

Parte 1: Bases de datos NoSQL y Relacionales

- 1)
 - a) **Bases de datos:** Las bases de datos existen en MongoDB. Las databases en mongo almacenan una o más **colecciones** de **documentos**.
 - b) **Tabla / Relación:** En MongoDB no hay tablas. Las **colecciones** son análogas a las tablas.
 - c) **Fila / tupla:** Como no hay tablas, tampoco hay filas. El equivalente a las filas en MongoDB son los **documentos**. Los documentos son similares a un objeto JSON, y están compuestos de pares clave-valor. El valor puede ser de cualquier tipo soportado por los documentos BSON, que son una representación binaria de JSON. Estos pueden ser documentos, arreglos, arreglos de documentos, etc.
 - d) **Columnas:** Así como no tiene filas, MongoDB tampoco tiene columnas. El equivalente serían las claves de los pares clave-valor, o nombres de campo. Si bien no es exactamente lo mismo, porque las colecciones no requieren que los documentos almacenados tengan el mismo esquema, es decir, que tengan todos los mismos campos, si pueden establecerse validaciones para que los campos se comporten más *como* una columna.
- 2) No existe el concepto de clave foránea en MongoDB. Pueden hacerse referencias a otros documentos utilizando su clave primaria, ya que esta es también única, pero MongoDB no ofrece un soporte nativo para manejar esto: no hay “joins” y la integridad referencial debe ser enteramente manejada por la aplicación. Si pueden utilizarse documentos embebidos, es decir, como la estructura de los documentos en Mongo es más flexible, Mongo recomienda almacenar datos relacionados juntos en el mismo documento: así, yo puedo guardarme en un campo de mi documento los datos que preciso de otro.
- 3) Existen diferentes tipos de índices para diferentes tipos de datos:
 - a) **Single field:** recolectan y ordenan datos de un campo en cada documento de una colección.
 - b) **Compound:** Recolectan y ordenan datos de dos o más campos en cada documento de una colección.
 - c) **Multikey:** Recolectan y ordenan datos almacenados en arreglos.
 - d) **Geospatial:** Mejoran la performance para queries que involucran datos de coordenadas geoespaciales.
 - e) **Text:** Soportan queries de búsqueda de texto en campos que contienen strings.
 - f) **Wildcard:** Índices que soportan queries contra campos que pueden almacenar datos arbitrarios o desconocidos.
 - g) **Hashed:** Recolecta y almacena hashes de los valores de los campos indexados.
- 4) Las vistas son objetos de solo lectura sobre las que se pueden realizar queries. Existen dos tipos: las estándar y las materializadas on-demand. Ambas devuelven los resultados de un **aggregation pipeline**, es decir, de una serie de operaciones de manipulación de datos realizadas sobre los documentos. La diferencia es que las vistas estándar son computadas cuando son leídas, y no se almacenan en disco. En

cambio, las on-demand materialized se almacenan y se leen de disco. Una view puede usarse para mostrar un documento con menos datos que quisieras mantener privados, por ejemplo, o para unir dos documentos y presentar la información de la forma que quieras, sin que la aplicación deba saber necesariamente la forma en que efectivamente está almacenada esa información. Las on-demand materialized pueden usarse, por ejemplo, para resumir y presentar información en determinados períodos de tiempo.

- 5) Si, Mongo permite **Schema Validation**, que permiten establecer parámetros para la inserción de nuevos documentos en una colección. Así, pueden definirse los campos, los tipos de datos permitidos o incluso rangos para estos. Una vez definido el esquema de validación, todas las inserciones de nuevos documentos deben matchear con el esquema, de lo contrario, la inserción será rechazada.
- 6) En MongoDB, las operaciones sobre un único documento son atómicas. Como se pueden usar arreglos y documentos embebidos, se elimina en muchos casos la necesidad de operar sobre varios documentos a la vez, por lo que la atomicidad de los documentos obvia en muchos la necesidad de transacciones. Aun así, para situaciones que requieren atomicidad en la manipulación de varios documentos, Mongo provee soporte para transacciones distribuidas.
- 7) **Documentos embebidos:** Las relaciones mediante documentos embebidos se implementan insertando dentro de un campo los datos del documento que quiero relacionar. Esto tiene una gran ventaja con respecto a la forma estándar de las bases de datos relacional y es la velocidad: no necesito hacer ningún tipo de operación de joins ya que tengo todos los datos que requiero en un mismo lugar. La mayor desventaja es que voy a tener datos repetidos; también puede ser una desventaja el tamaño que puede tomar el documento, por lo que esta forma de relación es más utilizada con datos concretos y concisos.
Referencia: Las referencias actúan de manera similar a las relaciones en un BBDD relacional, insertando la clave primaria de un documento en un campo de otro. Tienen mayor flexibilidad por la propia naturaleza de los documentos, pero tiene como desventaja que necesitan hacerse varias consultas para recuperar los datos, ya que Mongo no maneja “joins” de manera nativa, y que el manejo de la integridad referencial entre los documentos queda a cargo enteramente del administrador o de la aplicación, Mongo no provee integridad referencial como si lo hacen las BBDD relacionales.
- 8) Las relaciones mediante documentos embebidos son útiles cuando los datos se acceden frecuentemente juntos, y cuando tienen un tamaño razonable; de otra manera, probablemente sea conveniente mapear la relación como referencia. Un buen criterio podría ser mirar el método de fetch que implementan, aquellas relaciones que tienen un fetch eager sería conveniente mapearlas como documentos embebidos, mientras que las que tienen fetch lazy quizá sea conveniente hacerlo como referencia. Por ejemplo, una relación que podría mapearse como documento embebido sería la de Purchase con su Review; o la de Service con su Supplier. Por otro lado, en las relaciones muchos a muchos, como la de Routes con DriverUser o con TourGuideUser sería conveniente hacerlo mediante referencia.

9) i) y ii)

para crear la base utilice "use tours"

```
tours> db.recorridos.insertOne({nombre: "City Tour", "precio": 200, "stops": ["Diagonal Norte", "Avenida de Mayo", "Plaza del Congreso"],
"totalKm":5});
{
  acknowledged: true,
  insertedId: ObjectId('6656530877935b9214a26a13')
}
tours> db.products.find()
tours> db.recorridos.find()
[
  {
    _id: ObjectId('6656530877935b9214a26a13'),
    nombre: 'City Tour',
    precio: 200,
    stops: [ 'Diagonal Norte', 'Avenida de Mayo', 'Plaza del Congreso' ],
    totalKm: 5
  }
]
```

10) mongoimport --db tours --collection recorridos --file material_adicional_1.json --jsonArray

a) tours> db.recorridos.updateOne({nombre: "Cultural Odyssey"}, { \$set: { totalKm: 12}, \$currentDate: {lastUpdated: true}})

```
tours> db.recorridos.find ({nombre: "Cultural Odyssey"})
[
  {
    _id: ObjectId('6656607e87692663e83c12a6'),
    nombre: 'Cultural Odyssey',
    precio: 715.0000000000001,
    stops: [
      'Bosques de Palermo',
      'San Telmo',
      'La Boca - Caminito',
      'Recoleta',
      'El Monumental (Estadio River Plate)',
      'Av 9 de Julio',
      'Plaza Italia'
    ],
    totalKm: 12,
    lastUpdated: ISODate('2024-05-28T22:58:22.605Z')
  }
]
```

b) tours> db.recorridos.updateOne({nombre: "Delta Tour"}, {\$push: {stops: "Tigre"}})

```
tours> db.recorridos.find ({nombre: "Delta Tour"})
[
  {
    _id: ObjectId('6656607e87692663e83c129f'),
    nombre: 'Delta Tour',
    precio: 880.0000000000001,
    stops: [ 'Rio de la Plata', 'Bosques de Palermo', 'Delta', 'Tigre' ],
    totalKm: 8
  }
]
```

c) `tours> db.recorridos.updateMany({},{$mul: {precio: 1.10}})`

```
tours> db.recorridos.updateMany({},{$mul: {precio: 1.10}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 14,
  modifiedCount: 14,
  upsertedCount: 0
}
```

d) `tours> db.recorridos.deleteOne({nombre: "Temporal Route"})`

```
tours> db.recorridos.find ({nombre: "Temporal Route" })
tours> 
```

e) `tours> db.recorridos.updateOne({nombre: "Urban Exploration"}, {$set: {tags: ["Gastronomia"]}})`

```
tours> db.recorridos.find ({nombre: "Urban Exploration" })
[
  {
    _id: ObjectId('6656607e87692663e83c12a2'),
    nombre: 'Urban Exploration',
    precio: 495.00000000000006,
    stops: [
      'Avenida de Mayo',
      'Museo Moderno',
      'Paseo de la Historieta',
      'Usina del Arte',
      'San Telmo',
      'La Boca - Caminito',
      'Belgrano - Barrio Chino',
      'Recoleta',
      'El Monumental (Estadio River Plate)',
      'Costanera Sur',
      'Plaza Italia'
    ],
    totalKm: 14,
    tags: [ 'Gastronomia' ]
  }
]
```

11)

i) `tours> db.recorridos.find({nombre: "Museum Tour"})`

```
tours> db.recorridos.find ({nombre: "Museum Tour" })
[
  {
    _id: ObjectId('6656607e87692663e83c12a3'),
    nombre: 'Museum Tour',
    precio: 605,
    stops: [
      'Museo Nacional de Bellas Artes',
      'Teatro Colón',
      'Planetario',
      'Bosques de Palermo',
      'San Telmo',
      'La Boca - Caminito',
      'Recoleta',
      'El Monumental (Estadio River Plate)',
      'Av 9 de Julio'
    ],
    totalKm: 13
  }
]
```

ii) tours> db.recorridos.find({precio: {\$gt:600 }})

```
tours> db.recorridos.find( {precio: {$gt:600 }})
[
  {
    _id: ObjectId('6656607e87692663e83c129e'),
    nombre: 'Artistic Journey',
    precio: 660,
    stops: [
      'Museo Nacional de Bellas Artes',
      'Teatro Colón',
      'Usina del Arte',
      'Planetario',
      'San Telmo',
      'La Boca - Caminito',
      'Belgrano - Barrio Chino',
      'Av 9 de Julio'
    ],
    totalKm: 15
  },
  {
    _id: ObjectId('6656607e87692663e83c129f'),
    nombre: 'Delta Tour',
    precio: 880.00000000000001,
    stops: [ 'Rio de la Plata', 'Bosques de Palermo', 'Delta', 'Tigre' ],
    totalKm: 8
  },
  {
    _id: ObjectId('6656607e87692663e83c12a1'),
    nombre: 'Gastronomic Delight',
    precio: 770.00000000000001,
    stops: [
      'San Telmo',
      'La Boca - Caminito',
      'Belgrano - Barrio Chino',
      'Recoleta',
      'Costanera Sur'
    ],
    totalKm: 9
  },
  {

```

iii) tours> db.recorridos.find({precio: {\$gt:500 }, totalKm: {\$gt:10}})

```
tours> db.recorridos.find( {precio: {$gt:500 }, totalKm: {$gt:10}})
[
  {
    _id: ObjectId('6656607e87692663e83c129e'),
    nombre: 'Artistic Journey',
    precio: 660,
    stops: [
      'Museo Nacional de Bellas Artes',
      'Teatro Colón',
      'Usina del Arte',
      'Planetario',
      'San Telmo',
      'La Boca - Caminito',
      'Belgrano - Barrio Chino',
      'Av 9 de Julio'
    ],
    totalKm: 15
  },
  {
    _id: ObjectId('6656607e87692663e83c12a3'),
    nombre: 'Museum Tour',
    precio: 605,
    stops: [
      'Museo Nacional de Bellas Artes',
      'Teatro Colón',
      'Planetario',
      'Bosques de Palermo',
      'San Telmo',
      'La Boca - Caminito',
      'Recoleta',
      'El Monumental (Estadio River Plate)',
      'Av 9 de Julio'
    ],
    totalKm: 13
  },
  {
    _id: ObjectId('6656607e87692663e83c12a6'),
    nombre: 'Cultural Odyssey',
    precio: 715.0000000000001,
    stops: [
      'Bosques de Palermo',
      'San Telmo',
      'La Boca - Caminito',
      'Recoleta',
      'El Monumental (Estadio River Plate)',
      'Av 9 de Julio',
      'Plaza Italia'
    ],
    totalKm: 12,
  }
]
```

iv) tours> db.recorridos.find({stops: "San Telmo" })

```

tours> db.recorridos.find( {stops: "San Telmo" })
[
  {
    _id: ObjectId('6656607e87692663e83c129e'),
    nombre: 'Artistic Journey',
    precio: 660,
    stops: [
      'Museo Nacional de Bellas Artes',
      'Teatro Colón',
      'Usina del Arte',
      'Planetario',
      'San Telmo',
      'La Boca - Caminito',
      'Belgrano - Barrio Chino',
      'Av 9 de Julio'
    ],
    totalKm: 15
  },
  {
    _id: ObjectId('6656607e87692663e83c12a0'),
    nombre: 'Tango Experience',
    precio: 385.00000000000006,
    stops: [
      'Avenida de Mayo',
      'Plaza del Congreso',
      'Paseo de la Historieta',
      'San Telmo',
      'La Boca - Caminito',
      'Recoleta'
    ],
    totalKm: 10
  },
  {
    _id: ObjectId('6656607e87692663e83c12a1'),
    nombre: 'Gastronomic Delight',
    precio: 770.00000000000001,
    stops: [
      'San Telmo',
      'La Boca - Caminito',
      'Belgrano - Barrio Chino',
      'Recoleta',
      'Costanera Sur'
    ],
    totalKm: 9
  },
]

```

v) tours> db.recorridos.find({stops: {\$eq:"Recoleta", \$ne:"Plaza Italia"}})

```
tours> db.recorridos.find({stops: {$eq:"Recoleta", $ne:"Plaza Italia"}})
[
  {
    _id: ObjectId('6656607e87692663e83c12a0'),
    nombre: 'Tango Experience',
    precio: 465.85000000000014,
    stops: [
      'Avenida de Mayo',
      'Plaza del Congreso',
      'Paseo de la Historieta',
      'San Telmo',
      'La Boca - Caminito',
      'Recoleta'
    ],
    totalKm: 10
  },
  {
    _id: ObjectId('6656607e87692663e83c12a1'),
    nombre: 'Gastronomic Delight',
    precio: 931.7000000000003,
    stops: [
      'San Telmo',
      'La Boca - Caminito',
      'Belgrano - Barrio Chino',
      'Recoleta',
      'Costanera Sur'
    ],
    totalKm: 9
  },
  {
    _id: ObjectId('6656607e87692663e83c12a3'),
    nombre: 'Museum Tour',
    precio: 732.0500000000001,
    stops: [
      'Museo Nacional de Bellas Artes',
      'Teatro Colón',
      'Planetario',
      'Bosques de Palermo',
      'San Telmo',
      'La Boca - Caminito',
      'Recoleta',
      'El Monumental (Estadio River Plate)',
      'Av 9 de Julio'
    ],
    totalKm: 13
  }
]
```

vi) tours> db.recorridos.find({stops: "Delta", precio: {\$lt: 500} }, {nombre:1, totalKm:1, _id:0})

```
tours> db.recorridos.find( {stops: "Delta", precio: {$lt: 500} }, {nombre:1, totalKm:1, _id:0})
[ { nombre: 'Nature Escape' } ]
```


vii) tours> db.recorridos.find({\$and: [{stops: "Avenida de Mayo"}, {\$or: [{stops: "San Telmo"},{stops: "Recoleta"}]}]})

```
tours> db.recorridos.find( {$and: [{stops: "Avenida de Mayo"}, {$or: [{stops: "San Telmo"},{stops: "Recoleta"}]} ]})
[
  {
    _id: ObjectId('6656607e87692663e83c12a0'),
    nombre: 'Tango Experience',
    precio: 385.00000000000006,
    stops: [
      'Avenida de Mayo',
      'Plaza del Congreso',
      'Paseo de la Historieta',
      'San Telmo',
      'La Boca - Caminito',
      'Recoleta'
    ],
    totalKm: 10
  },
  {
    _id: ObjectId('6656607e87692663e83c12a2'),
    nombre: 'Urban Exploration',
    precio: 495.00000000000006,
    stops: [
      'Avenida de Mayo',
      'Museo Moderno',
      'Paseo de la Historieta',
      'Usina del Arte',
      'San Telmo',
      'La Boca - Caminito',
      'Belgrano - Barrio Chino',
      'Recoleta',
      'El Monumental (Estadio River Plate)',
      'Costanera Sur',
      'Plaza Italia'
    ],
    totalKm: 14,
    tags: [ 'Gastronomia' ]
  },
]
```

viii) tours> db.recorridos.find({\$expr:{\$gt:[\$size:"\$stops"], 5}})

```
tours> db.recorridos.find( {$expr:{$gt:[$size:"$stops"], 5}})
[
  {
    _id: ObjectId('6656607e87692663e83c129d'),
    nombre: 'Historical Adventure',
    precio: 330,
    stops: [
      'Avenida de Mayo',
      'Plaza del Congreso',
      'Rio de la Plata',
      'Teatro Colón',
      'Av 9 de Julio',
      'Plaza Italia'
    ],
    totalKm: 10
  },
  {
    _id: ObjectId('6656607e87692663e83c129e'),
    nombre: 'Artistic Journey',
    precio: 660,
    stops: [
      'Museo Nacional de Bellas Artes',
      'Teatro Colón',
      'Usina del Arte',
      'Planetario',
      'San Telmo',
      'La Boca - Caminito',
      'Belgrano - Barrio Chino',
      'Av 9 de Julio'
    ],
    totalKm: 15
  },
  {
    _id: ObjectId('6656607e87692663e83c12a0'),
    nombre: 'Tango Experience',
    precio: 385.00000000000006,
    stops: [
      'Avenida de Mayo',
      'Plaza del Congreso',
      'Paseo de la Historieta',
      'San Telmo',
      'La Boca - Caminito',
      'Recoleta'
    ],
    totalKm: 10
  },
]
```

ix) tours> db.recorridos.find({totalKm: {\$exists: false}})

```
tours> db.recorridos.find( {totalKm: {$exists: false}})
[
  {
    _id: ObjectId('6656607e87692663e83c12a9'),
    nombre: 'Nature Escape',
    precio: 440.00000000000006,
    stops: [ 'Delta', 'Rio de la Plata', 'Av 9 de Julio', 'Puerto Madero' ]
  }
]
```

x) tours> db.recorridos.find({stops: /Museo/, {nombre:1, stops:1,_id:0}})

```
tours> db.recorridos.find( {stops: /Museo/ }, {nombre:1, stops:1,_id:0})
[
  {
    nombre: 'Artistic Journey',
    stops: [
      'Museo Nacional de Bellas Artes',
      'Teatro Colón',
      'Usina del Arte',
      'Planetario',
      'San Telmo',
      'La Boca - Caminito',
      'Belgrano - Barrio Chino',
      'Av 9 de Julio'
    ]
  },
  {
    nombre: 'Urban Exploration',
    stops: [
      'Avenida de Mayo',
      'Museo Moderno',
      'Paseo de la Historieta',
      'Usina del Arte',
      'San Telmo',
      'La Boca - Caminito',
      'Belgrano - Barrio Chino',
      'Recoleta',
      'El Monumental (Estadio River Plate)',
      'Costanera Sur',
      'Plaza Italia'
    ]
  },
  {
    nombre: 'Museum Tour',
    stops: [
      'Museo Nacional de Bellas Artes',
      'Teatro Colón',
      'Planetario',
      'Bosques de Palermo',
      'San Telmo',
      'La Boca - Caminito',
      'Recoleta',
      'El Monumental (Estadio River Plate)',
      'Av 9 de Julio'
    ]
  }
]
```

xi) tours> db.recorridos.countDocuments()

```
tours> db.recorridos.countDocuments()
14
```

Parte 3: Aggregation Framework

12)

```
tours2> load('/home/luciano/Descargas/generator1.js')
DeprecationWarning: Collection.insert() is deprecated. Use insertOne, insertMany, or bulkWrite.
true
tours2> show collections
route
stop
tours2> db.route.find()
[
  {
    _id: ObjectId('6654d8e113c4802a27a26a8a'),
    name: 'route1',
    price: 532,
    totalKm: 15.8,
    stops: [ 56, 84, 18 ]
  },
  {
    _id: ObjectId('6654d8e113c4802a27a26a8b'),
    name: 'route2',
    price: 809,
    totalKm: 15.62,
    stops: [ 39, 111, 45 ]
  },
  {
    _id: ObjectId('6654d8e113c4802a27a26a8c'),
    name: 'route3',
    price: 1200,
    totalKm: 16.5,
    stops: [ 100, 150, 200 ]
  },
  {
    _id: ObjectId('6654d8e113c4802a27a26a8d'),
    name: 'route4',
    price: 950,
    totalKm: 14.2,
    stops: [ 70, 90, 60 ]
  },
  {
    _id: ObjectId('6654d8e113c4802a27a26a8e'),
    name: 'route5',
    price: 1100,
    totalKm: 17.1,
    stops: [ 120, 180, 220 ]
  },
  {
    _id: ObjectId('6654d8e113c4802a27a26a8f'),
    name: 'route6',
    price: 880,
    totalKm: 13.9,
    stops: [ 80, 110, 90 ]
  },
  {
    _id: ObjectId('6654d8e113c4802a27a26a90'),
    name: 'route7',
    price: 1050,
    totalKm: 15.3,
    stops: [ 110, 140, 170 ]
  },
  {
    _id: ObjectId('6654d8e113c4802a27a26a91'),
    name: 'route8',
    price: 920,
    totalKm: 14.7,
    stops: [ 95, 125, 105 ]
  },
  {
    _id: ObjectId('6654d8e113c4802a27a26a92'),
    name: 'route9',
    price: 1150,
    totalKm: 16.8,
    stops: [ 130, 160, 190 ]
  },
  {
    _id: ObjectId('6654d8e113c4802a27a26a93'),
    name: 'route10',
    price: 850,
    totalKm: 13.5,
    stops: [ 75, 105, 85 ]
  }
]
```

13) De la documentación: “Aggregation operations procesan registros de datos y devuelven resultados computados. Las operaciones de agregación agrupan valores de múltiples documentos, y pueden realizar una variedad de operaciones en los datos agrupados para devolver un resultado simple”.

a) Obtenga una muestra de 5 rutas aleatorias de la colección

```
db.route.aggregate(
  { $sample: {size: 5}}
)
```

```
tours2> db.route.aggregate(
...   { $sample: {size: 5}}
... )
[
  {
    _id: ObjectId('6654d8e413c4802a27a27cd4'),
    name: 'route4683',
    price: 572,
    totalKm: 4.49,
    stops: [ 37, 65 ]
  },
  {
    _id: ObjectId('6654d90113c4802a27a32d82'),
    name: 'route49913',
    price: 724,
    totalKm: 15.25,
    stops: [ 109, 18, 12, 50 ]
  },
  {
    _id: ObjectId('6654d8fa13c4802a27a300c3'),
    name: 'route38458',
    price: 740,
    totalKm: 15.78,
    stops: [ 45, 8, 67 ]
  },
  {
    _id: ObjectId('6654d8fc13c4802a27a31144'),
    name: 'route42683',
    price: 419,
    totalKm: 10.47,
    stops: [ 6, 34 ]
  },
  {
    _id: ObjectId('6654d8fe13c4802a27a31c09'),
    name: 'route45440',
    price: 826,
    totalKm: 10.5,
    stops: [ 42, 3, 113, 16, 22 ]
  }
]
```

b) Extienda la consulta anterior para incluir en el resultado toda la información de cada una de las Stops. Note que puede ligarlas por su código.

```

    db.route.aggregate([
    {$sample:{size: 5}},
    {$lookup: {from: "stop", localField: "stops", foreignField: "code", as: "stops"}}
    ])

```

```

tours2> db.route.aggregate([
... {$sample:{size: 5}},
... {$lookup: {from: "stop", localField: "stops", foreignField: "code", as: "stops"}}
... ])
[
  {
    _id: ObjectId('6654d90013c4802a27a3282f'),
    name: 'route48550',
    price: 283,
    totalKm: 11.94,
    stops: [
      {
        _id: ObjectId('6654d8e013c4802a27a26a25'),
        name: 'Hotel Castelar',
        code: 18,
        description: 'Stop number 18'
      },
      {
        _id: ObjectId('6654d8e013c4802a27a26a4d'),
        name: 'Museo de Artes Plásticas Sívori',
        code: 58,
        description: 'Stop number 58'
      }
    ]
  },
  {
    _id: ObjectId('6654d8ec13c4802a27a2b063'),
    name: 'route17882',
    price: 704,
    totalKm: 15.4,
    stops: [
      {
        _id: ObjectId('6654d8e013c4802a27a26a1e'),

```

c) Obtenga la información de las Routes (incluyendo la de sus Stops) que tengan un precio mayor o igual a 900

```

db.route.aggregate([
{ $match: {price:{$gte: 900}}},
{$lookup: { from: "stop",localField: "stops", foreignField: "code", as: "stops"}}
])

```

```

tours2> db.route.aggregate([
... { $match: {price:{$gte: 900}}}},
... {$lookup: { from: "stop",localField: "stops", foreignField: "code", as:
"stops"}}
... ])
[
  {
    _id: ObjectId('6654d8e113c4802a27a26a8c'),
    name: 'route3',
    price: 931,
    totalKm: 17.19,
    stops: [
      {
        _id: ObjectId('6654d8e013c4802a27a26a44'),
        name: 'Cementerio de la Recoleta',
        code: 49,
        desciprion: 'Stop number 49'
      },
      {
        _id: ObjectId('6654d8e013c4802a27a26a46'),
        name: 'Obelisco',
        code: 51,
        desciprion: 'Stop number 51'
      },
      {
        _id: ObjectId('6654d8e013c4802a27a26a49'),
        name: 'Galerías Pacífico',
        code: 54,
        desciprion: 'Stop number 54'
      },
      {
        _id: ObjectId('6654d8e113c4802a27a26a79'),
        name: 'Puente de la Mujer',
        code: 102,
        desciprion: 'Stop number 102'
      }
    ]
  },
  {
    _id: ObjectId('6654d8e113c4802a27a26a97'),
    name: 'route14',
    price: 950,
    totalKm: 17.19,
    stops: [
      {
        _id: ObjectId('6654d8e013c4802a27a26a44'),
        name: 'Cementerio de la Recoleta',
        code: 49,
        desciprion: 'Stop number 49'
      },
      {
        _id: ObjectId('6654d8e013c4802a27a26a46'),
        name: 'Obelisco',
        code: 51,
        desciprion: 'Stop number 51'
      },
      {
        _id: ObjectId('6654d8e013c4802a27a26a49'),
        name: 'Galerías Pacífico',
        code: 54,
        desciprion: 'Stop number 54'
      },
      {
        _id: ObjectId('6654d8e113c4802a27a26a79'),
        name: 'Puente de la Mujer',
        code: 102,
        desciprion: 'Stop number 102'
      }
    ]
  }
]

```

d) Obtenga la información de las Routes que tengan 5 Stops o más.

```

db.route.aggregate([
{ $match: {$expr: {$gte: [{$size: "$stops"},5]}}}

```

```
tours2> db.route.aggregate([
... { $match: { $expr: { $gte: [{ $size: "$stops" }, 5] } } }
... ])
[
  {
    _id: ObjectId('6654d8e113c4802a27a26a91'),
    name: 'route8',
    price: 654,
    totalKm: 9.43,
    stops: [ 66, 5, 85, 91, 16 ]
  },
  {
    _id: ObjectId('6654d8e113c4802a27a26a92'),
    name: 'route9',
    price: 201,
    totalKm: 5.4,
    stops: [ 70, 118, 26, 115, 99 ]
  },
  {
    _id: ObjectId('6654d8e113c4802a27a26a93'),
    name: 'route10',
    price: 419,
    totalKm: 13.59,
    stops: [ 15, 20, 104, 69, 64 ]
  },
  {
    ...
  }
]
```

e) Obtenga la información de las Routes que tengan incluido en su nombre el string "111".

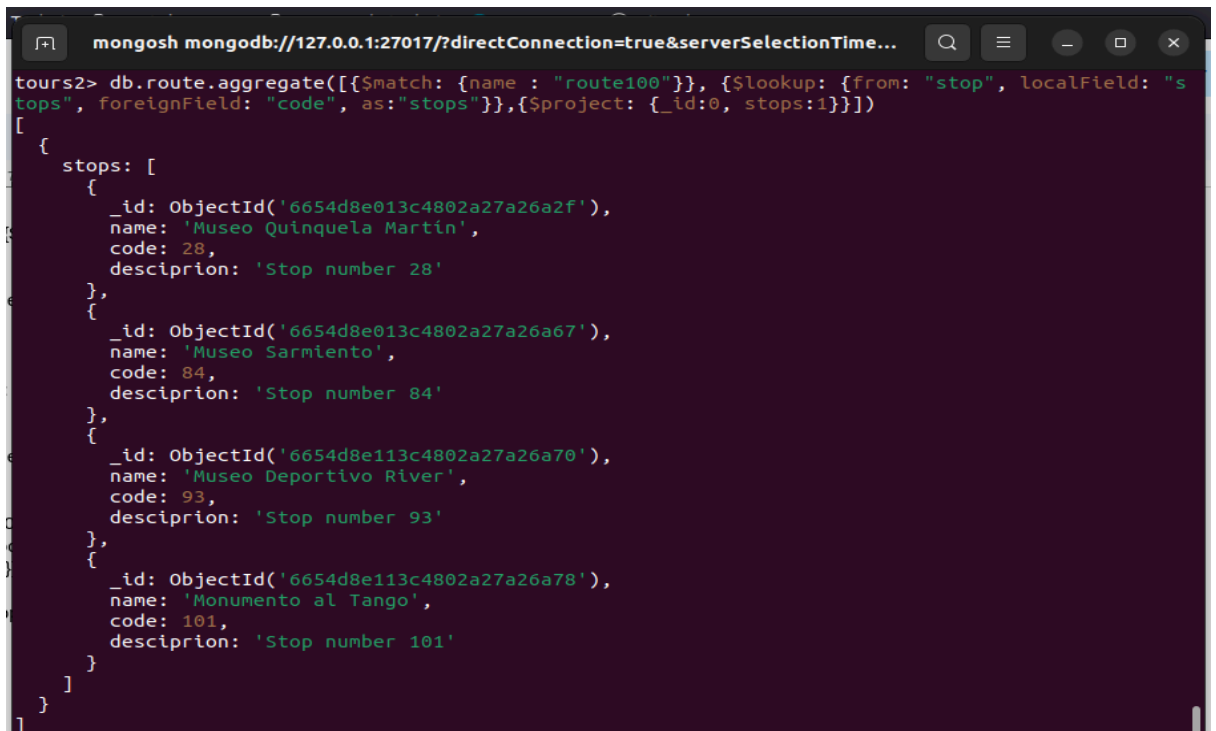
```
db.route.aggregate([
  { $match: {name:{$regex:  /111/}}}
])
```

```
tours2> db.route.aggregate([
... { $match: {name:{$regex: /111/}}}
... ])
[
  {
    _id: ObjectId('6654d8e113c4802a27a26af8'),
    name: 'route111',
    price: 183,
    totalKm: 10.57,
    stops: [ 69, 35, 90, 17, 54 ]
  },
  {
    _id: ObjectId('6654d8e213c4802a27a26edf'),
    name: 'route1110',
    price: 985,
    totalKm: 15.49,
    stops: [ 24, 13, 105, 115, 73 ]
  },
  {
    _id: ObjectId('6654d8e213c4802a27a26ee0'),
    name: 'route1111',
    price: 381,
    totalKm: 5.55,
    stops: [ 16, 18, 94, 71, 81 ]
  },
  {
    _id: ObjectId('6654d8e213c4802a27a26ee1'),
    name: 'route1112',
    price: 422,
    totalKm: 9.51,
    stops: [ 15, 14 ]
  },
]
```

- f) Obtenga solo las Stops de la Route con nombre "Route100"

```
db.route.aggregate([
  {$match: {name : "route100"}},
  {$lookup: {from: "stop", localField: "stops", foreignField: "code", as:"stops"}},
  {$project: {_id:0, stops:1}}])
```

Esto me devuelve las Stops de la ruta "route100" como un arreglo dentro de un campo llamado stops



```
mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTime...
tours2> db.route.aggregate([{$match: {name : "route100"}}, {$lookup: {from: "stop", localField: "s
tops", foreignField: "code", as:"stops"}},{$project: {_id:0, stops:1}}])
[
  {
    stops: [
      {
        _id: ObjectId('6654d8e013c4802a27a26a2f'),
        name: 'Museo Quinquela Martín',
        code: 28,
        descripion: 'Stop number 28'
      },
      {
        _id: ObjectId('6654d8e013c4802a27a26a67'),
        name: 'Museo Sarmiento',
        code: 84,
        descripion: 'Stop number 84'
      },
      {
        _id: ObjectId('6654d8e113c4802a27a26a70'),
        name: 'Museo Deportivo River',
        code: 93,
        descripion: 'Stop number 93'
      },
      {
        _id: ObjectId('6654d8e113c4802a27a26a78'),
        name: 'Monumento al Tango',
        code: 101,
        descripion: 'Stop number 101'
      }
    ]
  }
]
```

```
db.route.aggregate([
  {$match: {name : "route100"}},
  {$lookup: {from: "stop", localField: "stops", foreignField: "code", as:"stops"}},
  {$unwind: "$stops"},
  {$replaceRoot: {newRoot: "$stops"}}
])
```

Esa consulta me las devuelve como documentos independientes


```
tours2> db.route.aggregate([{$match: {name : "route100"}}, {$lookup: {from: "stop", localField: "stops", foreignField: "code", as:"stops"}},{$unwind: "$stops"},{$replaceRoot: {newRoot: "$stops"}}])
[
  {
    _id: ObjectId('6654d8e013c4802a27a26a2f'),
    name: 'Museo Quinquela Martin',
    code: 28,
    desciprion: 'Stop number 28'
  },
  {
    _id: ObjectId('6654d8e013c4802a27a26a67'),
    name: 'Museo Sarmiento',
    code: 84,
    desciprion: 'Stop number 84'
  },
  {
    _id: ObjectId('6654d8e113c4802a27a26a70'),
    name: 'Museo Deportivo River',
    code: 93,
    desciprion: 'Stop number 93'
  },
  {
    _id: ObjectId('6654d8e113c4802a27a26a78'),
    name: 'Monumento al Tango',
    code: 101,
    desciprion: 'Stop number 101'
  }
]
```

g) Obtenga la información del Stop que más apariciones tiene en Routes.

```
db.route.aggregate([
  { $unwind: "$stops" },
  {$sortByCount: "$stops"},
  {$limit: 1},
  {$lookup: {from: "stop", localField:"_id", foreignField:"code", as:"stop"}},
  {$unwind: "$stop"},
  {$replaceRoot: {newRoot: "$stop"}}
])
```

```
tours2> db.route.aggregate([
... { $unwind: "$stops" },
... {$sortByCount: "$stops"},
... {$limit: 1},
... {$lookup: {from: "stop", localField:"_id", foreignField:"code", as:"stop"}},
... {$unwind: "$stop"},
... {$replaceRoot: {newRoot: "$stop"}}
... ])
[
  {
    _id: ObjectId('6654d8e013c4802a27a26a64'),
    name: 'Museo Enrique Larreta',
    code: 81,
    desciprion: 'Stop number 81'
  }
]
```

h) Obtenga las Route que tengan un precio inferior a 150. A ellos agregué una nueva propiedad que especifique la cantidad de Stops que posee la Route. Cree una nueva colección llamada "rutas_economicas" y almacene estos elementos.

```
db.route.aggregate([
  {$match: {price: {$lt: 150}}},
```

```

    {$addFields: {totalStops: {$size: "$stops"}}},
    {$merge: "rutas_economicas"}
  ])

```

```

Type "it" for more
tours2> db.route.aggregate([{$match: {price: {$lt: 150}}}, {$addFields: {totalStops: {$size: "$stops"}}}, {$merge: "rutas_economicas"}])
]
tours2> db.rutas_economicas.find()
[
  {
    _id: ObjectId('6654d8e113c4802a27a26a95'),
    name: 'route12',
    price: 145,
    stops: [ 104, 110, 48, 50 ],
    totalKm: 15.45,
    totalStops: 4
  },
  {
    _id: ObjectId('6654d8e113c4802a27a26a9d'),
    name: 'route20',
    price: 109,
    stops: [ 74, 46, 44 ],
    totalKm: 9.7,
    totalStops: 3
  },
  {
    _id: ObjectId('6654d8e113c4802a27a26aab'),
    name: 'route34',
    price: 137,
    stops: [ 110, 5, 108, 35, 66 ],
    totalKm: 2.45,
    totalStops: 5
  },
  {
    _id: ObjectId('6654d8e113c4802a27a26ab7'),
    name: 'route46',

```

- i) Por cada Stop existente en su colección, calcule el precio promedio de las Routes que la incluyen

```

db.route.aggregate([
  { $unwind: "$stops" },

```

```

    { $group: { _id: "$stops", precioPromedio: { $avg: "$price" } } }
  })
}

tours2> db.route.aggregate([
...   { $unwind: "$stops" },
...   { $group: { _id: "$stops", precioPromedio: { $avg: "$price" } } }
... ])
[
  { _id: 110, precioPromedio: 548.4975507347796 },
  { _id: 13, precioPromedio: 551.1151473612063 },
  { _id: 93, precioPromedio: 553.6490156143924 },
  { _id: 37, precioPromedio: 551.9177718832891 },
  { _id: 31, precioPromedio: 554.7320217096336 },
  { _id: 99, precioPromedio: 544.1097074468086 },
  { _id: 10, precioPromedio: 537.8893528183717 },
  { _id: 84, precioPromedio: 551.3844103930713 },
  { _id: 65, precioPromedio: 551.382276843467 },
  { _id: 107, precioPromedio: 550.6231292517007 },
  { _id: 27, precioPromedio: 550.1591230551627 },
  { _id: 39, precioPromedio: 552.1550654720883 },
  { _id: 105, precioPromedio: 545.0237623762376 },
  { _id: 102, precioPromedio: 555.8469251336899 },
  { _id: 9, precioPromedio: 554.9601100412655 },
  { _id: 72, precioPromedio: 541.1248303934871 },
  { _id: 63, precioPromedio: 551.3634453781513 },
  { _id: 16, precioPromedio: 559.4059065934066 },
  { _id: 58, precioPromedio: 539.1511142061281 },
  { _id: 59, precioPromedio: 562.6357744653272 }
]

```

Parte 4: Indices

14)

```
tours3> load('/home/pepe-notebook/Facultad/2024/BBDD2/tp3/generator2.js')
true
tours3> showcollections()
ReferenceError: showcollections is not defined
tours3> show collections
routes
stops
tours3> db.routes.find({})
[
  {
    _id: ObjectId('6657ad30b91a9a897aa4b403'),
    name: 'Route 1',
    price: 498,
    startPoint: {
      type: 'Point',
      coordinates: [ -58.54519956469117, -34.65176665515504 ]
    }
  },
  {
    _id: ObjectId('6657ad32b91a9a897aa4b404'),
    name: 'Route 2',
    price: 748,
    startPoint: {
      type: 'Point',
      coordinates: [ -58.59051980043147, -34.62458094266242 ]
    }
  },
  {

```

15)

```
tours3> db.routes.getIndexes()
[ { v: 2, key: { _id: 1 }, name: '_id_' } ]
```

16) tours3> db.stops.find({name: /11/}).explain("executionStats")

```

executionStats: {
  executionSuccess: true,
  nReturned: 7373,
  executionTimeMillis: 140,
  totalKeysExamined: 0,
  totalDocsExamined: 161187,
  executionStages: {
    stage: 'COLLSCAN',
    filter: { name: { '$regex': '11' } },
    nReturned: 7373,
    executionTimeMillisEstimate: 25,
    works: 161188,
    advanced: 7373,
    needTime: 153814,
    needYield: 0,
    saveState: 161,
    restoreState: 161,
    isEOF: 1,
    direction: 'forward',
    docsExamined: 161187
  }
}

```

17) **tours3> db.stops.createIndex({"name": "text"})**
 name_text
tours3> db.stops.find({name: /11/}).explain("executionStats")

```

executionStats: {
  executionSuccess: true,
  nReturned: 7373,
  executionTimeMillis: 140,
  totalKeysExamined: 0,
  totalDocsExamined: 161187,
  executionStages: {
    stage: 'COLLSCAN',
    filter: { name: { '$regex': '11' } },
    nReturned: 7373,
    executionTimeMillisEstimate: 14,
    works: 161188,
    advanced: 7373,
    needTime: 153814,
    needYield: 0,
    saveState: 161,
    restoreState: 161,
    isEOF: 1,
    direction: 'forward',
    docsExamined: 161187
  }
}

```

utilizando esa función no cambia el comportamiento, pero si se utiliza \$text si

tours3> db.stops.find({ \$text: {\$search: "11"}}).explain("executionStats")

```

executionStats: {
  executionSuccess: true,
  nReturned: 6,
  executionTimeMillis: 0,
  totalKeysExamined: 6,
  totalDocsExamined: 6,
  executionStages: {
    stage: 'COLLSCAN',
    filter: { $text: { $search: '11' } },
    nReturned: 6,
    executionTimeMillisEstimate: 0,
    works: 6,
    advanced: 6,
    needTime: 0,
    needYield: 0,
    saveState: 0,
    restoreState: 0,
    isEOF: 1,
    direction: 'forward',
    docsExamined: 6
  }
}

```

18)

```
tours3> var cabaPoligono = {
...   "type": "MultiPolygon",
...   "coordinates": [[[
...     [-58.46305847167969, -34.53456089748654],
...     [-58.49979400634765, -34.54983198845187],
...     [-58.532066345214844, -34.614561581608186],
...     [-58.528633117675774, -34.6538270014492],
...     [-58.48674774169922, -34.68742794931483],
...     [-58.479881286621094, -34.68206400648744],
...     [-58.46855163574218, -34.65297974261105],
...     [-58.465118408203125, -34.64733112904415],
...     [-58.4585952758789, -34.63998735602951],
...     [-58.45344543457032, -34.63603274732642],
...     [-58.447265625, -34.63575026806082],
...     [-58.438339233398445, -34.63038297923296],
...     [-58.38100433349609, -34.62162507826766],
...     [-58.38237762451171, -34.59251960889388],
...     [-58.378944396972656, -34.5843230246475],
...     [-58.46305847167969, -34.53456089748654]
...   ]]]
... }

tours3> db.routes.find({ startPoint: { $geoWithin: { $geometry: cabaPoligono }}})
```

```
executionStats: {
  executionSuccess: true,
  nReturned: 21736,
  executionTimeMillis: 505,
  totalKeysExamined: 0,
  totalDocsExamined: 100000,
  executionStages: {
    stage: 'COLLSCAN',
    filter: {
      startPoint: {
        '$geoWithin': {
          '$geometry': {
            type: 'MultiPolygon',
            coordinates: [[[
              [-58.46305847167969, -34.53456089748654 ],
              [-58.49979400634765, -34.54983198845187 ],
              [-58.532066345214844, -34.614561581608186 ],
              [-58.528633117675774, -34.6538270014492 ],
              [-58.48674774169922, -34.68742794931483 ],
              [-58.479881286621094, -34.68206400648744 ],
              [-58.46855163574218, -34.65297974261105 ],
              [-58.465118408203125, -34.64733112904415 ],
              [-58.4585952758789, -34.63998735602951 ],
              [-58.45344543457032, -34.63603274732642 ],
              [-58.447265625, -34.63575026806082 ],
              [-58.438339233398445, -34.63038297923296 ],
              [-58.38100433349609, -34.62162507826766 ],
```

```

executionStats: {
  executionSuccess: true,
  nReturned: 21736,
  executionTimeMillis: 129,
  totalKeysExamined: 27503,
  totalDocsExamined: 27484,
  executionStages: {
    stage: 'FETCH',
    filter: {
      startPoint: {
        '$geoWithin': {
          '$geometry': {
            type: 'MultiPolygon',
            coordinates: [[[[ -58.46305847167969, -34.53456089748654 ],
              [ -58.49979400634765, -34.54983198845187 ],
              [ -58.532066345214844, -34.614561581608186 ],
              [ -58.528633117675774, -34.6538270014492 ],
              [ -58.48674774169922, -34.68742794931483 ],
              [ -58.479881286621094, -34.68206400648744 ],
              [ -58.46855163574218, -34.65297974261105 ],
              [ -58.465118408203125, -34.64733112904415 ],
              [ -58.4585952758789, -34.63998735602951 ],
              [ -58.45344543457032, -34.63603274732642 ],
              [ -58.447265625, -34.63575026806082 ],
              [ -58.438339233398445, -34.63038297923296 ],
              [ -58.38100433349609, -34.62162507826766 ],
              [ -58.38237762451171, -34.59251960889388 ],
              [ -58.378944396972656, -34.5843230246475 ],
              [ -58.46305847167969, -34.53456089748654 ]]]]]]],
            nReturned: 21736.

```

executionTimeMillisEstimate: 38,
works: 27504,
advanced: 21736,
needTime: 5767,
needYield: 0,
saveState: 27,
restoreState: 27,
isEOF: 1,
docsExamined: 27484,
alreadyHasObj: 0,
inputStage: {
stage: 'IXSCAN',
nReturned: 27484,
executionTimeMillisEstimate: 7,
works: 27504,
advanced: 27484,
needTime: 19,
needYield: 0,
saveState: 27,
restoreState: 27,
isEOF: 1,
keyPattern: { startPoint: '2dsphere' },
indexName: 'startPoint_2dsphere',
isMultiKey: false,
multiKeyPaths: { startPoint: [] },
isUnique: false,
isSparse: false,
isPartial: false,
indexVersion: 2,
direction: 'forward',
indexBounds: {
startPoint: [['-7710162562058289152, -7710162562058289152'],
['-7660622966157213696, -7660622966157213696'],
['-7657245266436685824, -7657245266436685824'],
['-7657051752390197248, -7657051752390197248'],
['-7657047354343686144, -7657047354343686144'],
['-7657047354343686143, -7657046804587872257'],
['-7657046254832058368, -7657046254832058368'],
['-7657046220472319999, -7657046186112581633'],
['-7657046186112581632, -7657046186112581632'],
['-7657045979954151424, -7657045979954151424'],
['-7657045911234674688, -7657045911234674688'],
['-7657045876874936319, -7657045842515197953'],
['-7657045842515197951, -7657045705076244481'],
['-7657045705076244479, -7657045155320430593'],
['-7657045155320430591, -7657044605564616705'],
['-7657044605564616703, -7657044055808802817'],
['-7657044055808802816, -7657044055808802816'],

['-7657044055808802815, -7657044021449064449'],
['-7657043987089326080, -7657043987089326080'],
['-7657043780930895872, -7657043780930895872'],
['-7657043506052988927, -7657042956297175041'],
['-7657034160204152832, -7657034160204152832'],
['-7657025295391653888, -7657025295391653888'],
['-7657025243852046336, -7657025243852046336'],
['-7657025230967144448, -7657025230967144448'],
['-7657025227745918976, -7657025227745918976'],
['-7657025226940612608, -7657025226940612608'],
['-7657025226739286016, -7657025226739286016'],
['-7657025226688954368, -7657025226688954368'],
['-7657025226680565759, -7657025226672177153'],
['-7657025226672177151, -7657025089233223681'],
['-7657025089233223680, -7657025089233223680'],
['-7657025089233223679, -7657024951794270209'],
['-7657024539477409792, -7657024539477409792'],
['-7657024402038456319, -7657024264599502849'],
['-7657024264599502848, -7657024264599502848'],
['-7657024264599502847, -7657023714843688961'],
['-7657023714843688959, -7657023165087875073'],
['-7657023165087875071, -7657022615332061185'],
['-7657022615332061183, -7657022065576247297'],
['-7657022065576247296, -7657022065576247296'],
['-7657022065576247295, -7657021928137293825'],
['-7657021928137293823, -7657021790698340353'],
['-7657021790698340352, -7657021790698340352'],
['-7657021240942526464, -7657021240942526464'],
['-7657021172223049728, -7657021172223049728'],
['-7657021155043180544, -7657021155043180544'],
['-7657021155043180543, -7657021146453245953'],
['-7657020966064619520, -7657020966064619520'],
['-7657016568018108416, -7657016568018108416'],
['-7656963791459975168, -7656963791459975168'],
['-7656119366529843200, -7656119366529843200']]],
keysExamined: 27503,
seeks: 20,
dupsTested: 0,
dupsDropped: 0}}