DAW/DAM. UD 8. BASES DE DATOS NOSQL: MONGODB. ACTIVIDADES NO EVALUABLES PARTE 2 (SOLUCIONADO)

DAW/DAM. Bases de datos (BD)

UD 8. BASES DE DATOS NOSQL: MONGODB

Parte 2. DML. Prácticas no evaluables (solucionado)

Abelardo Martínez y **Pau Miñana**
Basado y modificado de Sergio Badal (www.sergiobadal.com)
Curso 2023-2024

Aspectos a tener en cuenta

Importante

Estas actividades son opcionales y no evaluables pero es recomendable hacerlas para un mejor aprendizaje de la asignatura.

> **Si buscas las soluciones por Internet o preguntas al oráculo de ChatGPT, te estarás engañando a ti mismo**. Ten en cuenta que **ChatGPT no es infalible ni todopoderoso**.
>
> Es una gran herramienta para agilizar el trabajo una vez se domina una materia, pero usarlo como atajo en el momento de adquirir habilidades y conocimientos básicos perjudica gravemente tu aprendizaje. Si lo utilizas para obtener soluciones o asesoramiento respecto a las tuyas, revisa cuidadosamente las soluciones propuestas igualmente. Intenta resolver las actividades utilizando los recursos que hemos visto y la documentación extendida que encontrarás en el “Aula Virtual”.

Recomendaciones

Importante

- **No uses NUNCA tildes, ni eñes, ni espacios, ni caracteres no alfanuméricos** (salvo el guión bajo) **en los metadatos** (nombres de elementos de una base de datos).
- Sé coherente con el uso de mayúsculas/minúsculas.

1. Colección Películas

Actividad no evaluable

Utiliza la base de datos **pruebas** para crear la colección "**peliculas**".

1.1. Ejercicio

Crea una nueva colección llamada "peliculas" e inserta los siguientes datos:

- **_id**. Identificador único de la película (predefinido)
- **titulo**. Denominación comercial de la película en España
- **anyo**. Fecha de estreno (solo el año)
- **recauda**. Millones de euros recaudados
- **cuesta**. Millones de euros de presupuesto
- **saldo**. Diferencia entre RECAUDA-CUESTA
- **rentabilidad**. Porcentaje de beneficio obtenido
- **ranking**. Campo para aplicar un ranking u otro
- **cod_saga**. Código de la saga a la que pertenece (si no hay valor, no debe existir el campo)

_id	titulo	anyo	recauda	cuesta	saldo	rentabilidad	ranking	cod_saga
789	Divergente	2014	288,7	85	203,7	340	0	DIVER
963	Insurgente	2015	297,2	110	187,2	270	0	DIVER
874	Leal	2016	179,2	110	69,2	163	0	DIVER
151	El planeta de los simios	1968	32,5	5,4	27,1	602	0	SIMIOS1
666	El Exorcista	1973	441,3	12				
215	Regreso al planeta de los simios	1970	18,9	3	15,9	630	0	SIMIOS1
687	Huida del planeta de los simios	1971	12,3	2,5	9,8	492	0	SIMIOS1
278	La rebelión de los simios	1972	9,7	1,7	8	571	0	SIMIOS1
987	La batalla por el planeta de los simios	1973	8,8	1,8	7	489	0	SIMIOS1
345	El planeta de los simios	2001	362,2	100	262,2	362	0	
428	El origen del planeta de los simios	2011	481,8	93	388,8	518	0	SIMIOS2
887	El amanecer del planeta de los simios	2014	707,4	170	537,4	416	0	SIMIOS2
587	La guerra del planeta de los simios	2017	490,7	150	340,7	327	0	SIMIOS2

Solución

```
//conectar con la base de datos y borrar colección
use pruebas
db.peliculas.drop()

//poblar la base de datos
db.createCollection('peliculas')

// crear varios documentos
var j_item1 = {_id: 789, titulo: 'Divergente', anyo: 2014, recauda: 288.7, cuesta
var j_item2 = {_id: 963, titulo: 'Insurgente', anyo: 2015, recauda: 297.2, cuesta
var j_item3 = {_id: 874, titulo: 'Leal', anyo: 2016, recauda: 179.2, cuesta: 110,
var j_item4 = {_id: 151, titulo: 'El planeta de los simios', anyo: 1968, recauda:
var j_item5 = {_id: 215, titulo: 'Regreso al planeta de los simios', anyo: 1970,
var j_item6 = {_id: 687, titulo: 'Huida del planeta de los simios', anyo: 1971, r
var j_item7 = {_id: 278, titulo: 'La rebelión de los simios', anyo: 1972, recauda
var j_item8 = {_id: 987, titulo: 'La batalla por el planeta de los simios', anyo:
var j_item9 = {_id: 345, titulo: 'El planeta de los simios', anyo: 2001, recauda:
var j_item10 = {_id: 428, titulo: 'El origen del planeta de los simios', anyo: 20
var j_item11 = {_id: 887, titulo: 'El amanecer del planeta de los simios', anyo:
var j_item12 = {_id: 587, titulo: 'La guerra del planeta de los simios', anyo: 20
var j_item13 = {_id: 666, titulo: 'El exorcista', anyo: 1973, recauda: 441.3, cue

// insertar los documentos en la colección
db.peliculas.insertMany([j_item1, j_item2, j_item3, j_item4, j_item5, j_item6, j_
```

1.2. Ejercicio

- Actualiza los valores de la película con **_id** 666 a un **saldo** de 429.3 y una **rentabilidad** del 3678% (omitiendo el símbolo de %).
- Muestra esa película (**_id+titulo+anyo+saldo+rentabilidad**) después, para ver el resultado.

Solución

```
//actualizar
var j_filtro = {_id: 666}
var j_set    = {saldo: 429.3, rentabilidad: 3678}
var j_accion = {$set:j_set}
db.peliculas.updateMany(j_filtro, j_accion)
//consultar
var j_proyeccion = {titulo:1, anyo:1, saldo:1, rentabilidad:1}
db.peliculas.find(j_filtro, j_proyeccion)
```

1.3. Ejercicio

- Actualiza el **ranking** de cada película fijando un 5 si tiene una **rentabilidad** igual o superior a 500 (500%), un 4 si tiene una rentabilidad inferior a 500 e igual o superior a 400 y así hasta ranking 1.
- Muestra todas las películas (**titulo+anyo+rentabilidad+ranking**) después, ordenadas por ranking y rentabilidad descendentes para ver el resultado.

Solución

```
//actualizar
var j_valor  = {$gte: 500}
var j_filtro = {rentabilidad: j_valor}
var j_set    = {ranking:5}
var j_accion = {$set:j_set}
db.peliculas.updateMany(j_filtro, j_accion)

var j_valor  = {$lt: 500}
var j_filtro = {rentabilidad: j_valor}
var j_set    = {ranking:4}
var j_accion = {$set:j_set}
db.peliculas.updateMany(j_filtro, j_accion)

var j_valor  = {$lt: 400}
var j_filtro = {rentabilidad: j_valor}
var j_set    = {ranking:3}
var j_accion = {$set:j_set}
db.peliculas.updateMany(j_filtro, j_accion)

var j_valor  = {$lt: 300}
var j_filtro = {rentabilidad: j_valor}
var j_set    = {ranking:2}
var j_accion = {$set:j_set}
db.peliculas.updateMany(j_filtro, j_accion)
```

```
var j_valor  = {$lt: 200}
var j_filtro = {rentabilidad: j_valor}
var j_set    = {ranking:1}
var j_accion = {$set:j_set}
db.peliculas.updateMany(j_filtro, j_accion)
//consultar
var j_proyeccion = {titulo:1, anyo:1, rentabilidad:1, ranking:1, _id:0}
var j_filtro     = {}
var j_orden      = {ranking:-1, rentabilidad:-1}
db.peliculas.find(j_filtro, j_proyeccion).sort(j_orden)
```

1.4. Ejercicio

- Haz tres copias de la colección original "**peliculas**" y llámalas SAGA_DIVER, SAGA_SIMIOS1, SAGA_SIMIOS2. Utiliza: db.ORIGINAL.aggregate([{$out: "COPIA"}])
- Luego, borra en cada copia las películas que O BIEN tienen el campo **cod_saga** y no son de esa saga, O BIEN no tienen el campo cod_saga.
- Elimina TODOS los campos de esas tres copias excepto **_id**, **titulo** y **anyo**. Utiliza: $unset
- Muestra todas las películas (**TODOS LOS CAMPOS**) de esas tres colecciones después, ordenadas por año ascendente para ver el resultado.

Solución

```
//realizar copias
db.peliculas.aggregate([{$out: 'SAGA_DIVER'}])
db.peliculas.aggregate([{$out: 'SAGA_SIMIOS1'}])
db.peliculas.aggregate([{$out: 'SAGA_SIMIOS2'}])

//SAGA_DIVER
//borrar
var j_valor1                = {$exists: true}
var j_filtro1               = {cod_saga: j_valor1}
var j_valor2                = {$ne: 'DIVER'}
var j_filtro2               = {cod_saga: j_valor2}
var j_filtroExiste_y_Valor  = {$and:[j_filtro1, j_filtro2]}
var j_valor3                = {$exists: false}
var j_filtroNoExiste        = {cod_saga: j_valor3}
var j_filtroFinal           = {$or:[j_filtroNoExiste, j_filtroExiste_y_Valor]}
db.SAGA_DIVER.deleteMany(j_filtroFinal)
//actualizar
db.SAGA_DIVER.updateMany({}, {$unset: {cod_saga: "", recauda: "", cuesta: "", sal
//consultar
var j_filtro = {}
var j_orden  = {anyo:1}
```

```
db.SAGA_DIVER.find(j_filtro).sort(j_orden)

//SAGA_SIMIOS1
//borrar
var j_valor1                = {$exists: true}
var j_filtro1               = {cod_saga: j_valor1}
var j_valor2                = {$ne: 'SIMIOS1'}
var j_filtro2               = {cod_saga: j_valor2}
var j_filtroExiste_y_Valor  = {$and:[j_filtro1, j_filtro2]}
var j_valor3                = {$exists: false}
var j_filtroNoExiste        = {cod_saga: j_valor3}
var j_filtroFinal           = {$or:[j_filtroNoExiste, j_filtroExiste_y_Valor]}
db.SAGA_SIMIOS1.deleteMany(j_filtroFinal)
//actualizar
db.SAGA_SIMIOS1.updateMany({}, {$unset: {cod_saga: "", recauda: "", cuesta: "", s
//consulta
var j_filtro = {}
var j_orden  = {anyo:1}
db.SAGA_SIMIOS1.find(j_filtro).sort(j_orden)

//SAGA_SIMIOS2
//borrar
var j_valor1                = {$exists: true}
var j_filtro1               = {cod_saga: j_valor1}
var j_valor2                = {$ne: 'SIMIOS2'}
var j_filtro2               = {cod_saga: j_valor2}
var j_filtroExiste_y_Valor  = {$and:[j_filtro1, j_filtro2]}
var j_valor3                = {$exists: false}
var j_filtroNoExiste        = {cod_saga: j_valor3}
var j_filtroFinal           = {$or:[j_filtroNoExiste, j_filtroExiste_y_Valor]}
db.SAGA_SIMIOS2.deleteMany(j_filtroFinal)
//actualizar
db.SAGA_SIMIOS2.updateMany({}, {$unset: {cod_saga: "", recauda: "", cuesta: "", s
//consultar
var j_filtro = {}
var j_orden  = {anyo:1}
db.SAGA_SIMIOS2.find(j_filtro).sort(j_orden)
```

1.5. Ejercicio

Crea 2 nuevas colecciones duplicando las originales, una llamada "**mas_rentables**" para las películas con un 5 de **ranking** y otra llamada "**menos_rentables**" que incluya las películas con un 1 de **ranking**.

a) Colección "**más_rentables**":

- Crea una nueva colección, duplicando la original, llamada "**mas_rentables**".
- Muestra todas las películas (**titulo+anyo+rentabilidad+ranking**) de esa colección cuyo **ranking** es distinto de 5, ordenadas por ranking descendente.
- Elimina todas las películas cuyo ranking es distinto de 5.
- Muestra todas las películas (**titulo+anyo+rentabilidad+ranking**) después, ordenadas por ranking descendente para ver el resultado.

b) Colección "**menos_rentables**":

- Crea una nueva colección, duplicando la original, llamada "**menos_rentables**".
- Muestra todas las películas (**titulo+anyo+rentabilidad+ranking**) de esa colección cuyo **ranking** es distinto de 1, ordenadas por ranking descendente.
- Elimina todas las películas cuyo ranking es distinto de 1.
- Muestra todas las películas (**titulo+anyo+rentabilidad+ranking**) después, ordenadas por ranking descendente para ver el resultado.

Solución

```
//MAS_RENTABLES
//realizar copia
db.peliculas.aggregate([{$out: "MAS_RENTABLES"}])
//borrar
var j_valor1                = {$exists: true}
var j_filtro1               = {ranking: j_valor1}
var j_valor2                = {$ne: 5}
```

```
var j_filtro2                = {ranking: j_valor2}
var j_filtroExiste_y_Valor   = {$and:[j_filtro1, j_filtro2]}
var j_valor3                 = {$exists: false}
var j_filtroNoExiste         = {ranking: j_valor3}
var j_filtroFinal            = {$or:[j_filtroNoExiste, j_filtroExiste_y_Valor]}
var j_proyeccion             = {titulo:1, anyo:1, rentabilidad:1, ranking:1, _id:0
var j_orden                  = {ranking:-1}
db.MAS_RENTABLES.find(j_filtroFinal, j_proyeccion).sort(j_orden)
db.MAS_RENTABLES.deleteMany(j_filtroFinal)
//consultar
var j_filtro = {}
db.MAS_RENTABLES.find(j_filtro, j_proyeccion).sort(j_orden)

//MENOS_RENTABLES
//realizar copia
db.peliculas.aggregate([{$out: "MENOS_RENTABLES"}])
//borrar
var j_valor1                 = {$exists: true}
var j_filtro1                = {ranking: j_valor1}
var j_valor2                 = {$ne: 1}
var j_filtro2                = {ranking: j_valor2}
var j_filtroExiste_y_Valor   = {$and:[j_filtro1, j_filtro2]}
var j_valor3                 = {$exists: false}
var j_filtroNoExiste         = {ranking: j_valor3}
var j_filtroFinal            = {$or:[j_filtroNoExiste, j_filtroExiste_y_Valor]}
var j_proyeccion             = {titulo:1, anyo:1, rentabilidad:1, ranking:1, _id:0
var j_orden                  = {ranking:-1}
db.MENOS_RENTABLES.find(j_filtroFinal, j_proyeccion).sort(j_orden)
db.MENOS_RENTABLES.deleteMany(j_filtroFinal)
//consultar
var j_filtro = {}
db.MENOS_RENTABLES.find(j_filtro, j_proyeccion).sort(j_orden)
```

1.6. Ejercicio

Busca en la red una saga que te guste que tenga EXACTAMENTE 3 PELÍCULAS e inserta sus películas en la colección original. Luego, repite el proceso anterior para asignarles un valor al campo **ranking** y tener una nueva colección solo con esas películas y sin el campo ranking.

Muestra todas las películas (**titulo+anyo+cod_saga**) de esa nueva colección después, ordenadas por año ascendente para ver el resultado.

Solución

```
//insertar datos
var j_item1 = {_id: 111, titulo: 'The Terminator', anyo: 1984, recauda: 78.48, cu
var j_item2 = {_id: 121, titulo: 'Terminator 2: el juicio final', anyo: 1991, rec
var j_item3 = {_id: 131, titulo: 'Terminator 3: la rebelión de las máquinas', any
// inserta esos documentos en la colección
db.peliculas.insertMany([j_item1, j_item2, j_item3])
//duplicar la colección
db.peliculas.aggregate([{$out: "SAGA_TERMINATOR"}])
//eliminar las películas que no son de la nueva saga de Terminator
var j_valor  = {$ne: 'TERMINATOR'}
var j_filtro = {cod_saga: j_valor}
db.SAGA_TERMINATOR.deleteMany(j_filtro)
//eliminar campos
var j_set    = {recauda:"", cuesta:"", saldo:"", rentabilidad:"", ranking:"", cod
var j_accion = {$unset:j_set}
db.SAGA_TERMINATOR.updateMany({}, j_accion)
//consultar
var j_proyeccion = {titulo:1, anyo:1, _id:0}
var j_filtro     = {}
var j_orden      = {anyo:1}
db.SAGA_TERMINATOR.find(j_filtro, j_proyeccion).sort(j_orden)
```

2. Bibliografía

- ¿Qué es SQL y NoSQL? [Platzi]. https://www.youtube.com/watch?v=CuAYLX6reXE
- NO SQL: como se modelan las bbdd no relacionales? [HolaMundo]. https://www.youtube.com/watch?v=Zdlude8l8w4
- El concepto NoSQL, o cómo almacenar tus datos en una base de datos no relacional. https://www.genbeta.com/desarrollo/el-concepto-nosql-o-como-almacenar-tus-datos-en-una-base-de-datos-no-relacional
- Metodologías ágiles Scrum, Kanban 04 Triángulo de hierro. https://www.youtube.com/watch?v=PdzW4G_hbsw
- Proyectos ágiles. Triángulo de hierro. https://proyectosagiles.org/triangulo-hierro/
- Teorema CAP. Píldoras de conocimiento. https://www.youtube.com/watch?v=Ydv-y_oH_CY
- Una introducción a MongoDB. https://www.genbeta.com/desarrollo/una-introduccion-a-mongodb
- MongoDB: qué es, cómo funciona y cuándo podemos usarlo (o no). https://www.genbeta.com/desarrollo/mongodb-que-es-como-funciona-y-cuando-podemos-usarlo-o-no
- Tutorial gratuito de 30 vídeos. https://www.youtube.com/watch?v=nlOWsnO-d7Q&list=PLXXiznRYETLcJE_4U9qN2pysZOSYyL4Mh