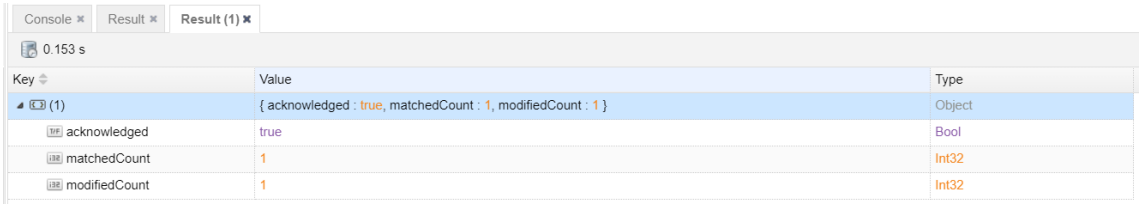
Podemos ver, como los nuevos datos han sido añadidos en la colección *elo_ratings* dándole un nuevo *objectid* a los nuevos datos.

2.2.2. Actualización de datos

Vamos a actualizar el dato de *Elo* del Valencia CF de la colección *elo_ratings* haciendo uso de la función *updateOne()*. Aplicaremos el siguiente código.

```
// 2.2.2. Actualización de datos
db.elo_ratings.updateOne(
  { Club: "Valencia CF" },
  {
    $set: { Elo: 1490.50 },
    $currentDate: { lastUpdated: true }
  }
)
```

El siguiente código, mediante la función *updateOne()* dentro de la colección *elo_ratings*, actualiza el Elo mediante la función *$set* a 1490.50 aplicándole también la fecha actual mediante la función *$currentDate*. El output es el siguiente:



Console × Result × Result (1) ×

0.153 s

Key	Value	Type
(1)	{ acknowledged : true, matchedCount : 1, modifiedCount : 1 }	Object
acknowledged	true	Bool
matchedCount	1	Int32
modifiedCount	1	Int32

En ello se nos muestra que se han modificado los datos mediante el código *modifiedCount:1*.

2.2.3. Proyección y filtrado

Para el siguiente ejercicio trataremos de filtrar 5 partidos donde el equipo local ganó, el equipo local fue agresivo en ataque (más de 3 tiros a puerta) y, además, jugó con pocas faltas (menos de 2).

El código es el siguiente:

```
// 2.2.3. Proyección y filtrado
db.matches.find({
  FTResult: "H",
  HomeTarget: { $gt: 3 },
  HomeFouls: { $lt: 2 }
},
{
  _id: 0,
  MatchDate: 1,
  HomeTeam: 1,
  AwayTeam: 1,
  FTHome: 1,
  FTAway: 1
}).limit(5)
```

En el hacemos uso de las funciones *find()* y *limit()* para realizar filtraciones, entre otras, y hacemos uso también de funciones para proyectar aquellos datos que queremos, por ejemplo, usamos _id: 0 para no proyectar el id.

El output del código es el siguiente:

	MatchDate	HomeTeam	AwayTeam	FTHome	FTAway
1	30/9/2000 2:00:00 - 25 years ago	Aston Villa	Derby	4.0	1.0
2	4/8/2007 2:00:00 - 18 years ago	Valenciennes	Toulouse	3.0	1.0
3	26/8/2007 2:00:00 - 18 years ago	Murcia	Zaragoza	2.0	1.0
4	18/5/2008 2:00:00 - 17 years ago	Sevilla	Ath Bilbao	4.0	1.0
5	22/8/2009 2:00:00 - 16 years ago	Notts County	Dag and Red	3.0	0.0

Observamos los 5 partidos (pues lo hemos limitado a 5) que cumplen con los requisitos que le hemos señalado.

2.3. Ejercicios sobre pipeline de agregación

2.3.1. Promedios y máximos de goles

Vamos a calcular estadísticas de goles en la colección matches. Agruparemos todos los documentos en uno solo y calcularemos el promedio y máximo de goles anotados por los equipos locales y visitantes. Para ello, haremos uso del siguiente código:

```
// 2.3.1. Promedios y máximos de goles
db.matches.aggregate([
  {
    $group: {
      _id: null,
      avgHomeGoals: { $avg: "$FTHome" },
      avgAwayGoals: { $avg: "$FTAway" },
      maxHomeGoals: { $max: "$FTHome" },
      maxAwayGoals: { $max: "$FTAway" }
    }
  },
  {
    $project: {
      _id: 0,
      avgHomeGoals: 1,
      avgAwayGoals: 1,
      maxHomeGoals: 1,
      maxAwayGoals: 1
    }
  }
])
```

El output es el siguiente:

	avgHomeGoals	avgAwayGoals	maxHomeGoals	maxAwayGoals
1	1.4884575301917031	1.1498900925674551	10.0	13.0

En el encontramos la media el *avgHomeGoals* (1.488), el *avgAwayGoals* (1.149), el *maxHomeGoals* (10) y el *maxAwayGoals* (13). Hemos hecho uso de las funciones *$avg* y *$max* para determinar las medias y los máximos de los datos registrados.

2.3.2. 5 equipos con más victorias en la base de datos

Vamos a obtener los 5 equipos con mayor numero de victorias en nuestra colección *matches,* acompañado del número de empates y de derrotas. Para ello hacemos uso del siguiente código:

```
// 2.3.2. Número total de partidos ganados, empatados y perdidos por cada equipo
db.matches.aggregate([
  {
    $group: {
      _id: "$HomeTeam",
      wins: { $sum: { $cond: [ { $eq: ["$FTResult", "H"] }, 1, 0 ] } },
      draws: { $sum: { $cond: [ { $eq: ["$FTResult", "D"] }, 1, 0 ] } },
      losses: { $sum: { $cond: [ { $eq: ["$FTResult", "A"] }, 1, 0 ] } }
    }
  },
  {
    $project: {
      _id: 0,
      HomeTeam: "$_id",
      wins: 1,
      draws: 1,
      losses: 1
    }
  },
  {
    $sort: { wins: -1 }
  },
  {
    $limit: 5
  }
])
```

Por ello vemos cómo hacemos uso de la función *$group* (para agrupar por equipo) y las funciones *$sum* (para realizar la suma de todas las victorias, los empates y las derrotas). Luego ordenamos por número de victorias, pues considero que es lo más importante, y lo limitamos a 5 para obtener aquellos 5 con mayor número de victorias. El output es el siguiente:

matches | 0.457 s | 5 Docs — 50 — p. 1 — 1 - 5 — Table

	wins	draws	losses	HomeTeam
1	346	58	20	Celtic
2	343	61	41	Real Madrid
3	337	62	43	Barcelona
4	325	90	40	Juventus
5	319	60	31	Bayern Munich

Podemos observar que el equipo con mayor número de victorias es el Celtic (346) seguido del Real Madrid (343), en tercer lugar, el Barcelona (337), en cuarta posición la Juventus (325) y finalmente el Bayer de Múnich (319).