[2 PUNTOS] CREAR Y POBLAR LA BASE DE DATOS

```
//Limpiamos pantalla
Cls
//Conectamos con la BD y la borramos
use MDBElecciones
db.dropDatabase()
use MDBElecciones
// Creamos una nueva colección llamada “Partidos”
db.createCollection("Partidos")
// Creamos los diferentes documentos dentro de Partidos
var partidos01 = {_id:"p01", siglas: "ASIR", nombre: "Alianza Social de Izquierda Radical",
    votos : [
        {id_dig: "F6E8D2C", circunscripcion: 'Impunícia', gratificacion: 500},
        {id_dig: "H5I4J3K", circunscripcion: "Impunícia"},
        {id_dig: "3A7F9B4", circunscripcion: "Impunícia", gratificacion: 1000},
        {id_dig: "U7T9S5R", circunscripcion: "Impunícia"}]
}
var partidos02 = {_id: "p02", siglas: "DAM", nombre: "Democracia, Acción y Modernidad",
    votos: [
        {id_dig: "5X7W9Y4", circunscripcion: "Impunícia"},
        {id_dig: "P1Q3R5S", circunscripcion: "Ladronia"},
        {id_dig: "C1D2E3F", circunscripcion: "Ladronia"},
        {id_dig: "G6H7I8J", circunscripcion: "Ladronia"},
        {id_dig: "Y2X4Z6W", circunscripcion: "Ladronia"},
        {id_dig: "Z5A4B3C", circunscripcion: "Impunícia"},
        {id_dig: "J5K4L3M", circunscripcion: "Ladronia"}],
    afiliados: 6000
 }
var partidos03 = {_id: "p03", siglas: "DAW", nombre: "Diversidad, Ambiente y Wifi gratis",
    afiliados: 4000
}
var partidos04 = {_id: "p04", siglas: "PHT", nombre: "Partido de la Honestidad y la Transparencia",
    votos: [
        {id_dig: "1A2B3C4", circunscripcion: "Impunícia", gratificacion: 1500},
        {id_dig: "7G8H9I0", circunscripcion: "Ladronia", gratificacion: 1000},
        {id_dig: "J1KL3M", circunscripcion: "Impunícia"},
        {id_dig: "R7S8T9U", circunscripcion: "Ladronia", gratificacion: 1200},
        {id_dig: "4D5E6F7", circunscripcion: "Ladronia"},
        {id_dig: "A1B2C3D", circunscripcion: "Impunícia", gratificacion: 700},
        {id_dig: "N4O5P6Q", circunscripcion: "Ladronia", afiliados: 800},
        {id_dig: "V1W2X3Y", circunscripcion: "Impunícia"}],
     afiliados: 12000
}
```

```
var partidos05 = {_id: "p05", siglas: "PRO", nombre: "Partido del Robo Organizado",
    votos: [
        {id_dig: "Z1X4C7V", circunscripcion: "Impunícia", gratificacion: 100},
        {id_dig: "O1P2Q3R", circunscripcion: "Impunícia", gratificacion: 50},
        {id_dig: "S7T8U9V", circunscripcion: "Impunícia", gratificacion: 200},
        {id_dig: "K0L9M8N", circunscripcion: "Ladronia", gratificacion: 1000},
        {id_dig: "T4U5V6W", circunscripcion: "Impunícia", gratificacion: 100},
        {id_dig: "X1Y2Z3A", circunscripcion: "Ladronia", gratificacion: 1500}],
    afiliados: 8000
}

// Insertamos los documentos en la colección

db.Partidos.insertMany([partidos01, partidos02, partidos03, partidos04, partidos05]);

// Listamos los campos de todos los registros ordenados por las siglas

db.Partidos.find().sort({siglas:1});
```

[2 PUNTOS] ACTUALIZAR LA INFORMACIÓN

1)

Para actualizar el campo "subvención" en el caso de Partidos con menos de 5.000 afiliados donde la subvención es de 1.000.000€.

/** Creamos las variables como condiciones de búsqueda con operadores de consulta, true para encontrar aquellos que tengan afiliados y $lt:5000 para encontrar los documentos de la colección donde se den los Partidos con menos de 5000 afiliados, refiriéndose al documento afiliados, y por ultimo también hemos creado el filtro donde se den las dos condiciones con el operador $and donde las dos condiciones juntas se cumplan. La operación $set indicará que el campo subvención debe ser dado en ese valor en los documentos incluidos por la operación de actualización*//

```
var j_condicion1 = {$exists: true}
var j_condicion2 = {$lt: 5000}
var j_afiliados1 = {afiliados: j_condicion1}
var j_afiliados2 = {afiliados: j_condicion2}
var filtro = { $and:[j_afiliados1, j_afiliados2]}
var accion = { $set: {subvencion: 1000000}}

// Actualizamos con el filtro y la acción.
db.Partidos.updateMany (filtro , accion)
db.Partidos.updateMany ({afiliados:[$exists: true, $lt: 5000], $set: {subvencion: 1000000})
```

Partidos entre 5.000 y 10.000, la subvención es de 3.000.000€.

/**Se crean las variables con las condiciones de búsqueda con operadores de consulta, true para encontrat aquellos que tengan afiliados y $sgte:5000 para encontrar los documentos que son mayores o igual de 5000 afiliados y menor o igual que 10000 afiliados, haciendo referencia al documento afiliados y por último creamos el filtro donde se dan las 3 condiciones en este caso con el operador $and donde se cumplan las mismas 3 condiciones. La operación $set indicará que el campo subvención debe ser dado en ese valor en los documentos incluidos por la operación de actualización *//

```
var j_condicion1 = {$exists: true}
var j_condicion2 = {$gte: 5000}
var j_condicion3 = {$lte:10000}
var j_afiliados1 = {afiliados: j_condicion1}
var j_afiliados2 = {afiliados: j_condicion2}
var j_afiliados3 = {afiliados: j_condicion3}
var j_filtro = { $and:[j_afiliados1, j_afiliados2, j_afiliados3]}
var j_accion = { $set: {subvencion: 3000000}}
// Actualizamos con el filtro y la acción.
db.Partidos.updateMany (j_filtro , j_accion)

db.Partidos.updateMany({afiliados: {$exists: true, $gte: 5000, $lte:10000}},{$set: {subvencion :
3000000}})
```

Partidos con más de 10.000 afiliados, la subvención es de 10.000.000€.

/**Se crean las variables con las condiciones de búsqueda con operadores de consulta, true para encontrar aquellos que tengan afiliados y $sgte:5000 para encontrar los documentos que son mayores a 10000 afiliados, haciendo referencia al documento afiliados y por último creamos el filtro donde se dan las 2 condiciones en este caso con el operador $and donde se cumplan las 3 condiciones. La operación $set indicará que el campo subvención debe ser dado en ese valor en los documentos incluidos por la operación de actualización *//

```
var j_condicion1 = {$exists: true}
var j_condicion2 = {$gt: 10000}
var j_afiliados1 = {afiliados: j_condicion1}
var j_afiliados2 = {afiliados: j_condicion2}
var j_filtro = { $and:[j_afiliados1, j_afiliados2]}
var j_accion = { $set: {subvencion: 10000000}}
// Actualizamos con el filtro y la acción.
db.Partidos.updateMany (j_filtro , j_accion)
```

```
db.Partidos.updateMany({afiliados: {$exists: true, $gt: 10000}},{$set: {subvención : 10000000}}).pretty()
```

Los partidos que no presentan su número de afiliados no recibirán subvención (0€).

/**Creamos también las variables con las condiciones de búsqueda con operadores de consulta, false para encontrar aquellos que no tengan afiliados en el Partido haciendo referencia al campo afiliados y por último creamos el filtro donde se da la condición. La operación $set indicará que el campo subvención debe ser dado en ese valor 0 donde no hay afiliados en los documentos incluidos por la operación de actualización *//

```
var j_condicion = {$exists: false}
var j_filtro = {afiliados: j_condicion}
var j_accion = {$set: {subvencion: 0}}
// Actualizamos con el filtro y la acción.
db.Partidos.updateMany (j_filtro , j_accion)
```

```
db.Partidos.updateMany({afiliados: {$exists: false}},{$set: {subvencion : 0}}).pretty()
```

2) Listar el nombre del partido y el valor de la subvención, ordenados por el valor de la subvención de mayor a menor.

Listamos todos los Partidos donde aparecerá inicialmente el nombre del partido (1) con su valor de subvención (1), incluyendo el campo _id que no se puede dejar de mencionar, ordenados (sort) por el valor de la subvención de mayor a menor (-1)

var j_proyeccion = {nombre:1, subvencion:1, _id:0}
var j_orden = {subvencion:-1}

// Muestra los documentos que cumplen la proyección y el orden.
db.Partidos.find({}, j_proyeccion).sort(j_orden).pretty()

db.Partidos.find({}, { nombre: 1, subvención: 1, _id: 0}).sort({subvencion: -1}).pretty()

```
MDBElecciones> db.Partidos.find({}, j_proyeccion).sort(j_orden).pretty()
[
  {
    nombre: 'Partido de la Honestidad y la Transparencia',
    'subvención': 10000000
  },
  { nombre: 'Democracia, Acción y Modernidad', 'subvención': 3000000 },
  { nombre: 'Partido del Robo Organizado', 'subvención': 3000000 },
  {
    nombre: 'Diversidad, Ambiente y Wifi gratis',
    'subvención': 1000000
  },
  { nombre: 'Alianza Social de Izquierda Radical', 'subvención': 0 }
]
```

[3 PUNTOS] MINERÍA DE DATOS

1)/**Utilizamos primero unwind para separar los elementos del array. Con el filtro especificamos que la gratificación sea superior a 500. Luego filtramos los documentos basándonos en el filtro con el $match. Definimos los campos que se piden en el ejercicio que se muestren en el resultado en la proyección, excluimos el campo _id ya que se considera buena praxis. Luego se proyectan los campos definidos en j_proy, donde luego definimos el orden, primero por las siglas en orden ascendente(1) y luego por las gratificaciones en orden ascendente(1).*//

```
var stage_separar = {$unwind:"$votos"}
var j_filtro = {"votos.gratificacion":{$gt:500}}
var stage_filtro= {$match:j_filtro}
var j_proy = {siglas:1, "votos.id_dig":1, "votos.circunscripcion":1, "votos.gratificacion":1, _id:0}
var stage_proy = {$project: j_proy}
var orden = {siglas:1, "votos.gratificacion":1}
var stage_sort = {$sort: orden}
//Ejecutamos la agregación con los estados definidos anteriormente.
db.Partidos.aggregate([stage_separar, stage_filtro, stage_proy, stage_sort]).pretty()
```

2)

/**Seguimos utilizando la variable para separar los votos con unwind, nombramos una variable j_cuenta que englobe el grupo a considerar donde está el campo siglas renombrado como _id, y añadimos un campo totalGastado que suma las gratificaciones de los votos por cada partido. Agrupamos los documentos por siglas de partido y el total gastado con la suma $sum de cada campo en todos los documentos. A continuación ordenamos los resultados por el campo totalGastado de manera descendente(-1) para que aparezca primero el partido con mas gratificaciones, luego limitamos los resultados a 1 ($limit:1) para que solo devuelva el partido con mas gratificaciones. Luego en la proyección seleccionamos los campos que den como resultado las siglas y el total gastado para luego proyectarlo en la variable stage_proy, renombrando el campo _id como siglas

```
var stage_separar = {$unwind:"$votos"}
var j_cuenta = {"_id": "$siglas", totalGastado: {$sum: "$votos.gratificacion"}}
var stage_group = {$group: j_cuenta}
var stage_sort = {$sort: {totalGastado: -1}}
var stage_limit = {$limit:1}
var j_proy = {siglas: "$_id", totalGastado:1, _id: 0}
var stage_proy = {$project: j_proy}
// Ejecutamos la agregación con las fases anteriores.
db.Partidos.aggregate([stage_separar, stage_group, stage_sort, stage_limit, stage_proy])
```

```
MDBElecciones> db.Partidos.aggregate([stage_separar, stage_group, stage_sort, s
tage_limit, stage_proy])
[ { totalGastado: 4400, siglas: 'PHT' } ]
```

[2 PUNTOS] CONTADORES

Crea el script necesario para contar el numero de partidos que no tienen afiliados o teniendo votos no han pagado ninguna gratificación.

/** Utilizando documentos embebidos seleccionamos los partidos que no tienen el campo afiliados como condición 1, como condicion2 seleccionamos con el operador $and donde los partidos tienen votos y la gratificación es igual a null ya que el enunciado requiere que no hayan pagado ninguna gratificación. Combinamos las condiciones de filtro con $or y de este modo pasara el filtro un partido si cumple al menos una de las dos condiciones. En la etapa de agregación utilizamos el $match para filtrar los documentos de la colección de partidos según lo solicitado en el filtro. Y por ultimo utilizamos el $count para contar el numero total de partidos que cumplen el filtro.*/

```
var j_condicion1 = {afiliados:{$exists: false}}
var j_condicion2 ={$and:[{"votos":{$exists: true}}, {"votos.gratificacion":{$exists: true, $eq:null}}]}
var j_filtro = {$or: [j_condicion1, j_condicion2]}
var stage_filtro = {$match: j_filtro}
var stage_cuenta = {$count: "partidosTotal"}

//Ejecución de la agregación en la colección.
db.Partidos.aggregate([stage_filtro, stage_cuenta])
```

```
MDBElecciones> db.Partidos.aggregate([stage_filtro, stage_cuenta])
[ { partidosTotal: 2 } ]
```

[1 PUNTO] BORRADO

1) Crea el script necesario para listar las siglas y nombres de los partidos sin votos o que no han sido votados en la circunscripción de "Ladronia".

/** Creamos las condiciones en las variables con documentos embebidos sobre los partidos sin votos({votos:{$exists:false}}) y la condición2 para seleccionar los partidos que no han sido votados en la circ. de "Ladronia". La variable j_filtro combina las condiciones con un $or, pasando el documento que cumpla al menos una de las dos condiciones. En la variable stage_filtro se aplican las condiciones del filtro a la colección de Partidos. Definimos la variable j_proy para definir los campos que necesitamos listar de las siglas y nombres en el resultado, excluyendo _id como buena praxis. En la proyección (stage_proy) proyecta los campos definidos en j_proy*/

```
var j_condicion1 = {votos:{$exists:false}}
var j_condicion2 = {"votos.circunscripcion":{$ne: "Ladronia"}}
var j_filtro = {$or:[j_condicion1, j_condicion2]}
var stage_filtro = {$match: j_filtro}
var j_proy = { siglas: 1, nombre: 1, _id: 0}
var stage_proy = {$project: j_proy}

//Ejecución de la agregación en la colección.
db.Partidos.aggregate([stage_filtro, stage_proy]).pretty()
```

```
MDBElecciones> db.Partidos.aggregate([stage_filtro, stage_proy]).pretty()
[
  { siglas: 'ASIR', nombre: 'Alianza Social de Izquierda Radical' },
  { siglas: 'DAW', nombre: 'Diversidad, Ambiente y Wifi gratis' }
]
```

2) Crea el script necesario para borrar los partidos del listado anterior.

/** Utilizamos documentos embebidos para las condiciones y poder referenciar el listado anterior. Luego pasamos el filtro igualmente que cumpla el listado anterior. Y luego ejecutamos el borrado en la colección*/

```
var j_condicion1 = {votos:{$exists:false}}
var j_condicion2 = {"votos.circunscripcion":{$ne: "Ladronia"}}
var j_filtro = {$or:[j_condicion1, j_condicion2]}

//Ejecución del borrado en la colección.
db.Partidos.deleteMany(filtro)
```

```
MDBElecciones> db.Partidos.deleteMany(filtro)
{ acknowledged: true, deletedCount: 1 }
```