AMPLIACION de BASES DE DATOS

(Profesor: Héctor Gómez Gauchía)

Práctica 5 semana 13: Actualizaciones en una BD en MongoDB

➔ ***QUÉ SUBIR al CV): un .zip con lo siguiente***

-Un SOLO documento PDF que describa en cada apartado:

- la solución,

- el resultado después de la ejecución (basta con que sean algunos documentos que ilustren el efecto)

APARTADO 6.- Trabajando con la colección *aficiones* en MongoDB: Actualizaciones

a) Repasa ejemplos de update e insert: ejemplos-Libro-MongoDB-v3.pdf

b)

b) Arreglar la colección *aficiones*. Usando varias update sucesivas puedes añadir y quitar atributos sin perder sus valores. Arregla los errores más destacados que son:

b.1) MotoGP: cambiar el atributo NombreEquipo por Nombre (sin perder sus valores)

b.2) Precio: En Futbol y Baloncesto hay valores exagerados: quitar ceros hasta dejar cifras de tres dígitos.

b.3) Añadir el precio a MotoGP y Ajedrez con valor fijo de 100 (en un solo *update*)

b.4) En MotoGP: Atributo incorrecto: hay "Puntuacón" donde debería haber "Puntuacion" sin perder su valor

b.5) **(sacar nota)** En Ajedrez : unir los valores de Nombre y Apellidos en el atributo Nombre. Quitar el atributo Apellidos.

c) Esos errores son ocasionados porque la colección no tenía un validador automático. Crea una colección *misAficiones* solo para tu Tema (afición) y define el validador adecuado siguiendo las pautas del método en *validar-con-Schema.js* de la carpeta /ejemplosMongodb-en-Teoria/ . Inserta en *misAficiones* los documentos de tu tema que están en colección *aficiones* (puedes hacerlo copiándolos e insertándolos con la opción del menú *insert*). Deben superar la validación para que se inserten.

Nota: ver otro ejemplo del `$jsonSchema` en https://docs.mongodb.com/manual/core/schema-validation/

d) Queremos evitar repeticiones de datos, ej.: tener el mismo libro repetido para todos los que les guste. Para ello queremos normalizar *aficiones* para tener una colección *solodatos* con la lista de detalles de las *aficiones*. Además, para no estropear *aficiones*, crea otra colección *soloaficiones* con lo que debe quedar en *aficiones* después de normalizarla: quedarán los atributos más frecuentes en las consultas: los Campos Obligatorios del enunciado, y el identificador del nuevo objeto con los detalles en *solodatos*.

e) El efecto del apartado anterior es que ahora la consulta completa uniendo ambas colecciones es más compleja y más lenta. Cómo se hace?

f) **(para nota)** Crea una colección *resconstruida* que sea el resultado de la unión de *solodatos* y *soloaficiones* (el resultado del apartado anterior)

APARTADO 5.- (REVISAR el de la semana pasada con lo explicado en clase de Teoría)

Siguiendo las pautas para diseñar una BD no-sql en las diapositivas de la Teoría: diseña tú una BD de tema libre y describe qué operaciones quiere hacer. Teniendo en cuenta que sea un tema donde una BD tipo SQL no sea adecuada.

APARTADO 5.- EXTRA (REVISAR el de la semana pasada con lo explicado en clase de Teoría)

Deseamos introducir elementos compuestos, ej.: como en un equipo de futbol si incluimos cada jugador con sus datos personales. Y queremos hacer muchas consultas sobre esos elementos compuestos ej.: datos personales. ¿Conviene normalizar o desnormalizar?. ¿Cómo debería quedar la representación de la colección?