

Dataset Meteorite Landings

Bases de datos NoSQL

Dennis Simon Wolter. 12/04/2022

Introducción

Este es un informe que recopila los apartados de este mini proyecto para poner en uso mis conocimientos en bases de datos NO SQL en MongoDB. Mi intención es realizar un caso práctico con un Dataset, realizar diversas consultas y sacar una conclusión a partir de los resultados. Esta herramienta nos permite, a diferencia de otros softwares, trabajar de forma más rápida y flexible, ya que escala muy cómodamente con los grandes conjuntos de datos.

El conjunto de datos con los que vamos a trabajar proviene de [GitHub](#), que recopila parte de los asteroides caídos durante la historia. Estos datos están facilitados por la NASA y los cuales registraran la localización de impacto, la masa, composición y año de caída entre otros. Algunas de las entradas contienen datos faltantes o incorrecciones en los análisis de la propia NASA como, por ejemplo, las coordenadas latitud y longitud cero (oeste africano), donde es más difícil avistarlos. Lo mismo pasa con la zona de la Antártida.

Cada meteorito se compone de las siguientes categorías:

- **name:** Nombre del meteorito (suele ser la zona donde se cae, modificado con su composición, año, etc.).
- **id:** identificador único.
- **nametype:** variable categórica que tiene dos posibles valores:
 - *valid*: meteorito clásico.
 - *relict*: meteorito que ha sufrido alta degradación climatológica terrestre.
- **recclass:** la clase del meteorito; un número basado en la composición física, química y otras propiedades. Este [artículo](#) en Wikipedia los recopila.
- **mass:** masa del meteorito en gramos.
- **fall:** determina si el meteorito se vio caer, o fue descubierto tras el impacto:
 - *fell*: el meteorito fue avistado antes del impacto.
 - *found*: el meteorito fue descubierto tras impacto.
- **year:** año en el que se avistó o encontró el meteorito (depende el valor del campo anterior).
- **reclat:** latitud donde aterrizó el meteorito.
- **reclong:** longitud donde aterrizó el meteorito.

- **Geolocation:** combina las coordenadas de latitud y longitud en forma de tupla.

Cargar/importar un Dataset

Una vez encontrado un conjunto de datos interesante, debemos comprobar si este puede descargarse en formato json. Gracias a las nuevas funcionalidades de Mongo, podemos importar el fichero de forma sencilla (sin tener que acudir a la consola del sistema) en Import ➊ Import from JSON and CSV files... ➋ Add file y seleccionar la inserción *'Drop collection first if it exists'* para que elimine la colección si ya existe, y así realizar una importación limpia.

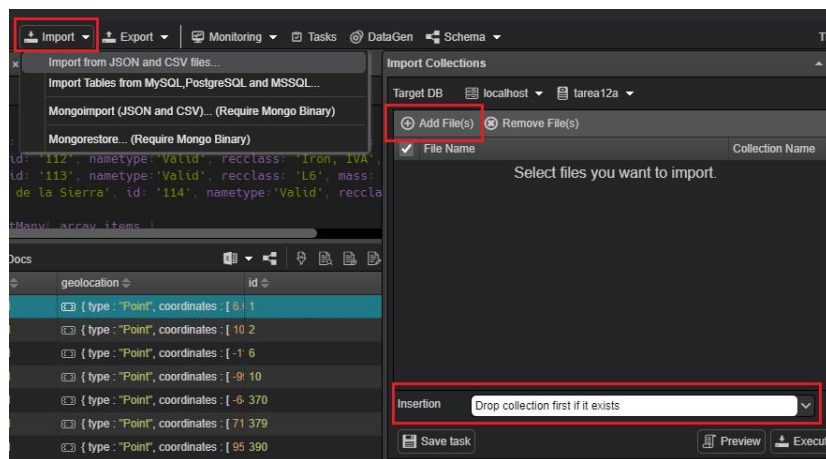


Ilustración 1. Importar un Dataset

Consultas sobre los datos

Antes de consultar los datos, vamos a ver cómo podríamos nosotros insertar nuestras propias entradas y actualizar sus campos. Estas inserciones y actualizaciones los realizo en una colección aparte con el fin de no alterar los resultados del siguiente apartado.

Vamos a insertar primero un único objeto con todos los campos sin nulos y en el mismo formato que en el Dataset original.

```
var nuevo_item = {
  name: 'Madrid',
  id: '110',
  nametype: 'Valid',
  recclass: 'Iron',
  mass: '1000',    fall:
  'Fell',    year: '2005-
01-01T00:00:00.000',
```

```
reclat: '45',    reclong: '45',    geolocation:
{type:'Point', coordinates: [45,45]} }
```

```
db.meteorite2.insertOne( nuevo_item )
```

Key	Value	Type
4 (1)	{ acknowledged : true, insertedId : ObjectId("6255ae25f123e6f9c40288c4") }	Object
acknowledged	true	Bool
insertedId	6255ae25f123e6f9c40288c4	ObjectId

Ilustración 2. Insert de un objeto

Insertar de uno en uno no es la forma más productiva de rellenar una base de datos, es por eso por lo que nos puede interesar insertar un array de objetos. (Con el objetivo de no rellenar páginas con código redundante se escribirán solo las primeras líneas).

```
var array_items= [
    { name: 'Paris', id: '111', recclass: 'Iron', ... },
    { name: 'Londres', id: '112', recclass: 'L6', ... },
    { ... }]
```

```
db.meteorite2.insertMany( array_items )
```

Una vez insertados ya podemos realizar búsquedas y modificar los datos en caso de haber introducido alguna errata.

```
var query= { 'recclass': 'Iron' }
db.meteorite2.find( query )
```

	_id	fall	geolocation	id	mass	name	nametype	recclass	reclat	reclong	year
1	6255a	Fell	{ type : "Point"	110	1000	Madrid	Valid	Iron	45	45	2005-01-01T00:00:00.000
2	6255a	Fell	{ type : "Point"	111	55120	Paris	Valid	Iron	48.85824324	2.2944261500096	2012-01-01T00:00:00.000

Ilustración 3. Búsqueda ejemplo

Nos hemos dado cuenta de que la masa está en formato *string*, lo que nos puede dar problemas en un futuro si quisiéramos realizar algún cálculo. Vamos a actualizar el campo de la masa convirtiendo su valor a *double*.

```
db.meteorite2.updateMany({ },[{$set:{"mass":{$toDouble:'$mass'}}}])
```

Con las fechas nos pasa lo mismo, además de que le acompaña información que es irrelevante. Vamos a crear un nuevo campo que solo contenga el año.

```
var anio = { $year: { $dateFromString: { dateString: '$year' } } }
var fecha = { 'Anio': anio }
```

```
db.meteorite2.updateMany({}, [{ $set: { 'Anio': anio } }, { $unset: 'year' } ])
```

meteorite2		0.302 s	5 Docs					
	_id	Anio	fall	geolocation	id	mass	name	nam
1	6255a	2005 (2.0k)	Fell	{ type: "Point" 110	1000	Madrid	Val	
2	6255a	2012 (2.0k)	Fell	{ type: "Point" 111	55120	Paris	Val	
3	6255a	2008 (2.0k)	Found	{ type: "Point" 112	12300.90	Londres	Val	
4	6255a	2007 (2.0k)	Fell	{ type: "Point" 113	9700	Unknown	Val	
5	6255a	1998 (2.0k)	Fell	{ type: "Point" 114	3200.54	Santa Cruz de la	Val	

Ilustración 4. Nuevo campo año

Las consultas sobre los datos siguen un orden de menos a más de dificultad y en algunos casos una consulta surge del resultado de la anterior. El primer paso es realizar un vistazo general al conjunto y comprobar que todo parece estar en orden.

1. `db.meteorite.find()`
2. `db.meteorite.find().count()`

Con estas dos instrucciones obtenemos un primer contacto con los datos. A primera vista, sabemos que deberemos tener en cuenta para un futuro que, por ejemplo, la masa y las fechas no están en su correspondiente formato.













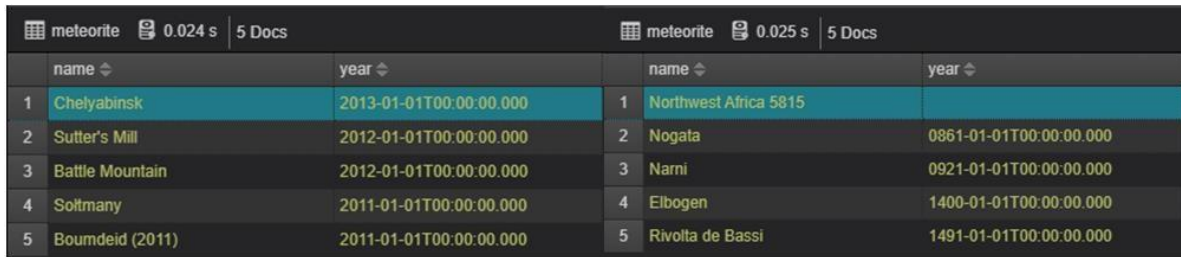
	_id	fall	geolocation	id	mass	name	nametype	reclass	reclat	reclong	year
1	 62 Fell	 { type: "Point", coordinates: [6.1	21	Aachen	Valid	L5	50.775000	6.083330	1880-01-01T00:00:00.000		
2	 62 Fell	 { type: "Point", coordinates: [-9.10	1914	Acapulco	Valid	Acapulcoite	16.883330	-99.900000	1976-01-01T00:00:00.000		
3	 62 Fell	 { type: "Point", coordinates: [10.2	720	Aarhus	Valid	H6	56.183330	10.233330	1951-01-01T00:00:00.000		
4	 62 Fell	 { type: "Point", coordinates: [-8.2276	32000	Allegan	Valid	H5	42.533330	-85.883330	1899-01-01T00:00:00.000		
5	 62 Fell	 { type: "Point", coordinates: [-1.2278	2000000	Allende	Valid	CV3	26.966670	-105.316670	1969-01-01T00:00:00.000		
6	 62 Fell	 { type: "Point", coordinates: [44.2284	6000	Alta'ameem	Valid	LL5	35.273330	44.215560	1977-01-01T00:00:00.000		

Ilustración 5. Tabla de los datos

Empezamos con dos consultas sencillas, en las que consultamos que meteoritos son los más recientes y antiguos.

3. `db.meteorite.find({}, { _id: 0, 'name': 1, 'year': 1 }).sort({ year: -1 }).limit(5)`

```
4. db.meteorite.find({},
    {_id: 0, 'name': 1, 'year': 1}).sort({year: 1}).limit(5)
```



The screenshot shows two query results in MongoDB Compass. The left query (0.024 s) returns 5 documents sorted by year. The right query (0.025 s) returns 5 documents sorted by year.

name	year
Chelyabinsk	2013-01-01T00:00:00.000
Sutter's Mill	2012-01-01T00:00:00.000
Battle Mountain	2012-01-01T00:00:00.000
Softmany	2011-01-01T00:00:00.000
Boumdeld (2011)	2011-01-01T00:00:00.000

name	year
Northwest Africa 5815	0861-01-01T00:00:00.000
Nogata	0921-01-01T00:00:00.000
Narni	1400-01-01T00:00:00.000
Elbogen	1491-01-01T00:00:00.000
Rivolta de Bassi	1491-01-01T00:00:00.000

Ilustración 6. Salida consultas 3 y 4

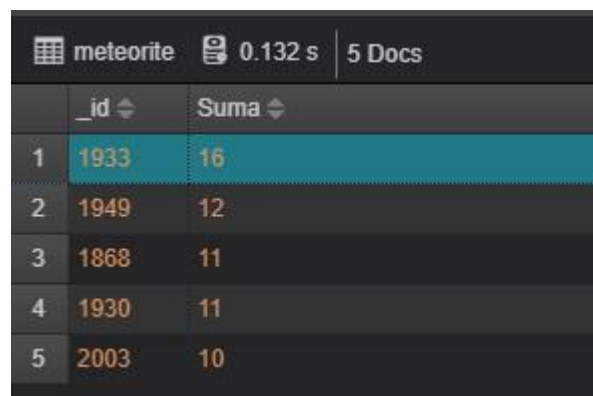
Viendo los resultados de la consulta anterior, nos puede interesar saber cuántos meteoritos han sido registrados por año. Para eso, primero tenemos preparar el campo `year`, ya que se encuentra en formato de texto y contiene información irrelevante. Esta consulta es más fácil de ver si separamos los distintos pasos que transforman la fecha y formatean la salida.

```
var anio = { $year:{ $dateFromString:{ dateString: '$year' }}} }
var fecha = { 'Anio': anio }

var fase1 = { $addFields: fecha }
var fase2 = { $group: {_id: '$Anio', Suma: {$sum: 1}} }
var fase3 = { $sort: {Suma: -1} }
var fase4 = { $limit: 5 }

var etapas = [ fase1, fase2, fase3, fase4 ]
```

```
5. db.meteorite.aggregate(etapas)
```



The screenshot shows the result of an aggregation query in MongoDB Compass. The query (0.132 s) returns 5 documents showing the frequency of meteorite sightings by year.

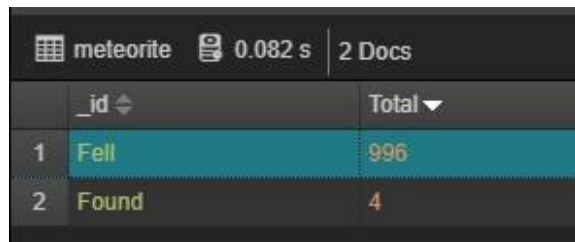
_id	Suma
1933	16
1949	12
1868	11
1930	11
2003	10

Ilustración 7. Frecuencia de caída por año

Vamos a ver cuál es el total de meteoritos que han sido avistados en el cielo o encontrados en la superficie.

```
6. db.meteorite.aggregate([{$group:
```

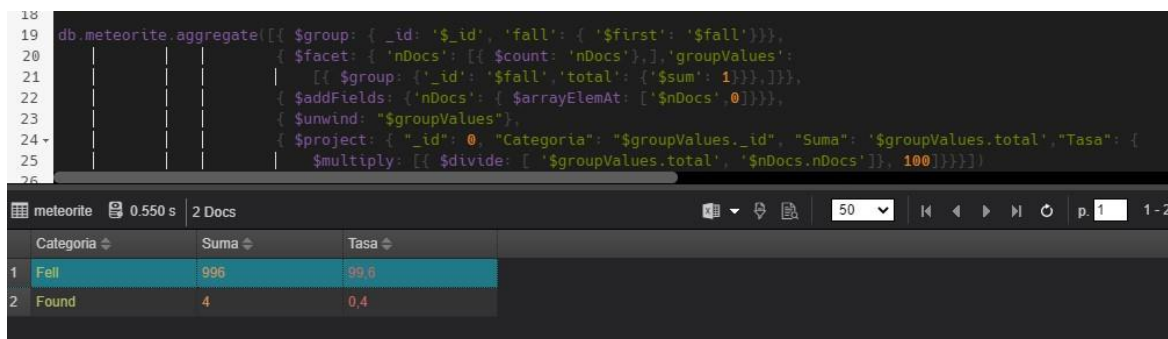
```
{_id: '$fall', Total: {$sum: 1}}}, {$sort: {Total: -
1}}])
```



	_id	Total
1	Fell	996
2	Found	4

Ilustración 8. Meteoritos avistados/encontrados totales

En este caso, hacer la proporción es bastante irrelevante, ya que sólo son dos posibles dos estados (avistado o encontrado) y el total de muestras es 1000. Con la siguiente consulta seríamos capaces de sacar la proporción sin problemas.



```
18 db.meteorite.aggregate([{$group: { '_id': '$_id', 'fall': { '$first': '$fall' } }},
19 { '$facet': { 'nDocs': [{ '$count': 'nDocs' } ], 'groupValues':
20 [{ '$group': { '_id': '$fall', 'total': { '$sum': '1' } } } ] }},
21 { '$addFields': { 'nDocs': { '$arrayElemAt': [ '$nDocs', 0 ] } } },
22 { '$unwind': '$groupValues' },
23 { '$project': { '_id': 0, 'Categoria': '$groupValues._id', 'Suma': '$groupValues.total', 'Tasa': {
24   '$multiply': [ { '$divide': [ '$groupValues.total', '$nDocs.nDocs' ] }, 100 ] } } } ]})
```

	Categoria	Suma	Tasa
1	Fell	996	99.6
2	Found	4	0.4

Ilustración 9. Proporción completa con resultado Vamos

a comprobar que meteoritos no tienen geolocalización registrada.

```
7. db.meteorite.find(
  'geolocation': { $exists: false, $not: { $size: 0 } }, { '_id': 0,
'name': 1, 'fall': 1, 'reclat': 1, 'reclong': 1,
  'year': 1 }).sort({ fall: -1 }).count()
```



	fall	name	year
1	Found	Cumulus Hills 04075	2003-01-01T00:00:00.000
2	Found	Dominion Range 03239	2002-01-01T00:00:00.000
3	Found	Dominion Range 03240	2002-01-01T00:00:00.000
4	Fell	Bulls Run	1964-01-01T00:00:00.000
5	Fell	Clohars	1822-01-01T00:00:00.000
6	Fell	Jalanash	1990-01-01T00:00:00.000
7	Fell	Jemlanur	1901-01-01T00:00:00.000

Ilustración 10. Meteoritos sin geolocalización

Como podemos observar, los meteoritos que no tienen geolocalización tampoco tienen información en los campos latitud y longitud. Es curioso que tres de los cuatro meteoritos que fueron encontrados no tengan registrado unas coordenadas. Si al final de la consulta añadimos `‘.count()’` sabremos el total de objetos sin geolocalización (12 en total).

Pasemos ahora a realizar consultas sobre las clases y tipo de meteoritos. Vamos a ver cuántas categorías existen.

```
8. db.meteorite.aggregate([{$group:
    {_id: '$recclass', Total: {$sum: 1}}},
    {$sort: {Total: -1}}])
```

	_id ↕	Total ↕		_id ↕	Total ↕
1	L6	242	22	CV3	6
2	H5	143	23	Iron, IIAB	6
3	H6	77	24	LL4	6
4	L5	69	25	Ureilite	5
5	H4	48	26	H3-5	5
6	LL6	39	27	Eucrite-cm	5
7	Stone-unc	34	28	Iron, IVA	4
8	OC	21	29	CI1	4

Ilustración 11. Número de categorías

Con esta consulta tenemos un problema, y es que con un `‘.count()’` nos dice que existen 118 categorías diferentes. Esta consulta en realidad nos está mostrando las distintas combinaciones u subcategorías que existen (se puede ver en las entradas 23 y 28 de la ilustración 8).

Siguiendo el ejemplo del hierro, no nos puede quedar claro cuál es el total de meteoritos metálicos. Por ello, vamos a filtrar los tipos para que muestre, por ejemplo, aquellos meteoritos que pertenezcan al grupo metálico. (Count: 45)

```
9. db.meteorite.find({recclass: {$regex: /^iron/i}},
    {'_id': 0, 'name': 1, 'recclass': 1, 'mass': 1 })
    .sort({'name': 1})
```


	mass ↕	name ▲	recclass ↕
1	50000	Akyumak	Iron, IVA
2	1230	Avce	Iron, IIAB
3	10322	Bahjoi	Iron, IAB-sLL
4	16700	Ban Rong Du	Iron, ungrouped
5	8800	Bogou	Iron, IAB-MG
6	256000	Boguslavka	Iron, IIAB
7	39000	Braunau	Iron, IIAB
8	2250	Bulls Run	Iron?

Ilustración 12. Meteoritos metálicos

La salida anterior nos muestra que el grupo metálico tiene una masa relativamente relevante. Vamos a comprobar si es esto cierto.

```
10. db.meteorite.aggregate([{$unwind: '$recclass'}, {$group:
  {$_id: '$recclass', Media: {$avg: { $toDouble: '$mass' } }},
  { $sort: {Media: -1}},
  { $project: {$_id: '$_id', roundAvg: {$ceil: '$Media'}}},
  { $limit: 10}]])
```

	_id ↕	roundAvg ↕
1	Iron, IIAB	3.885.196 (3.9M)
2	CV3	336.217 (0.34M)
3	Mesosiderite-A3/4	320.000 (0.32M)
4	CO3.2	200.000 (0.20M)
5	L/LL5	167.315 (0.17M)
6	L5-6	153.457 (0.15M)
7	Pallasite, PMG	147.500 (0.15M)
8	Aubrite	134.083 (0.13M)
9	Mesosiderite	128.800 (0.13M)
10	L/LL4	112.667 (0.11M)

Ilustración 13. Media pesos

Durante la elaboración de esta tarea me han surgido varias cuestiones relacionadas con las coordenadas de impacto en los meteoritos y si había alguna variable que afectase sobre esta, como pudiera ser la gravedad o el campo magnético.

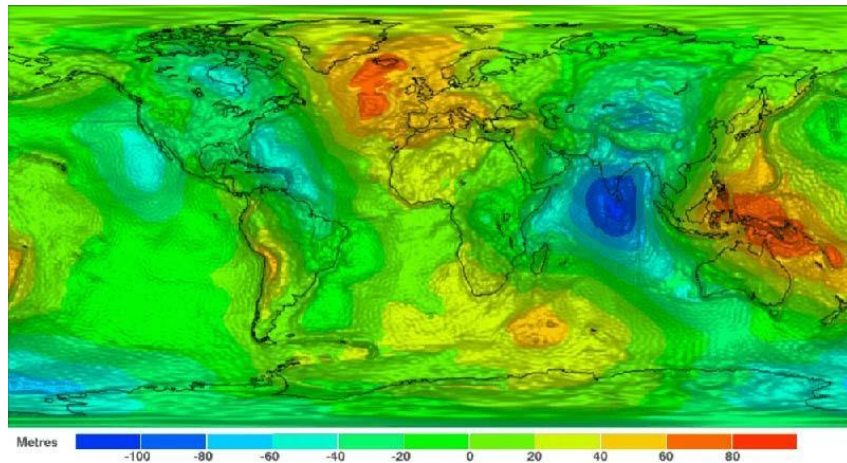


Ilustración 14. Zonas de gravedad en La Tierra

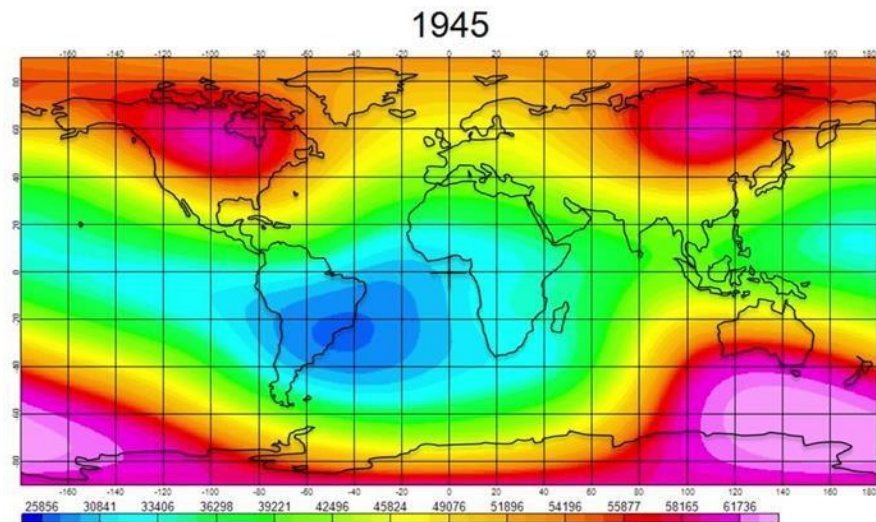


Ilustración 15. Zonas magnéticas de La Tierra

Con el objetivo de buscar alguna relación entre las variables, vamos a ver si los meteoritos tienen una zona favorita de impacto y si coinciden con las zonas de más alto nivel de gravedad y magnetismo. Para ello, vamos a dividir el proceso en fases y así tener una mejor visión de como se van tratando los datos.

```
var zona ={$round:[{$divide : [{ $toDouble: '$reclat'}, 10]], 0}}
var zonaCampo = { 'Zona': zona } var fase1 = {
$addFields: zonaCampo } var fase2 = { $group: {_id:
'$Zona', Suma: {$sum: 1}}}
var fase3 = { $sort: {_id: -1}}

var etapas = [ fase1, fase2, fase3]
11. db.meteorite.aggregate(etapas)
```

Esta consulta se realiza dos veces, una con la latitud y otra con la longitud, con el objetivo de poder construir una zona en un mapa e intentar darle una explicación. La salida latitudinal abarca 180°, desde la Antártida (-90°), hasta el polo norte (+90°), y la salida longitudinal los 360° (de Alaska a Nueva Zelanda).

	_id ↕	Suma ▼
1	4	260
2	5	212
3	3	176
4	2	85
5	1	63
6	-3	53
7	6	53

Ilustración 16. Salida grupos latitud

	_id ↕	Suma ▼
1	1	119
2	0	102
3	8	100
4	3	85
5	2	71
6	-10	50
7	4	47

Ilustración 17. Salida grupos longitud

Si reflejamos en un mapa las cinco primeras zonas donde más se repiten las colisiones nos queda de la siguiente forma:

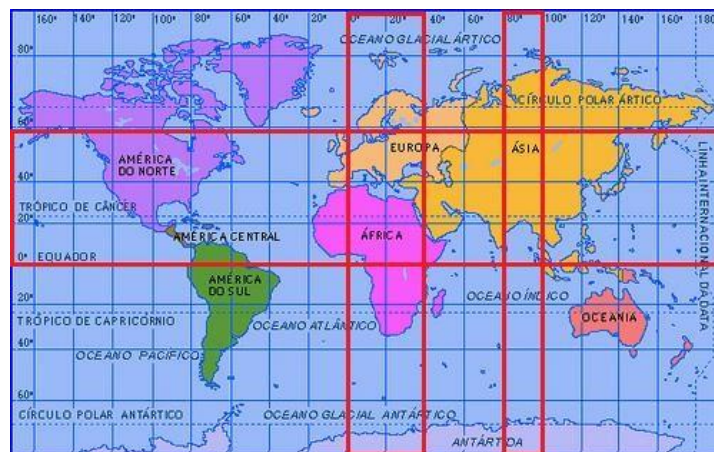


Ilustración 18. Mapamundi con zonas

Ahora vamos a ver si sucede lo mismo con meteoritos metálicos, para ello, añadimos al código anterior una nueva fase de filtrado que recupera los meteoritos que tengan algún componente metálico.

```
var filtro = {recclass: {$regex: /^iron/i}}
var faseFiltro = { $match: filtro }

var etapas = [ fase1, faseFiltro, fase2, fase3]
12. db.meteorite.aggregate( etapas )
```

	_id	Suma
1	4	15
2	3	11
3	5	8
4	2	4
5	1	2
6	-1	2
7	0	1

Ilustración 19. Meteoritos metálicos en latitud

	_id	Suma
1	8	5
2	3	5
3	0	5
4	14	3
5	1	3
6	-9	3
7	13	2

Ilustración 20. Meteoritos metálicos en longitud Hacemos

lo mismo en el mapamundi con los cinco primeros resultados:

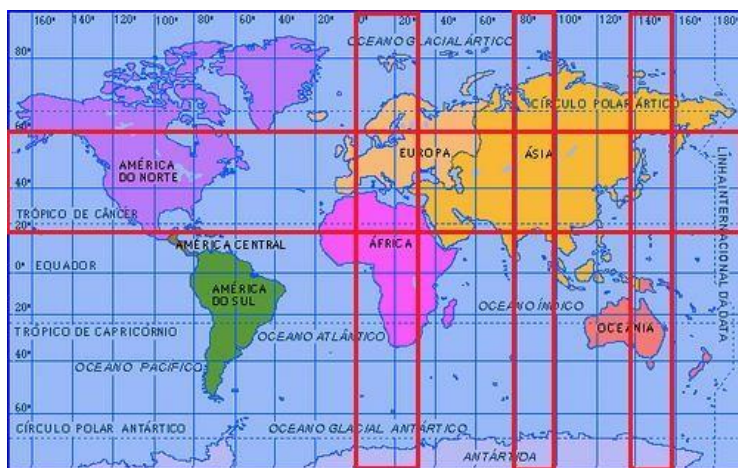


Ilustración 21. Mapamundi con zonas de meteoritos metálicos

Conclusiones

A primera vista parece que las coordenadas sí tienen alguna relación, por lo menos, con el campo gravitatorio terrestre. Puede ser común pensar que la variable magnética también afecte a la trayectoria, pero debido a que los meteoritos no tienen una composición 100% metálica, concluimos que dicho factor tiene poca relevancia. Aunque influyan otros aspectos que no tengamos en cuenta, como es la aerodinámica, la velocidad o ángulo con el que entran en la atmósfera, parece ser que los meteoritos tienen un comportamiento predecible, cuando se puede llegar a pensar lo contrario. Tal vez sea coincidencia con que la zona central sea la parte más expuesta de La Tierra. Para ello deberíamos realizar un análisis con herramientas como R Studio y verificar dicha conjetura.

Un campo que a mi parecer hubiera sido interesante conocer sería el radio o la densidad del meteorito, ya que me quedé un poco extrañado con la consulta sobre la masa media, ya que los que contenían propiedades metálicas, aparecían siempre en el top cinco de peso promedio.

Este Dataset contiene la cantidad de mil registros, sin embargo, en toda la historia de la humanidad se han registrado más de 45.000 muestras. En un principio parecen pocos registros, pero, hay que tener en cuenta que muchas de estas entradas al fin de al cabo son piedras que no llegan ni al kilo, y observar estos cuerpos en el pasado no era tarea sencilla, por eso, la gran mayoría de entradas tienen fechas superiores al siglo XIX.