

MongoDB

BASE DE DATOS NOSQL

Contenidos

- Relaciones
- GridFS
- Replicación
- Sharding
- MapReduce
- Administración y Optimización

Relaciones por Referencia


- En determinadas ocasiones es necesario almacenar la información en documentos separados pero relacionados, normalmente en colecciones o bases de datos separadas.
- MongoDB utiliza dos métodos para relacionar documentos separados:
 - **Referencias manuales:** Se almacena un campo `_id` de un documento en otro documento como referencia. La aplicación es la responsable de lanzar una segunda consulta para recoger la información relacionada.
 - **DBRefs:** Referencias de un documento a otro usando el `_id` del primer documento, nombre de colección y base de dato (opcional) Se utiliza sobre todo para localizar documentos en más de una colección desde una única colección (relaciones 1:n)
 - La aplicación debe realizar queries adicionales para devolver el document referenciado.

```
{ "$ref" : <value>, "$id" : <value>, "$db" : <value> }
```

- **Ejemplo de aplicación:** Visualizar autores de comentarios de una publicación en el caso de que los datos de usuario puedan cambiar y sea una consulta recurrente

Relaciones por Referencia: Ejemplo

```
{
  "_id": ObjectId("53402597d852426020000002"),
  "address": {
    "$ref": "address_home",
    "$id": ObjectId("534009e4d852427820000002"),
    "$db": "w3cschoolcc"},
  "contact": "987654321",
  "dob": "01-01-1991",
  "name": "Tom Benzamin"
}
```



```
>var user = db.users.findOne({"name":"Tom Benzamin"})
>var dbRef = user.address
>db[dbRef.$ref].findOne({"_id":(dbRef.$id)})
```

```
{
  "_id" : ObjectId("534009e4d852427820000002"),
  "building" : "22 A, Indiana Apt",
  "pincode" : 123456,
  "city" : "Los Angeles",
  "state" : "California"
}
```

Relaciones por Referencia: \$lookup y \$graphLookup

- Etapas del pipe aggregate para obtener información de otros documentos relacionados por DBRefs:

- **\$lookup:**

```
{
  $lookup:
  {
    from: <collection to join>,
    localField: <field from the input documents>,
    foreignField: <field from the documents of the "from" collection>,
    as: <output array field>
  }
}
```

- **\$graphLookup:** Realiza una búsqueda recursiva en una colección

-

```
{
  $graphLookup: {
    from: <collection>,
    startWith: <expression>,
    connectFromField: <string>,
    connectToField: <string>,
    as: <string>,
    maxDepth: <number>,
    depthField: <string>,
    restrictSearchWithMatch: <document>
  }
}
```

Relaciones por Referencia: \$lookup

- Ejemplo:
- Orden de compra y detalle:

```
db.orders.insert([
  { "_id" : 1, "item" : "almonds", "price" : 12, "quantity" : 2 },
  { "_id" : 2, "item" : "pecans", "price" : 20, "quantity" : 1 },
  { "_id" : 3 }
])
```

```
db.orders.aggregate([
  {
    $lookup:
    {
      from: "inventory",
      localField: "item",
      foreignField: "sku",
      as: "inventory_docs"
    }
  }
])
```

```
db.inventory.insert([
  { "_id" : 1, "sku" : "almonds", description: "product 1",
    "instock" : 120 },
  { "_id" : 2, "sku" : "bread", description: "product 2", "instock" :
    80 },
  { "_id" : 3, "sku" : "cashews", description: "product 3", "instock"
    : 60 },
  { "_id" : 4, "sku" : "pecans", description: "product 4", "instock" :
    70 },
  { "_id" : 5, "sku": null, description: "Incomplete" },
  { "_id" : 6 }
])
```

```
SELECT *, inventory_docs
FROM orders
WHERE inventory_docs IN (SELECT *
FROM inventory
WHERE sku= orders.item);
```

GridFS

- Es un protocolo para almacenar y recuperar archivos que exceden el tamaño máximo de documentos de 16 MB. En vez de almacenar un archivo en un único documento, se divide el archivo en partes (*chunks*) y se almacena cada parte en documentos separados. Por defecto, cada trozo está limitado a 255 k.
- Se usan dos colecciones para almacenar los archivos, una para almacenar los archivos de trozos y otro para almacenar archivos de metadatos asociados.
- Cuando se realiza una consulta a un almacén GridFS por un archivo, se reensamblan los trozos necesarios.
- Se pueden realizar consultas sobre secciones arbitrarias de los archivos
- No hace falta cargar todo el archivo en memoria para acceder a él.

GridFS

- Para trabajar con GridFS desde MongoDB se utiliza el comando **mongofiles**
- Ejemplo: guardar un fichero c:\video.mp4

```
mongofiles -d media put c:\mongodb\video.mp4
```

- Desde la db: show collections:

```
> show collections
fs.chunks
fs.files
multimedia
```

- Mostrar el contenido del documento:

```
db.fs.files.find()
```

- Para recuperar un fichero y guardarlo en un directorio determinado:

```
mongofiles -d media get c:\video.mp4
```


MapReduce

- Map-reduce es un framework creado por Google, y pensado para realizar operaciones de forma paralela sobre grandes colecciones de datos.
- Las operaciones map-reduce constan de tres partes:
- Una etapa de **map**, en la que se procesa cada documento y se emite uno o varios objetos por cada documento procesado.
 - para cada dato de origen se genera una lista de tuplas clave-valor que se pasa a la etapa *reduce*.
- Una etapa **reduce** en la que se combinan las salidas de la etapa anterior.
 - se realizan operaciones para cada elemento de la lista de pares
- Una tercera etapa opcional, finalize, en la que se permite realizar algunas modificaciones adicionales a las salidas de la etapa reduce.

MapReduce

- Las funciones involucradas en map-reduce se implementan en JavaScript en la consola de mongo.
- Aunque la flexibilidad es total al poderse implementar cualquier tipo de función que se desee, en general este modo de funcionamiento es menos eficiente y más complejo que la agregación mediante tuberías. Además, en general se pueden conseguir los mismos resultados con ambos modos de funcionamiento.

MapReduce

- SINTAXIS:

```
db.collection.mapReduce(
```

```
  <map>,
```

Una función JavaScript que asocia o mapea los valores con claves y emite pares clave valor.

```
  <reduce>,
```

Una función JavaScript que reduce los resultados del mapeo a un único objeto que contiene todos los valores asociados a una clave concreta.

```
{
```

```
  out: <collection>,
```

Especifica dónde agrupar la salida de la operación map-reduce. Es posible especificar una colección o únicamente devolver los resultados en la consola.

```
  query: <document>,
```

```
  sort: <document>,
```

```
  limit: <number>,
```

```
  finalize: <function>,
```

Función JavaScript que determina qué operaciones aplicar a la salida de la función reduce.

```
  scope: <document>,
```

```
  jsMode: <boolean>,
```

```
  verbose: <boolean>,
```

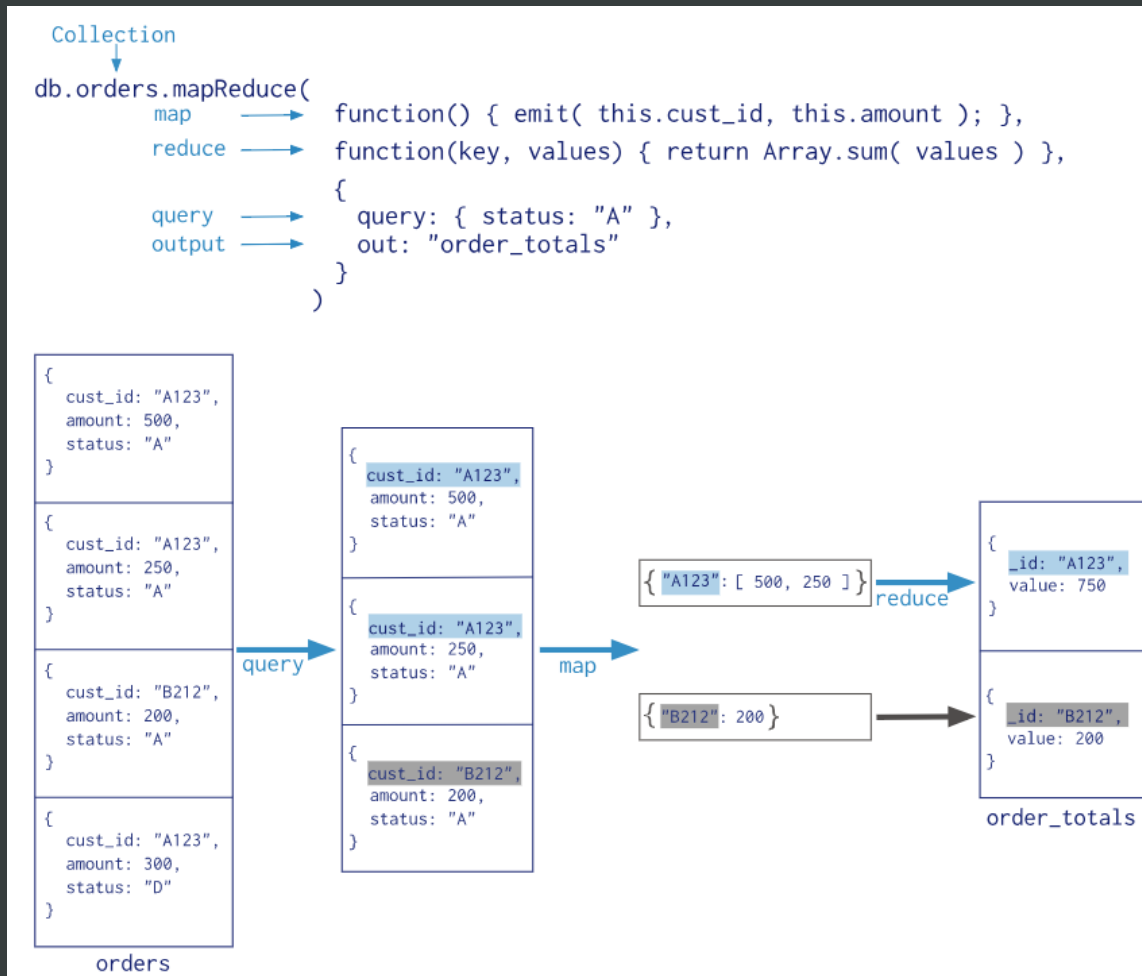
```
  bypassDocumentValidation: <boolean>
```

```
}
```

```
)
```

MapReduce

- Ejemplo página oficial:



1. Cargar el fichero people.json en una bd personas
2. Generar por map-reduce una colección de personas (campo name) y su puntuación (campo points)
3. Comprobar el resultado sobre la nueva colección
4. Generar el mismo resultado mediante una función aggregate

1. Obtener por map-reduce una colección de directores y oscars de la colección movieDetails de la bd movies.