CASO 1: Crear una nueva colección llamada “sensores” e introducir en ella cuatro documentos, correspondientes a cada uno de los sensores que se encuentran instalados en este momento: dos en Sevilla y dos en Valladolid.

La colección deberá contener las siguientes claves:

- Ubicacion: tipo *string*, indica la ciudad en la que está instalado el sensor.

- “medidas_sensor”: *array* de objetos JSON. Cada JSON tendrá dos subclaves: “tipo_medida”, *string* que identifica lo que mide el sensor; “unidad”: *string* que identifica las unidades en las que mide el sensor cada tipo de medida.

- Coordenadas: *array* de numéricos. Los sensores de Valladolid tienen las coordenadas [41.638597, -4.740186] y los de Sevilla [37.409311, -5.949939].

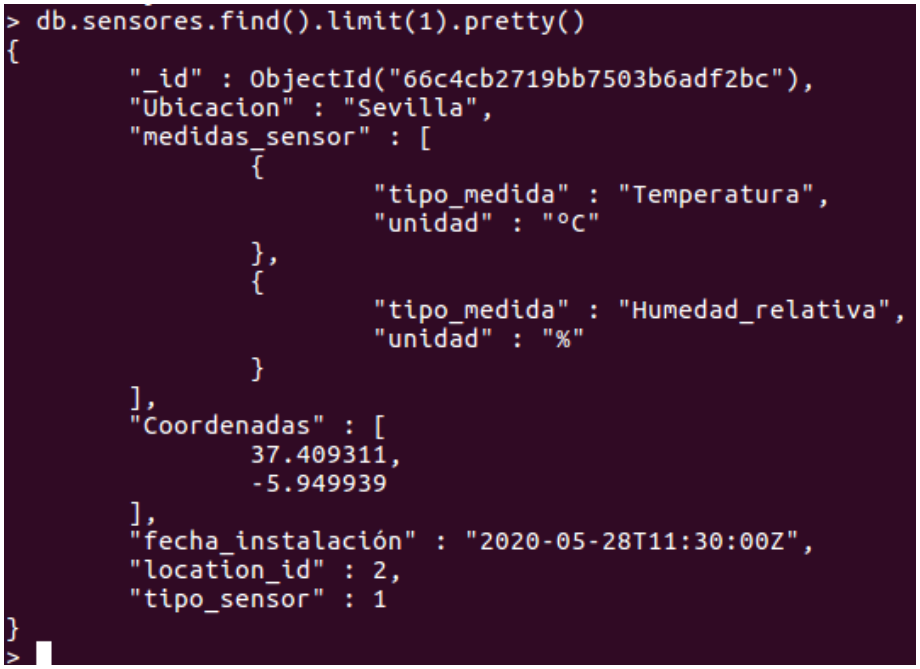
- “fecha_instalación”: *timestamp*. Los dos sensores de Valladolid se instalaron el día 25 de mayo 2020 a las 10:30 AM, mientras que los de Sevilla se instalaron el 28 de mayo 2020 a las 11:30 AM.

- “location_id”: numérico que indica el ID de la ubicación a la que pertenece el sensor. Valor uno para los sensores de Valladolid y valor dos para los sensores de Sevilla.

- “tipo_sensor”: numérico que indica el tipo de sensor. Valor uno para el tipo de sensor que mide temperatura y humedad relativa. Valor dos para el tipo de sensor que mide emisión de CO2 y consumo eléctrico.

La colección se creará automáticamente al insertar los cuatro documentos en una misma consulta. Adjuntar captura de la consulta de inserción, así como el resultado tras ejecutarla.

```
tedb.sensores.insert([{"Ubicacion":"Sevilla", "medidas_sensor":[{"tipo_medida":"Temperatura
","unidad":"ºC"}, {"tipo_medida":"Humedad_relativa","unidad":"%"}], "Coordenadas":[37.40931
1, -5.949939], "fecha_instalación":"2020-05-28T11:30:00Z", "location_id":2, "tipo_sensor":1
},{"Ubicacion":"Sevilla", "medidas_sensor":[{"tipo_medida":"Emision_CO2","unidad":"gCO2/m2"
}, {"tipo_medida":"Consumo_electrico","unidad":"kWh/m2"}], "Coordenadas":[37.409311, -5.949
939], "fecha_instalación":"2020-05-28T11:30:00Z", "location_id":2, "tipo_sensor":2},{"Ubica
cion":"Valladolid", "medidas_sensor":[{"tipo_medida":"Temperatura","unidad":"ºC"}, {"tipo_m
edida":"Humedad_relativa","unidad":"%"}], "Coordenadas":[41.638597, -4.740186], "fecha_inst
alación":"2020-05-25T10:30:00Z", "location_id":1, "tipo_sensor":1},{"Ubicacion":"Valladolid
", "medidas_sensor":[{"tipo_medida":"Emision_CO2","unidad":"gCO2/m2"}, {"tipo_medida":"Cons
umo_electrico","unidad":"kWh/m2"}], "Coordenadas":[41.638597, -4.740186], "fecha_instalació
n":"2020-05-25T10:30:00Z", "location_id":1, "tipo_sensor":2}])
BulkWriteResult({
        "writeErrors" : [ ],
        "writeConcernErrors" : [ ],
        "nInserted" : 4,
        "nUpserted" : 0,
        "nMatched" : 0,
        "nModified" : 0,
        "nRemoved" : 0,
        "upserted" : [ ]
})
>
```



CASO 2: Mostrar el número de datos recogidos por el sensor que mide la temperatura en Valladolid, así como el número de datos recogidos por el de Sevilla entre los días 1 y 10 julio (incluido el día 10 entero).

Haciendo cuentas, se reciben cuatro datos por hora, 96 por día, y, por tanto, en 10 días debería haber 960 datos enviados por cada sensor. Comentar si no se han recibido datos de temperatura en algún intervalo de 15 minutos para alguno de los sensores.

Como podemos ver en la captura anterior, el sensor que mide la temperatura (“sensor_id”:1) en Valladolid (“location_id”:1) recibe 960 datos; en cambio, el que mide la temperatura en Sevilla (“location_id”:2) solo recibe 959.
Es decir, el sensor de Sevilla no recibió datos de temperatura en alguno de los intervalos de 15 minutos.

CASO 3: Se pide identificar si hay algún documento que pueda distorsionar los resultados del estudio. Por ello, se quiere identificar posibles valores erróneos de temperatura enviados por el sensor.

Se pide, por tanto, en primer lugar, identificar los documentos de la colección "datos_sensores" que tengan una temperatura superior a 55 ºC, tanto para Valladolid como para Sevilla (contar el número de casos en cada ciudad y mostrar también los documentos erróneos).

[$elemMatch es un operador para arrays; permite que se cumplan dos condiciones a la vez (temperatura y >55ºC)]

```
> db.datos_sensores.find({
... "location_id":1,
... "medidas":{
... $elemMatch: {"tipo_medida":"Temperatura", "valor":{$gt:55}}
... }
... }).count()
2
> db.datos_sensores.find({
... "location_id":2,
... "medidas":{
... $elemMatch: {"tipo_medida":"Temperatura", "valor":{$gt:55}}
... }
... }).count()
1
>
```

Es decir, hay 2 documentos erróneos en Valladolid y 1 en Sevilla.

A la hora de mostrar los documentos con errores, solamente se devolverán las claves: *timestamp*, "location_id" y de la clave medidas mostrar solo el objeto correspondiente a la temperatura, sin mostrar el objeto de humedad relativa.

Documento con error de Sevilla:

```
> db.datos_sensores.find({
... "location_id":2,
... "medidas":{
... $elemMatch: {"tipo_medida":"Temperatura", "valor":{$gt:55}}
... }
... },{
... "timestamp":1,
... "location_id":1,
... "medidas":{
... $elemMatch: {"tipo_medida":"Temperatura"}
... }
... }).pretty()
{
        "_id" : ObjectId("66c4865b24c80c33545e1db9"),
        "timestamp" : "2020-07-08T03:00:00Z",
        "location_id" : 2,
        "medidas" : [
                {
                        "tipo_medida" : "Temperatura",
                        "valor" : 82.64,
                        "unidad" : "ºC"
                }
        ]
}
>
```

Documentos con errores de Valladolid:

```
> db.datos_sensores.find({
... "location_id":1,
... "medidas":{
... $elemMatch: {"tipo_medida":"Temperatura", "valor":{$gt:55}}
... }
... },{
... "timestamp":1,
... "location_id":1,
... "medidas":{
... $elemMatch: {"tipo_medida":"Temperatura"}
... }
... }).pretty()
{
        "_id" : ObjectId("66c4864224c80c33545e1a1f"),
        "timestamp" : "2020-07-05T17:15:00Z",
        "location_id" : 1,
        "medidas" : [
                {
                        "tipo_medida" : "Temperatura",
                        "valor" : 67.27,
                        "unidad" : "ºC"
                }
        ]
}
{
        "_id" : ObjectId("66c4867e24c80c33545e2343"),
        "timestamp" : "2020-07-11T19:30:00Z",
        "location_id" : 1,
        "medidas" : [
                {
                        "tipo_medida" : "Temperatura",
                        "valor" : 125.48,
                        "unidad" : "ºC"
                }
        ]
}
```

Una vez se hayan identificado, se procede a eliminar estos documentos de la colección, es decir: para ese *timestamp* solo quedará el documento correspondiente a la emisión de CO2 y el consumo eléctrico.

Además de las anteriores, se deberá adjuntar también una captura que demuestre que se han eliminado correctamente dichos documentos, si es que existiera alguno.

```
> db.datos_sensores.deleteMany({
...     "location_id":1,
...     "medidas":{
...         $elemMatch: {"tipo_medida":"Temperatura", "valor":{$gt:55}}
...     }
... })
{ "acknowledged" : true, "deletedCount" : 2 }
> db.datos_sensores.find({
... "location_id":1,
... "medidas":{
... $elemMatch: {"tipo_medida":"Temperatura", "valor":{$gt:55}}
... }
... }).count()
0
> db.datos_sensores.deleteMany({
...     "location_id":2,
...     "medidas":{
...         $elemMatch: {"tipo_medida":"Temperatura", "valor":{$gt:55}}
...     }
... })
{ "acknowledged" : true, "deletedCount" : 1 }
> db.datos_sensores.find({
... "location_id":2,
... "medidas":{
... $elemMatch: {"tipo_medida":"Temperatura", "valor":{$gt:55}}
... }
... }).count()
0
>
```

Vemos que, una vez eliminados los documentos, al volver a realizar la consulta del número de documentos con más de 55 grados nos sale que no hay ninguno.

CASO 4: Buscar el valor mínimo de temperatura en Sevilla. Se considera que es un valor poco realista para Sevilla y que es necesario multiplicarlo por un factor 1,2 para hacerlo un valor algo más real.

Nota: una vez hallado el ID del objeto a actualizar, investigar el funcionamiento de la función **$mul** para comprobar si puede utilizarse en este caso para actualizar el valor o es necesario utilizar otro modificador.

Se deberá adjuntar una captura de la consulta de búsqueda junto con el documento al que le corresponde la temperatura mínima de Sevilla. Además, también se adjuntará la consulta de actualización y, por último, se mostrará todo el documento ya actualizado.

Para encontrar el documento con menor temperatura, usamos un sort que ordene la búsqueda de menor a mayor.
[“medidas.0” es el primer objeto JSON de la clave “medidas”. Es decir, es el objeto en el que se encuentra la temperatura]

```
> db.datos_sensores.find({
... "location_id": 2,
... "sensor_id": 1,
... },{
... "medidas":{
... $elemMatch: {"tipo_medida":"Temperatura"}
... }
... }).sort({"medidas.0.valor":1}).limit(2).pretty()
{
        "_id" : ObjectId("66c4867e24c80c33545e2342"),
        "medidas" : [
                {
                        "tipo_medida" : "Temperatura",
                        "valor" : 9.14,
                        "unidad" : "ºC"
                }
        ]
}
{
        "_id" : ObjectId("66c4862924c80c33545e1673"),
        "medidas" : [
                {
                        "tipo_medida" : "Temperatura",
                        "valor" : 10.75,
                        "unidad" : "ºC"
                }
        ]
}
>
```

Para actualizar el valor se emplea la función $mul:

```
> db.datos_sensores.updateOne({
... "_id": ObjectId("66c4867e24c80c33545e2342")
... },{
... $mul: {"medidas.0.valor": 1.2}
... })
{ "acknowledged" : true, "matchedCount" : 1, "modifiedCount" : 1 }
>
>
```

El documento quedaría de la siguiente manera:

```
> db.datos_sensores.find({"_id": ObjectId("66c4867e24c80c33545e2342")}).pretty()
{
        "_id" : ObjectId("66c4867e24c80c33545e2342"),
        "timestamp" : "2020-07-11T19:30:00Z",
        "sensor_id" : 1,
        "location_id" : 2,
        "medidas" : [
                {
                        "tipo_medida" : "Temperatura",
                        "valor" : 10.968,
                        "unidad" : "ºC"
                },
                {
                        "tipo_medida" : "Humedad_relativa",
                        "valor" : 33.42,
                        "unidad" : "%"
                }
        ]
}
>
```

CASO 5: Obtener los tres máximos valores de consumo eléctrico en un día de fin de semana en ambas ciudades y comparar sus valores (se considera fin de semana a partir de las 16:00 del viernes y hasta las 7:45 del lunes).

Devolver para cada ciudad un documento con el *timestamp* en que se producen esos máximos, el día de la semana y la hora a las que corresponden esos máximos y el valor del consumo eléctrico (subclave “tipo_medida” “Consumo_electrico” + subclave “valor” + subclave “unidad”, es decir: no se quieren obtener los valores de ese día de emisión de CO2).

En primer lugar, creamos variables para los dos fines de semana entre el 1-14 de julio del 2020 (días en los que se tomaron las medidas).

```
> fecha_ini_1 = "2020-07-03T16:00:00Z"
2020-07-03T16:00:00Z
> fecha_fin_1 = "2020-07-06T07:45:00Z"
2020-07-06T07:45:00Z
> fecha_ini_2 = "2020-07-10T16:00:00Z"
2020-07-10T16:00:00Z
> fecha_fin_2 = "2020-07-13T07:45:00Z"
2020-07-13T07:45:00Z
```

Una vez almacenadas las fechas, buscamos los tres valores máximos de consumo eléctrico en Valladolid:

Como podemos ver en la figura anterior, los tres registros de consumo máximo coinciden en el mismo día, el 11 de julio (el máximo a las 01:45).

En cuanto a el sensor de Sevilla, estos son los resultados:

El consumo eléctrico máximo se concentra en el día 5 de julio, con el valor más elevado a las 15:00

El consumo en Sevilla es bastante más elevado que el de Valladolid; además, aunque el consumo de Valladolid en el fin de semana alcanza su máximo durante la noche, el de Sevilla lo hace mayoritariamente por la tarde.
Teniendo en cuenta el mes (Julio), Sevilla seguramente tenga un consumo más elevado debido a las altas temperaturas. En estas condiciones, el uso de aire acondicionado es muy alto por la tarde. Valladolid, en cambio, no tiene unas temperaturas tan extremas, pero sí que desciende la temperatura por la noche, por lo que el consumo podría deberse al uso de calefactores u otros electrodomésticos.

CASO 6: Se considera como perjudicial para la salud cualquier día en que el acumulado de las emisiones de CO2 durante el día completo supere los 420 g CO2/m2. De los 14 días que comprende nuestro estudio, se pretende recuperar cuántos días se superó ese límite para Valladolid y, por otro lado, cuántos días se superó para Sevilla.

Se deben adjuntar tanto la captura del número de veces que se supera el límite para cada ciudad (clave "días_sobre_limite_permitido"), así como otra captura que muestre todos los documentos de Valladolid, por un lado, agrupados por la clave "dia_mes" y el acumulado total de emision CO2 bajo la clave "suma_Emision_CO2", ordenados de mayor a menor emisión.

Presentar las mismas dos capturas para Sevilla y comentar los resultados. ¿Qué tienen los cuatro días de ambas ciudades en los que la emisión de CO2 es más pequeña?

Las consultas serán idénticas para Valladolid y Sevilla, distinguiéndose únicamente por el filtrado sobre la clave "location_id".

Comenzamos mostrando los documentos de Valladolid agrupados por día con su emisión CO2 acumulada:

```
> db.datos_sensores.aggregate([
...   {
...     $match: {
...       "location_id":1,
...       "sensor_id":2
...     }
...   },
...   {
...     $addFields: {
...       fecha: {
...         $dateFromString: { dateString: "$timestamp" }
...       }
...     }
...   },
...   {
...     $group: {
...       _id: {
...         dia_mes: { $dayOfMonth: "$fecha" }
...       },
...       suma_Emision_CO2: {
...         $sum: {
...           $arrayElemAt: [
...             "$medidas.valor",
...             { $indexOfArray: ["$medidas.tipo_medida", "Emision_CO2"] }
...           ]
...         }
...       }
...     }
...   },
...   {
...     $sort: {
...       suma_Emision_CO2: -1
...     }
...   }
... ])
{ "_id" : { "dia_mes" : 9 }, "suma_Emision_CO2" : 425.432 }
{ "_id" : { "dia_mes" : 13 }, "suma_Emision_CO2" : 422.799 }
{ "_id" : { "dia_mes" : 7 }, "suma_Emision_CO2" : 421.447 }
{ "_id" : { "dia_mes" : 2 }, "suma_Emision_CO2" : 420.337 }
{ "_id" : { "dia_mes" : 6 }, "suma_Emision_CO2" : 418.368 }
{ "_id" : { "dia_mes" : 8 }, "suma_Emision_CO2" : 417.688 }
{ "_id" : { "dia_mes" : 14 }, "suma_Emision_CO2" : 416.577 }
{ "_id" : { "dia_mes" : 1 }, "suma_Emision_CO2" : 411.776 }
{ "_id" : { "dia_mes" : 10 }, "suma_Emision_CO2" : 374.376 }
{ "_id" : { "dia_mes" : 3 }, "suma_Emision_CO2" : 369.989 }
{ "_id" : { "dia_mes" : 11 }, "suma_Emision_CO2" : 164.694 }
{ "_id" : { "dia_mes" : 5 }, "suma_Emision_CO2" : 164.522 }
{ "_id" : { "dia_mes" : 4 }, "suma_Emision_CO2" : 164.474 }
{ "_id" : { "dia_mes" : 12 }, "suma_Emision_CO2" : 163.997 }
```

En cuanto a la implementación del código:

- $dateFromString se usa para convertir la cadena de texto que identifica el "timestamp" de cada documento al formato Date.
  Este formato es el necesario para poder extraer el día del mes ($dayOfMonth)

El número de veces que se supera el límite de emisión de CO2 en Valladolid es el siguiente:

```
> db.datos_sensores.aggregate([
...   {
...     $match: {
...       "location_id":1,
...       "sensor_id":2
...     }
...   },
...   {
...     $addFields: {
...       fecha: {
...         $dateFromString: { dateString: "$timestamp" }
...       }
...     }
...   },
...   {
...     $group: {
...       _id: {
...         dia_mes: { $dayOfMonth: "$fecha" }
...       },
...       suma_Emision_CO2: {
...         $sum: {
...           $arrayElemAt: [
...             "$medidas.valor",
...             { $indexOfArray: ["$medidas.tipo_medida", "Emision_CO2"] }
...           ]
...         }
...       }
...     }
...   },
...   {
...     $match: {
...       suma_Emision_CO2: { $gt:420 }
...     }
...   },
...   {
...     $count: "dias_sobre_limite_permitido"
...   }
... ])
{ "dias_sobre_limite_permitido" : 4 }
>
```

Documentos de Sevilla agrupados por día con su emisión CO2 acumulada:

```
> db.datos_sensores.aggregate([
...   {
...     $match: {
...       "location_id":2,
...       "sensor_id":2
...     }
...   },
...   {
...     $addFields: {
...       fecha: {
...         $dateFromString: { dateString: "$timestamp" }
...       }
...     }
...   },
...   {
...     $group: {
...       _id: {
...         dia_mes: { $dayOfMonth: "$fecha" }
...       },
...       suma_Emision_CO2: {
...         $sum: {
...           $arrayElemAt: [
...             "$medidas.valor",
...             { $indexOfArray: ["$medidas.tipo_medida", "Emision_CO2"] }
...           ]
...         }
...       }
...     }
...   },
...   {
...     $sort: {
...       suma_Emision_CO2: -1
...     }
...   }
... ])
{ "_id" : { "dia_mes" : 1 }, "suma_Emision_CO2" : 509.796 }
{ "_id" : { "dia_mes" : 7 }, "suma_Emision_CO2" : 507.367 }
{ "_id" : { "dia_mes" : 6 }, "suma_Emision_CO2" : 503.715 }
{ "_id" : { "dia_mes" : 2 }, "suma_Emision_CO2" : 501.444 }
{ "_id" : { "dia_mes" : 9 }, "suma_Emision_CO2" : 501.261 }
{ "_id" : { "dia_mes" : 13 }, "suma_Emision_CO2" : 500.754 }
{ "_id" : { "dia_mes" : 8 }, "suma_Emision_CO2" : 499.94100000000003 }
{ "_id" : { "dia_mes" : 14 }, "suma_Emision_CO2" : 483.623 }
{ "_id" : { "dia_mes" : 3 }, "suma_Emision_CO2" : 364.4 }
{ "_id" : { "dia_mes" : 10 }, "suma_Emision_CO2" : 362.891 }
{ "_id" : { "dia_mes" : 12 }, "suma_Emision_CO2" : 203.911 }
{ "_id" : { "dia_mes" : 4 }, "suma_Emision_CO2" : 203.811 }
{ "_id" : { "dia_mes" : 11 }, "suma_Emision_CO2" : 203.318 }
{ "_id" : { "dia_mes" : 5 }, "suma_Emision_CO2" : 203.205 }
```

Número de veces que se supera el límite de emisión de CO2 en Sevilla:

```
> db.datos_sensores.aggregate([
...   {
...     $match: {
...       "location_id":2,
...       "sensor_id":2
...     }
...   },
...   {
...     $addFields: {
...       fecha: {
...         $dateFromString: { dateString: "$timestamp" }
...       }
...     }
...   },
...   {
...     $group: {
...       _id: {
...         dia_mes: { $dayOfMonth: "$fecha" }
...       },
...       suma_Emision_CO2: {
...         $sum: {
...           $arrayElemAt: [
...             "$medidas.valor",
...             { $indexOfArray: ["$medidas.tipo_medida", "Emision_CO2"] }
...           ]
...         }
...       }
...     }
...   },
...   {
...     $match: {
...       suma_Emision_CO2: { $gt:420 }
...     }
...   },
...   {
...     $count: "dias_sobre_limite_permitido"
...   }
... ])
{ "dias_sobre_limite_permitido" : 8 }
>
```

¿Qué tienen los cuatro días de ambas ciudades en los que la emisión de CO2 es más pequeña?

Los días con emisiones de CO2 más pequeñas en ambas ciudades son el 4, 5, 11 y 12 de julio. Es decir, los fines de semana (sábado-domingo) se produce una menor emisión de CO2.
Esto podría deberse a varios factores: reducción de la actividad laboral, aumento de las actividades al aire libre, etc.

CASO 7: Obtener la media de emisiones de CO2 de cada ciudad por separado en la hora punta de personas en la oficina, que se considera las 10 AM. Por tanto, se obtendrán por un lado los registros con hora 10 en el *timestamp*, que midan CO2 y que sean de Valladolid y obtener la media de emisiones de CO2 durante los 14 días en esa hora como la clave "Avg_Emision_CO2" (recordar que cada hora se reciben cuatro valores desde el sensor para cada ciudad). Mostrar los resultados ordenados por la clave "Avg_Emision_CO2", mostrando como primer documento la media más baja. Redondear esta clave a dos decimales. Realizar la misma operativa con Sevilla.

Para este apartado se usará un código similar al del Caso 6 pero con algunas modificaciones:

- La fecha pasará de String a Date al comienzo, ya que debemos almacenar la hora para quedarnos con las que sean equivalentes a 10 en el match.
- Agrupando por el día, cambiamos $sum por $avg.
- El $project del final nos permite redondear a dos decimales la media calculada.

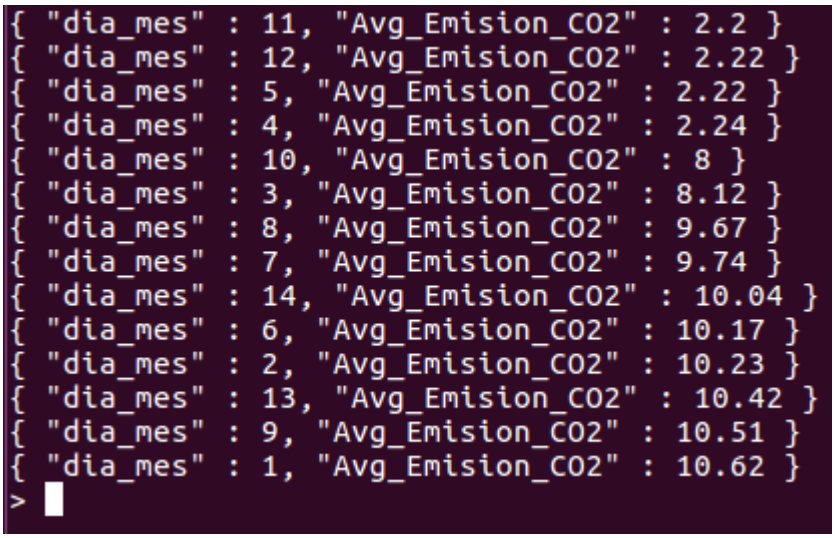
Este es el código que calcula la media de emisiones CO2 en Valladolid a las 10:00 AM:

```
> db.datos_sensores.aggregate([
...   {
...     $addFields: {
...       fecha: { $dateFromString: { dateString: "$timestamp" } }
...     }
...   },
...   {
...     $project: {
...       hora: { $hour: "$fecha" },
...       fecha: 1,
...       medidas: 1,
...       location_id: 1,
...       sensor_id: 1
...     }
...   },
...  {
...     $match: {
...       "location_id":1,
...       "sensor_id":2,
...       "hora": 10
...     }
...   },
...   {
...     $group: {
...       _id: {
...         dia_mes: { $dayOfMonth: "$fecha" }
...       },
...       Avg_Emision_CO2: {
...         $avg: {
...           $arrayElemAt: [
...             "$medidas.valor",
...             { $indexOfArray: ["$medidas.tipo_medida", "Emision_CO2"] }
...           ]
...         }
...       }
...     }
...   },
...   {
...     $project: {
...       _id: 0,
...       dia_mes: "$_id.dia_mes",
...       Avg_Emision_CO2: { $round: ["$Avg_Emision_CO2", 2] }
...     }
...   },
...
...   {
...     $sort: {
...       Avg_Emision_CO2: 1
...     }
...   }
... ])
```

Se obtienen los siguientes resultados:

```
{ "dia_mes" : 11, "Avg_Emision_CO2" : 1.75 }
{ "dia_mes" : 12, "Avg_Emision_CO2" : 1.79 }
{ "dia_mes" : 4, "Avg_Emision_CO2" : 1.81 }
{ "dia_mes" : 5, "Avg_Emision_CO2" : 1.83 }
{ "dia_mes" : 7, "Avg_Emision_CO2" : 8.46 }
{ "dia_mes" : 1, "Avg_Emision_CO2" : 8.81 }
{ "dia_mes" : 6, "Avg_Emision_CO2" : 8.93 }
{ "dia_mes" : 2, "Avg_Emision_CO2" : 9.07 }
{ "dia_mes" : 10, "Avg_Emision_CO2" : 9.12 }
{ "dia_mes" : 14, "Avg_Emision_CO2" : 9.15 }
{ "dia_mes" : 3, "Avg_Emision_CO2" : 9.17 }
{ "dia_mes" : 9, "Avg_Emision_CO2" : 9.24 }
{ "dia_mes" : 8, "Avg_Emision_CO2" : 9.31 }
{ "dia_mes" : 13, "Avg_Emision_CO2" : 9.51 }
>
```

En cuanto a la media de Sevilla, se usaría el mismo código mostrado previamente, pero cambiando “location_id” a 2. Estos son los resultados para Sevilla:



Como podemos observar en los resultados, y como se comentó en apartados previos, Sevilla presenta una emisión de CO2 más elevada en que Valladolid. Además, los días de menos emisión siguen siendo el 4, 5, 11 y 12 de Julio para ambas ciudades.

CASO 8: Se ha descubierto que el sensor de temperatura de Valladolid mide 1,5 ºC de más. Por ello, se pide actualizar todos los valores correspondientes a este sensor decrementando el valor de la temperatura en 1,5 ºC.

Antes de realizar la actualización, se ordenarán mostrando primero el de mayor temperatura, y muestra los dos primeros documentos del sensor de Valladolid con la mayor temperatura. Solo mostrar las siguientes claves: *timestamp*, "location_id". Del *array* de medidas solo mostrar el primer ítem, por ejemplo: **"medidas" : [{ "tipo_medida" : "Temperatura", "valor" : 15.75, "unidad" : "ºC" }] }**. No incluir el **ObjectId)**

Realizar el mismo proceso una vez que has actualizado los valores y realiza capturas de los dos documentos anteriores para comprobar que se han actualizado

Modificando el código empleado en apartados anteriores, encontramos los dos documentos de Valladolid con mayor temperatura:

```
> db.datos_sensores.find({
... "location_id": 1,
... "sensor_id": 1
... },{
... "_id":0,
... "timestamp":1,
... "location_id":1,
... "medidas":{
... $elemMatch: {"tipo_medida":"Temperatura"}
... }
... }).sort({"medidas.0.valor":-1}).limit(2).pretty()
{
        "timestamp" : "2020-07-14T17:30:00Z",
        "location_id" : 1,
        "medidas" : [
                {
                        "tipo_medida" : "Temperatura",
                        "valor" : 40.88,
                        "unidad" : "ºC"
                }
        ]
}
{
        "timestamp" : "2020-07-03T16:30:00Z",
        "location_id" : 1,
        "medidas" : [
                {
                        "tipo_medida" : "Temperatura",
                        "valor" : 40.87,
                        "unidad" : "ºC"
                }
        ]
}
>
>
```

Se modifican de la siguiente manera:

```
> db.datos_sensores.updateMany({
... "location_id": 1,
... "sensor_id": 1,
... "medidas":{
... $elemMatch: {"tipo_medida":"Temperatura"}
... }
... },{
... $inc: {"medidas.$.valor": -1.5}
... })
{ "acknowledged" : true, "matchedCount" : 1342, "modifiedCount" : 1342 }
>
```

Para comprobar que las temperaturas han cambiado, volvemos a mostrar los dos documentos anteriores:

CASO 9: Se quieren analizar los porcentajes de humedad relativa en horario laboral (8-18:00) de ambas ciudades. Recuperar por un lado los cinco documentos con los valores mínimos de humedad relativa en Sevilla y los cinco documentos con humedad relativa mínima en Valladolid por separado (ambos en horario laboral).

Sólo se quieren recuperar de estos documentos el *timestamp* y la subclave **medidas.tipo_medida = Humedad_relativa**, junto con el valor y la unidad asociados.

Analizar si los mínimos se producen siempre en la misma franja horaria (mismo intervalo de 1-2 horas, o es variable). Comentar si varían de una ciudad a otra.

Lo primero que se hace es convertir la cadena con el timestamp a formato Date y extraer la hora, para hacer match solo de los documentos en horario laboral (8:00-18:00). Además, debido a que $elemMatch daba problemas en Project, se decidió usar $filter en su lugar, para mostrar solo los datos sobre la humedad relativa.

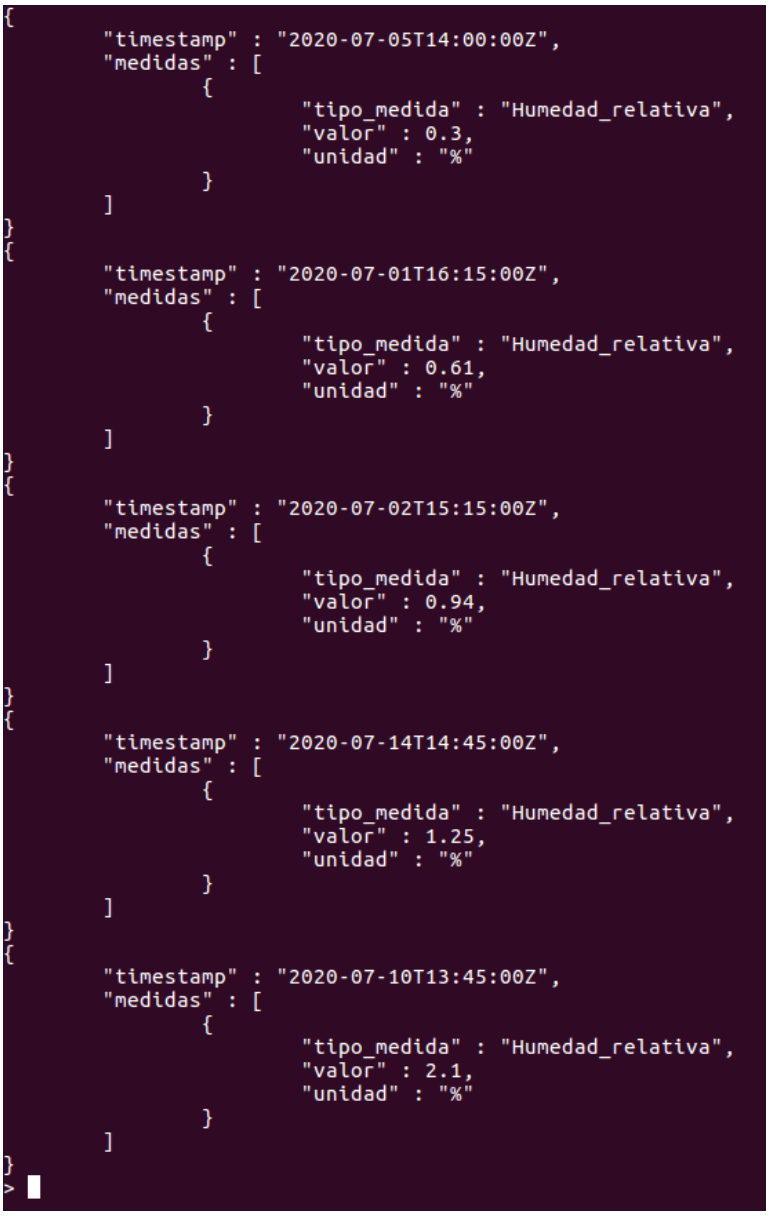
A continuación, se muestra el código para obtener los cinco documentos con los valores mínimos de humedad en Valladolid en horario laboral:

```
> db.datos_sensores.aggregate([
...   {
...     $addFields: {
...       hora: { $hour: {$dateFromString:{dateString:"$timestamp"}}}
...     }
...   },
...   {
...     $match: {
...       "location_id": 1,
...       "sensor_id":1,
...       "medidas.tipo_medida": "Humedad_relativa",
...       "hora": {$gte:8, $lte:18}
...     }
...   },
...   {
...     $project: {
...       "_id": 0,
...       "timestamp": 1,
...       "medidas": {
...         $filter: {
...           input: "$medidas",
...           as: "medida",
...           cond: { $eq: ["$$medida.tipo_medida", "Humedad_relativa"] }
...         }
...       }
...     }
...   },
...   {
...     $sort: {"medidas.valor":1}
...   },
...   {
...     $limit: 5
...   }
... ]).pretty()
```

Estos son los resultados:

```
{
        "timestamp" : "2020-07-03T16:45:00Z",
        "medidas" : [
                {
                        "tipo_medida" : "Humedad_relativa",
                        "valor" : 5.74,
                        "unidad" : "%"
                }
        ]
}
{
        "timestamp" : "2020-07-06T17:00:00Z",
        "medidas" : [
                {
                        "tipo_medida" : "Humedad_relativa",
                        "valor" : 6.81,
                        "unidad" : "%"
                }
        ]
}
{
        "timestamp" : "2020-07-10T17:45:00Z",
        "medidas" : [
                {
                        "tipo_medida" : "Humedad_relativa",
                        "valor" : 7.6,
                        "unidad" : "%"
                }
        ]
}
{
        "timestamp" : "2020-07-14T17:30:00Z",
        "medidas" : [
                {
                        "tipo_medida" : "Humedad_relativa",
                        "valor" : 8.14,
                        "unidad" : "%"
                }
        ]
}
{
        "timestamp" : "2020-07-07T14:45:00Z",
        "medidas" : [
                {
                        "tipo_medida" : "Humedad_relativa",
                        "valor" : 8.53,
                        "unidad" : "%"
                }
        ]
}
>
```

Para Sevilla sería el mismo código, pero cambiando “location_id”. Estos son los documentos con mínimo de humedad en horario laboral en Sevilla:



Analizar si los mínimos se producen siempre en la misma franja horaria (mismo intervalo de 1-2 horas, o es variable). Comentar si varían de una ciudad a otra.

Sevilla presenta un porcentaje de humedad mucho menor que el de Valladolid (0.3% en Sevilla, 5.74% en Valladolid). Y aunque los intervalos se acercan bastante (ambas ciudades presentan mínima humedad por la tarde), en Sevilla el mínimo es sobre las 13:45-16:15 mientras que en Valladolid es un poco más tarde, sobre las 14:45-17:45.

CASO 10: Se quieren actualizar todos los documentos para introducir en el *array* de medidas dos nuevos elementos que van a ser constantes y que van a tener el valor del precio del kWh y de la superficie total de la sede.

Solamente se añadirán a los documentos que tengan como medida el consumo eléctrico (los que tienen temperatura y humedad no).

- Los ítems a introducir son para Valladolid:

{"precio_kWh":0.102,"unidad":"€/kWh"},{"superficie":450,"unidad":"m2"}

- Los ítems a introducir son para Sevilla:

{"precio_kWh":0.107,"unidad":"€/kWh"},{"superficie":550,"unidad":"m2"}

Para comprobar que se ha actualizado correctamente la colección, muestra cuatro documentos de Sevilla y cuatro de Valladolid, correspondientes a los *timestamps*: **2020-07-01T08:00:00Z, 2020-07-01T23:15:00Z**. Cada intervalo se envían dos documentos desde cada ciudad: **(Temperatura y Humedad_relativa) + (Emision_CO2 y Consumo_electrico).**

Para introducir elementos en un array se emplean las funciones $push y $each, como podemos ver en el código que se encuentra a continuación.

```
> db.datos_sensores.updateMany({
... "location_id": 1,
... "sensor_id": 2
... },{$push:{"medidas":{$each: [{"tipo_medida":"precio_kWh", "valor":0.102,"uni
dad":"€/kWh"},{"tipo_medida":"superficie", "valor":450, "unidad":"m2"}]}}})
{ "acknowledged" : true, "matchedCount" : 1344, "modifiedCount" : 1344 }
> db.datos_sensores.updateMany({
... "location_id": 2,
... "sensor_id": 2
... },{$push:{"medidas":{$each:[{"tipo_medida":"precio_kWh", "valor":0.107,"unid
ad":"€/kWh"},{"tipo_medida":"superficie", "valor":550, "unidad":"m2"}]}}})
{ "acknowledged" : true, "matchedCount" : 1344, "modifiedCount" : 1344 }
>
```

Una vez actualizados los “sensor_id”:2 de ambas ciudades (los que miden consumo eléctrico y emisión CO2), se muestran los documentos que se corresponden a las fechas del enunciado para ambos tipos de sensor en ambas ciudades.

```
> db.datos_sensores.find({
... "location_id":1,
... $or: [
... {"timestamp": "2020-07-01T08:00:00Z"},
... {"timestamp": "2020-07-01T23:15:00Z"}
... ]
...
... }).pretty()
```

Estos son los registros para Valladolid el 1 de Julio a las 08:00.


```
{
        "_id" : ObjectId("66c4861a24c80c33545e138b"),
        "timestamp" : "2020-07-01T08:00:00Z",
        "sensor_id" : 1,
        "location_id" : 1,
        "medidas" : [
                {
                        "tipo_medida" : "Temperatura",
                        "valor" : 18.7,
                        "unidad" : "ºC"
                },
                {
                        "tipo_medida" : "Humedad_relativa",
                        "valor" : 77.36,
                        "unidad" : "%"
                }
        ]
}
{
        "_id" : ObjectId("66c4861a24c80c33545e138d"),
        "timestamp" : "2020-07-01T08:00:00Z",
        "sensor_id" : 2,
        "location_id" : 1,
        "medidas" : [
                {
                        "tipo_medida" : "Emision_CO2",
                        "valor" : 6.7,
                        "unidad" : "gCO2/m2"
                },
                {
                        "tipo_medida" : "Consumo_electrico",
                        "valor" : 0.01788,
                        "unidad" : "kWh/m2"
                },
                {
                        "tipo_medida" : "precio_kWh",
                        "valor" : 0.102,
                        "unidad" : "€/kWh"
                },
                {
                        "tipo_medida" : "superficie",
                        "valor" : 450,
                        "unidad" : "m2"
                }
        ]
}
```

Valladolid el 1 de Julio a las 23:15.


```
{
        "_id" : ObjectId("66c4861f24c80c33545e147f"),
        "timestamp" : "2020-07-01T23:15:00Z",
        "sensor_id" : 1,
        "location_id" : 1,
        "medidas" : [
                {
                        "tipo_medida" : "Temperatura",
                        "valor" : 10.06,
                        "unidad" : "ºC"
                },
                {
                        "tipo_medida" : "Humedad_relativa",
                        "valor" : 85.96,
                        "unidad" : "%"
                }
        ]
}
{
        "_id" : ObjectId("66c4861f24c80c33545e1481"),
        "timestamp" : "2020-07-01T23:15:00Z",
        "sensor_id" : 2,
        "location_id" : 1,
        "medidas" : [
                {
                        "tipo_medida" : "Emision_CO2",
                        "valor" : 1.629,
                        "unidad" : "gCO2/m2"
                },
                {
                        "tipo_medida" : "Consumo_electrico",
                        "valor" : 0.00164,
                        "unidad" : "kWh/m2"
                },
                {
                        "tipo_medida" : "precio_kWh",
                        "valor" : 0.102,
                        "unidad" : "€/kWh"
                },
                {
                        "tipo_medida" : "superficie",
                        "valor" : 450,
                        "unidad" : "m2"
                }
        ]
}
```

Sevilla el 1 de Julio a las 08:00.


```
{
        "_id" : ObjectId("66c4861a24c80c33545e138c"),
        "timestamp" : "2020-07-01T08:00:00Z",
        "sensor_id" : 1,
        "location_id" : 2,
        "medidas" : [
                {
                        "tipo_medida" : "Temperatura",
                        "valor" : 22.6,
                        "unidad" : "ºC"
                },
                {
                        "tipo_medida" : "Humedad_relativa",
                        "valor" : 40.73,
                        "unidad" : "%"
                }
        ]
}
{
        "_id" : ObjectId("66c4861a24c80c33545e138e"),
        "timestamp" : "2020-07-01T08:00:00Z",
        "sensor_id" : 2,
        "location_id" : 2,
        "medidas" : [
                {
                        "tipo_medida" : "Emision_CO2",
                        "valor" : 10.262,
                        "unidad" : "gCO2/m2"
                },
                {
                        "tipo_medida" : "Consumo_electrico",
                        "valor" : 0.01467,
                        "unidad" : "kWh/m2"
                },
                {
                        "tipo_medida" : "precio_kWh",
                        "valor" : 0.107,
                        "unidad" : "€/kWh"
                },
                {
                        "tipo_medida" : "superficie",
                        "valor" : 550,
                        "unidad" : "m2"
                }
        ]
}
```

Sevilla el 1 de Julio a las 23:15.