

BBDD NOSQL Práctica 1 - Uso de MongoDB

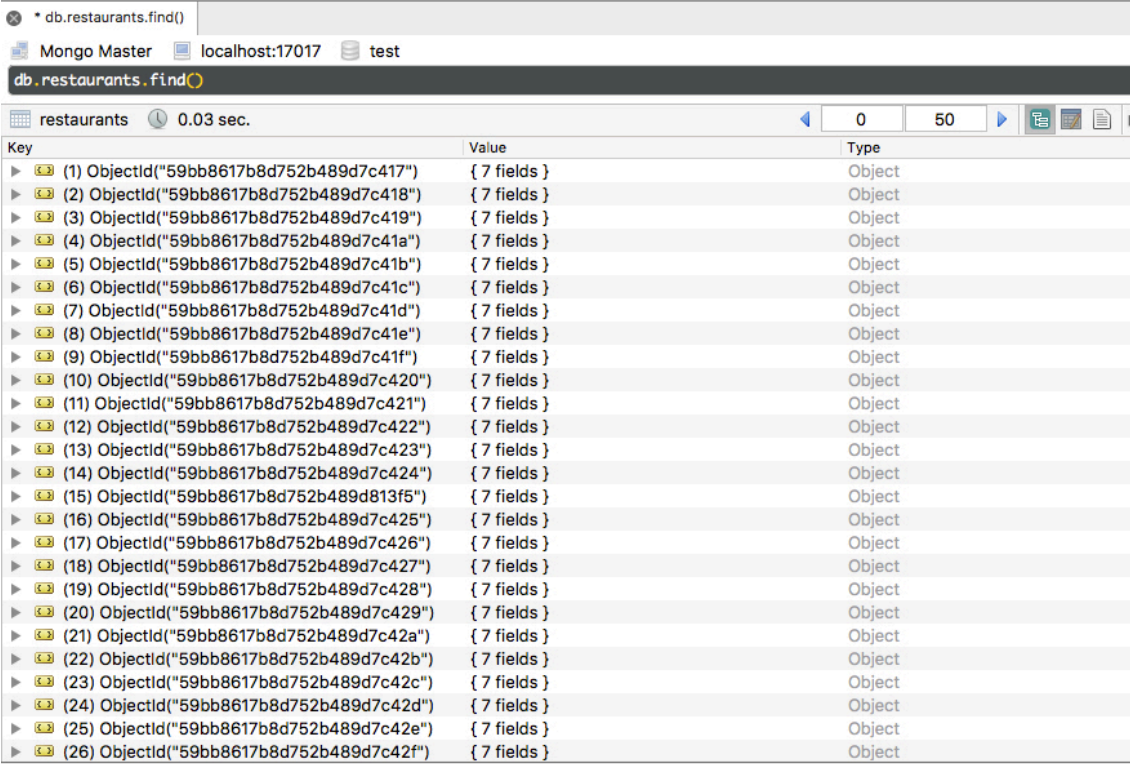
José Manuel Bustos Muñoz

Ejercicios

Los ejercicios han sido realizados sobre la colección **restaurants** y utilizando la herramienta “Robo 3T”.

1. (3 puntos) Analizar qué estructura tiene un documento de la colección **restaurants**, indicando: campos, tipo de datos de cada campo, si hay campos que únicamente admiten un conjunto de valores, etc.

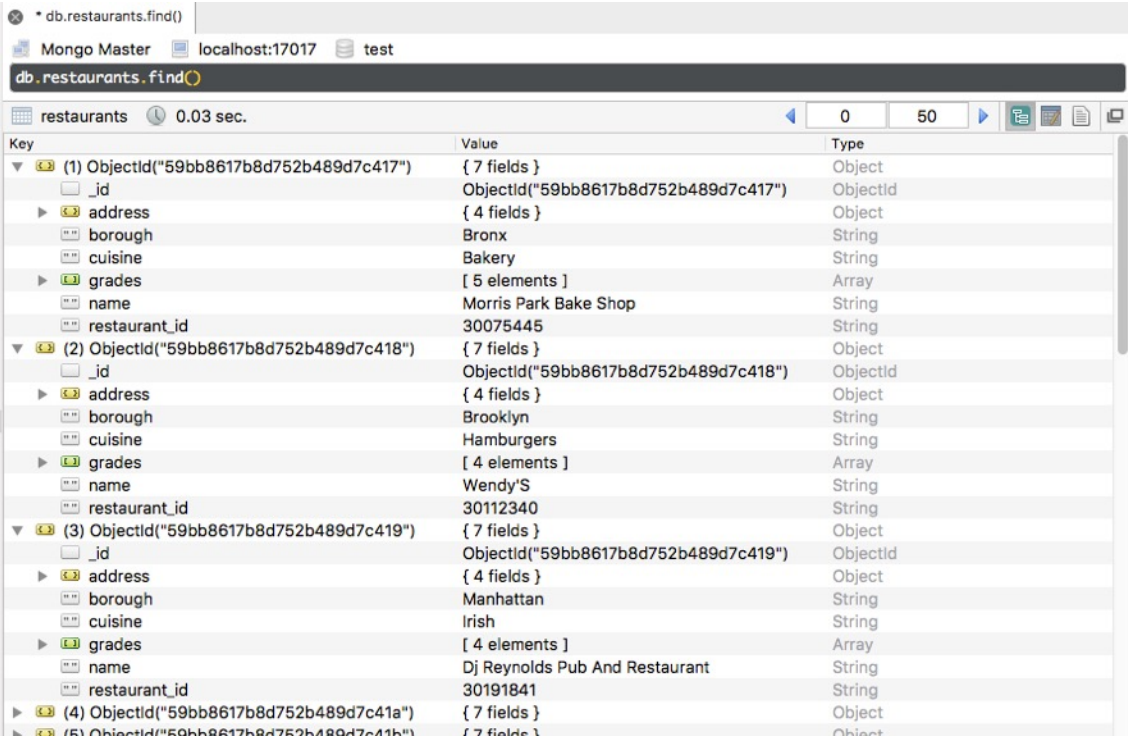
Con `db.restaurants.find()` listamos todos los objetos de la colección. Se puede ver que cada objeto tiene 7 campos o elementos.



Key	Value	Type
(1) ObjectId("59bb8617b8d752b489d7c417")	{ 7 fields }	Object
(2) ObjectId("59bb8617b8d752b489d7c418")	{ 7 fields }	Object
(3) ObjectId("59bb8617b8d752b489d7c419")	{ 7 fields }	Object
(4) ObjectId("59bb8617b8d752b489d7c41a")	{ 7 fields }	Object
(5) ObjectId("59bb8617b8d752b489d7c41b")	{ 7 fields }	Object
(6) ObjectId("59bb8617b8d752b489d7c41c")	{ 7 fields }	Object
(7) ObjectId("59bb8617b8d752b489d7c41d")	{ 7 fields }	Object
(8) ObjectId("59bb8617b8d752b489d7c41e")	{ 7 fields }	Object
(9) ObjectId("59bb8617b8d752b489d7c41f")	{ 7 fields }	Object
(10) ObjectId("59bb8617b8d752b489d7c420")	{ 7 fields }	Object
(11) ObjectId("59bb8617b8d752b489d7c421")	{ 7 fields }	Object
(12) ObjectId("59bb8617b8d752b489d7c422")	{ 7 fields }	Object
(13) ObjectId("59bb8617b8d752b489d7c423")	{ 7 fields }	Object
(14) ObjectId("59bb8617b8d752b489d7c424")	{ 7 fields }	Object
(15) ObjectId("59bb8617b8d752b489d7c425")	{ 7 fields }	Object
(16) ObjectId("59bb8617b8d752b489d7c426")	{ 7 fields }	Object
(17) ObjectId("59bb8617b8d752b489d7c427")	{ 7 fields }	Object
(18) ObjectId("59bb8617b8d752b489d7c428")	{ 7 fields }	Object
(19) ObjectId("59bb8617b8d752b489d7c429")	{ 7 fields }	Object
(20) ObjectId("59bb8617b8d752b489d7c42a")	{ 7 fields }	Object
(21) ObjectId("59bb8617b8d752b489d7c42b")	{ 7 fields }	Object
(22) ObjectId("59bb8617b8d752b489d7c42c")	{ 7 fields }	Object
(23) ObjectId("59bb8617b8d752b489d7c42d")	{ 7 fields }	Object
(24) ObjectId("59bb8617b8d752b489d7c42e")	{ 7 fields }	Object
(25) ObjectId("59bb8617b8d752b489d7c42f")	{ 7 fields }	Object
(26) ObjectId("59bb8617b8d752b489d7c430")	{ 7 fields }	Object

Desplegamos un objeto, y analizamos sus campos:

- **_id**: campo de tipo objectId, actúa como identificador único del objeto.
- **address**: es un campo de tipo objeto, por lo tanto almacena otro documento dentro del propio objeto restaurante.
- **borough**: campo string que almacena el barrio donde está ubicado el restaurante.
- **cuisine**: es un campo string que almacena el tipo del restaurante.
- **grades**: es un campo de tipo array, por lo tanto una lista de x elementos.
- **name**: campo string que almacena el nombre del restaurante.
- **restaurant_id**: es un campo string que almacena otro identificador de cada restaurante.



Command: `db.restaurants.find()`

Test: localhost:17017

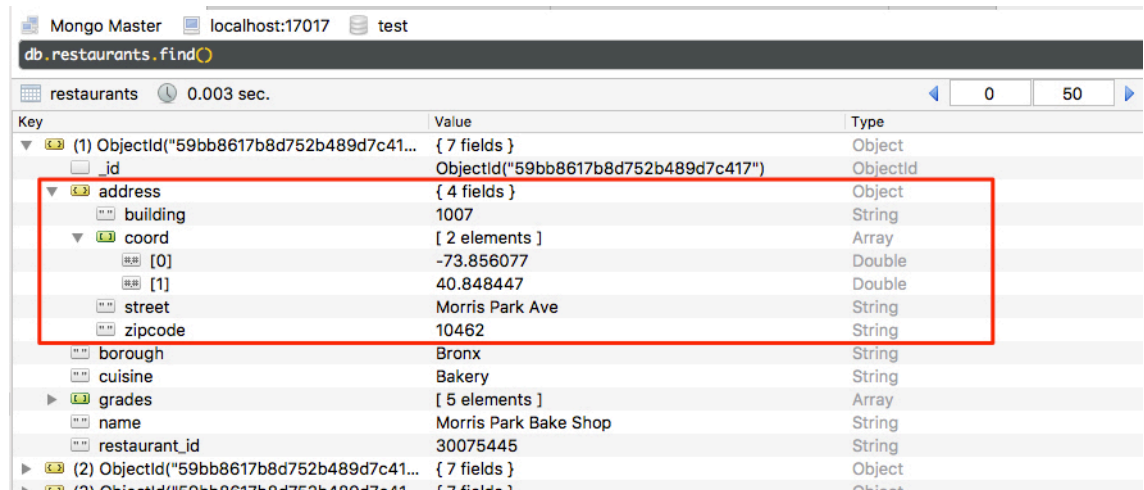
Query: `db.restaurants.find()`

restaurants 0.03 sec. 0 50

Key	Value	Type
(1) ObjectId("59bb8617b8d752b489d7c417")	{ 7 fields }	Object
_id	ObjectId("59bb8617b8d752b489d7c417")	ObjectId
address	{ 4 fields }	Object
borough	Bronx	String
cuisine	Bakery	String
grades	[5 elements]	Array
name	Morris Park Bake Shop	String
restaurant_id	30075445	String
(2) ObjectId("59bb8617b8d752b489d7c418")	{ 7 fields }	Object
_id	ObjectId("59bb8617b8d752b489d7c418")	ObjectId
address	{ 4 fields }	Object
borough	Brooklyn	String
cuisine	Hamburgers	String
grades	[4 elements]	Array
name	Wendy'S	String
restaurant_id	30112340	String
(3) ObjectId("59bb8617b8d752b489d7c419")	{ 7 fields }	Object
_id	ObjectId("59bb8617b8d752b489d7c419")	ObjectId
address	{ 4 fields }	Object
borough	Manhattan	String
cuisine	Irish	String
grades	[4 elements]	Array
name	Dj Reynolds Pub And Restaurant	String
restaurant_id	30191841	String
(4) ObjectId("59bb8617b8d752b489d7c41a")	{ 7 fields }	Object
(5) ObjectId("59bb8617b8d752b489d7c41b")	{ 7 fields }	Object

El objeto **address** tiene en su interior la siguiente estructura:

- **building**: dato de tipo string.
- **coord.**: dato de tipo array que almacena las coordenadas x e y de la dirección con formato double.
- **street**: dato de tipo string que almacena el nombre de la calle.
- **zipcode**: dato de tipo string que almacena el código postal.



Mongo Master localhost:17017 test

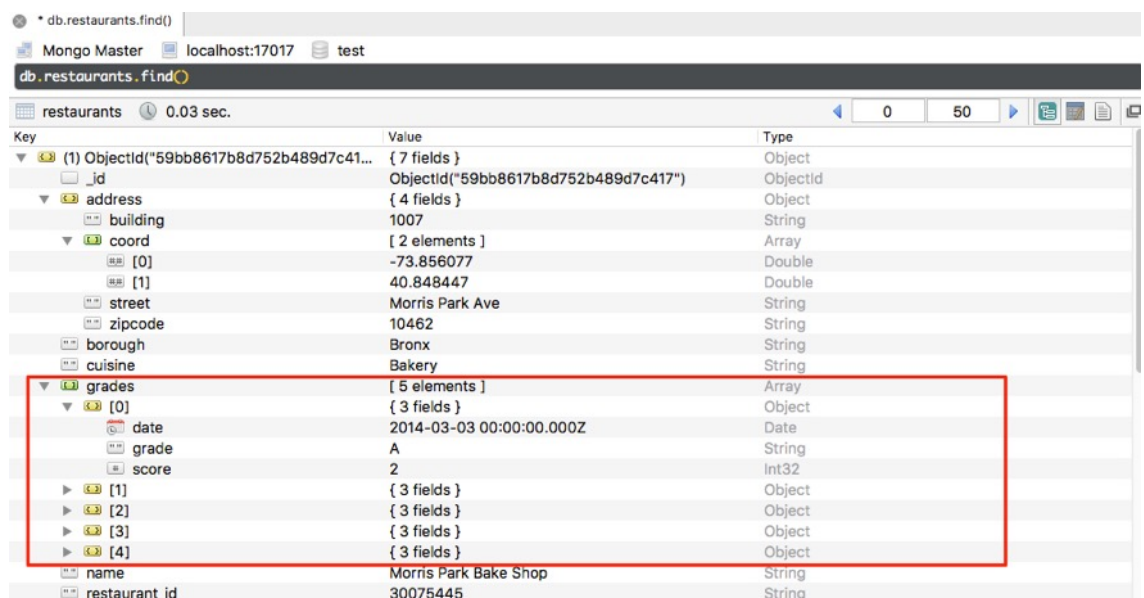
db.restaurants.find()

restaurants 0.003 sec.

Key	Value	Type
(1) ObjectId("59bb8617b8d752b489d7c41...")	{ 7 fields }	Object
_id	ObjectId("59bb8617b8d752b489d7c417")	ObjectId
address	{ 4 fields }	Object
building	1007	String
coord	[2 elements]	Array
[0]	-73.856077	Double
[1]	40.848447	Double
street	Morris Park Ave	String
zipcode	10462	String
borough	Bronx	String
cuisine	Bakery	String
grades	[5 elements]	Array
name	Morris Park Bake Shop	String
restaurant_id	30075445	String
(2) ObjectId("59bb8617b8d752b489d7c41...")	{ 7 fields }	Object
(3) ObjectId("59bb8617b8d752b489d7c41...")	{ 7 fields }	Object

El array **grades** contiene x elementos y cada uno de ellos tiene la siguiente estructura:

- **date**: tipo de dato fecha.
- **grade**: tipo de dato string para almacenar el grado del restaurante.
- **score**: tipo de dato numérico entero para almacenar la puntuación del restaurante.



* db.restaurants.find()

Mongo Master localhost:17017 test

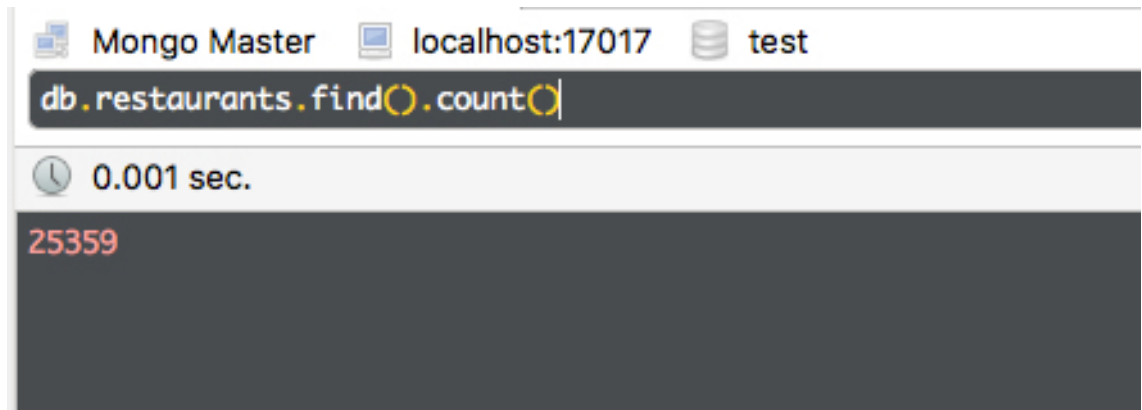
db.restaurants.find()

restaurants 0.03 sec.

Key	Value	Type
(1) ObjectId("59bb8617b8d752b489d7c41...")	{ 7 fields }	Object
_id	ObjectId("59bb8617b8d752b489d7c417")	ObjectId
address	{ 4 fields }	Object
building	1007	String
coord	[2 elements]	Array
[0]	-73.856077	Double
[1]	40.848447	Double
street	Morris Park Ave	String
zipcode	10462	String
borough	Bronx	String
cuisine	Bakery	String
grades	[5 elements]	Array
[0]	{ 3 fields }	Object
date	2014-03-03 00:00:00.000Z	Date
grade	A	String
score	2	Int32
[1]	{ 3 fields }	Object
[2]	{ 3 fields }	Object
[3]	{ 3 fields }	Object
[4]	{ 3 fields }	Object
name	Morris Park Bake Shop	String
restaurant_id	30075445	String

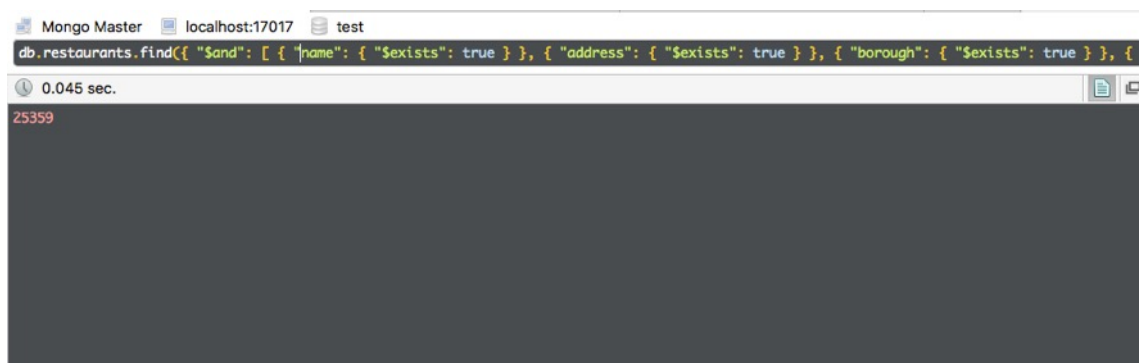
Se puede ver con el comando `$exists` si los distintos elementos del objeto descritos anteriormente existen en el resto de objetos de la colección, con este comando y haciendo un `count()` para ver que todos los objetos de la colección tienen esta estructura.

Hay 25359 objetos en la colección.



The screenshot shows the Mongo Master application interface. At the top, it displays 'Mongo Master', 'localhost:17017', and a database icon labeled 'test'. The command bar contains the query `db.restaurants.find().count()`. Below the command bar, a clock icon indicates a execution time of '0.001 sec.'. The result area shows the number '25359' in red text.

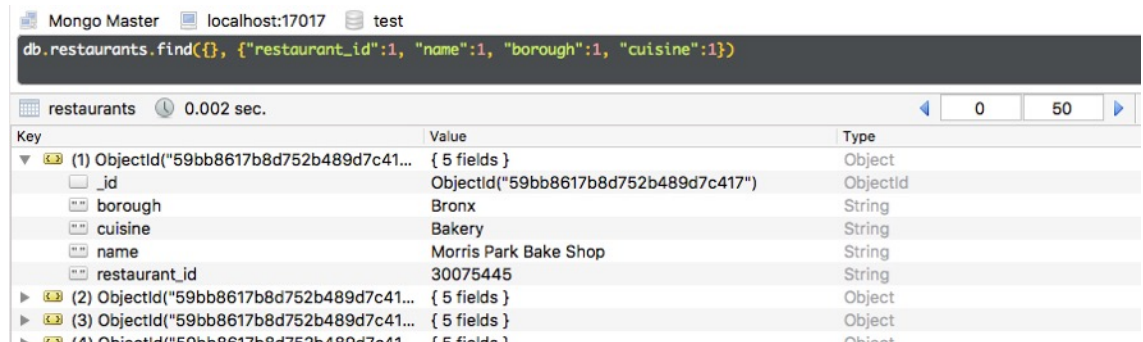
Viendo si existen los distintos elementos de la estructura, también nos devuelve 25359.



The screenshot shows the Mongo Master application interface with a more complex query. The command bar contains the query `db.restaurants.find({ "$and": [{ "name": { "$exists": true } }, { "address": { "$exists": true } }, { "borough": { "$exists": true } }, {`. Below the command bar, a clock icon indicates a execution time of '0.045 sec.'. The result area shows the number '25359' in red text.

2. (0,5 puntos) Listar todos los restaurantes de la colección, pero devolviendo únicamente los campos: "restaurant_id", "name", "borough" y "cuisine".

`db.restaurants.find({}, {"restaurant_id":1, "name":1, "borough":1, "cuisine":1})`

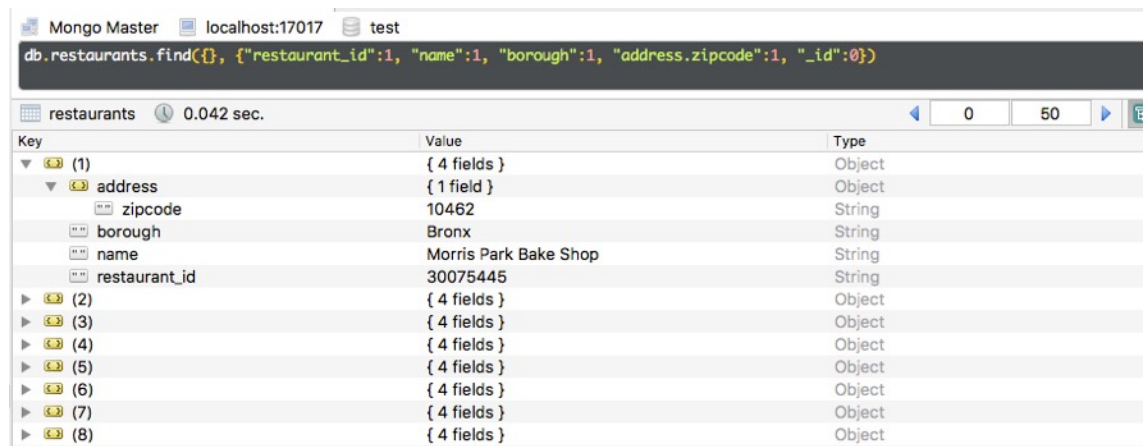


Key	Value	Type
(1) Objectid("59bb8617b8d752b489d7c41...")	{ 5 fields }	Object
_id	Objectid("59bb8617b8d752b489d7c417")	Objectid
borough	Bronx	String
cuisine	Bakery	String
name	Morris Park Bake Shop	String
restaurant_id	30075445	String
(2) Objectid("59bb8617b8d752b489d7c41...")	{ 5 fields }	Object
(3) Objectid("59bb8617b8d752b489d7c41...")	{ 5 fields }	Object
(4) Objectid("59bb8617b8d752b489d7c41...")	{ 5 fields }	Object

3. (0,5 puntos) Obtener todos los restaurantes de la colección, devolviendo los campos: "restaurant_id", "name", "borough" y "zip code", pero sin mostrar el campo "_id".

El campo _id se muestra por defecto, si no se quiere mostrar debe especificarse explícitamente con el valor 0.

`db.restaurants.find({}, {"restaurant_id":1, "name":1, "borough":1, "address.zipcode":1, "_id":0})`

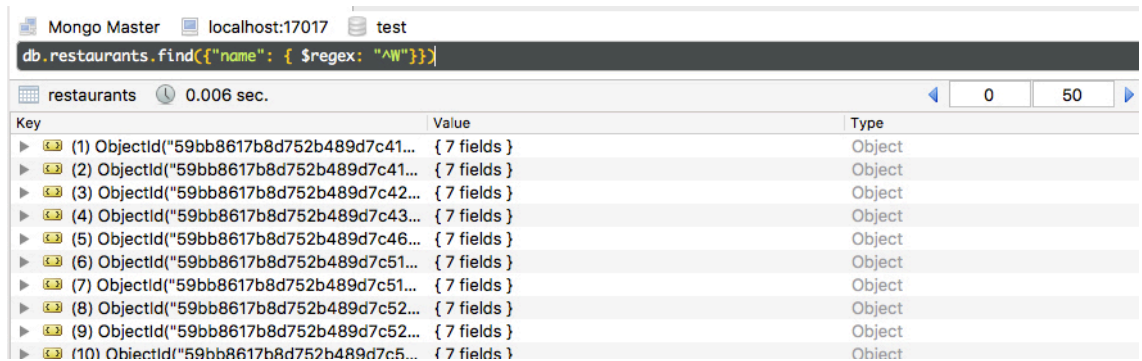


Key	Value	Type
(1)	{ 4 fields }	Object
address	{ 1 field }	Object
zipcode	10462	String
borough	Bronx	String
name	Morris Park Bake Shop	String
restaurant_id	30075445	String
(2)	{ 4 fields }	Object
(3)	{ 4 fields }	Object
(4)	{ 4 fields }	Object
(5)	{ 4 fields }	Object
(6)	{ 4 fields }	Object
(7)	{ 4 fields }	Object
(8)	{ 4 fields }	Object

4. (1,5 puntos) Mostrar todos los restaurantes que empiezan por la letra W. ¿Cuántos aparecen? ¿Existen nombres repetidos? Si existen, ¿qué consulta deberías realizar para no mostrar los nombres repetidos?.

Con la siguiente instrucción se obtienen los restaurantes que su nombre empieza con la letra W:

`db.restaurants.find({"name": { $regex: "^W"}})`



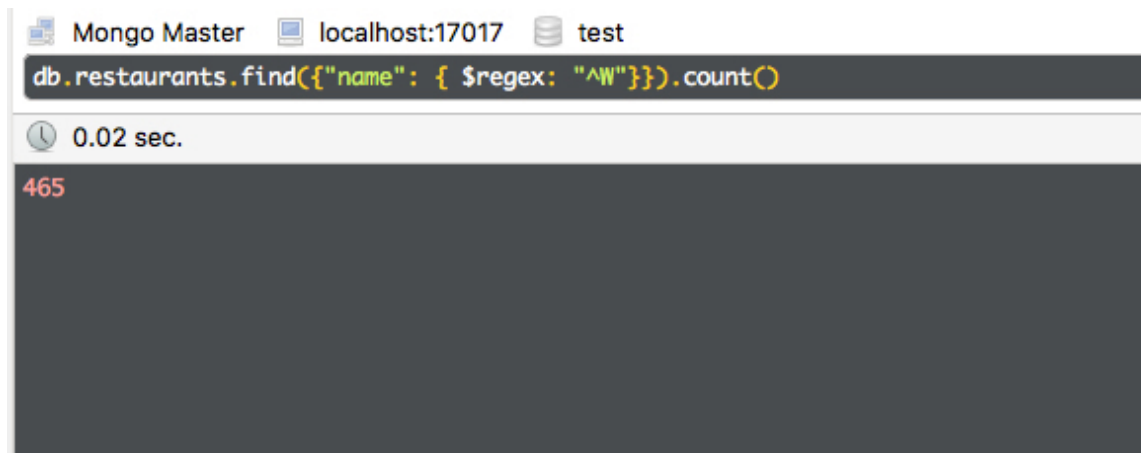
Mongo Master localhost:17017 test

```
db.restaurants.find({"name": { $regex: "^W"}})
```

restaurants 0.006 sec.

Key	Value	Type
▶ (1) ObjectId("59bb8617b8d752b489d7c41...")	{ 7 fields }	Object
▶ (2) ObjectId("59bb8617b8d752b489d7c41...")	{ 7 fields }	Object
▶ (3) ObjectId("59bb8617b8d752b489d7c42...")	{ 7 fields }	Object
▶ (4) ObjectId("59bb8617b8d752b489d7c43...")	{ 7 fields }	Object
▶ (5) ObjectId("59bb8617b8d752b489d7c46...")	{ 7 fields }	Object
▶ (6) ObjectId("59bb8617b8d752b489d7c51...")	{ 7 fields }	Object
▶ (7) ObjectId("59bb8617b8d752b489d7c51...")	{ 7 fields }	Object
▶ (8) ObjectId("59bb8617b8d752b489d7c52...")	{ 7 fields }	Object
▶ (9) ObjectId("59bb8617b8d752b489d7c52...")	{ 7 fields }	Object
▶ (10) ObjectId("59bb8617b8d752b489d7c5...")	{ 7 fields }	Object

Al contar se obtienen 465 objetos.



Mongo Master localhost:17017 test

```
db.restaurants.find({"name": { $regex: "^W"}}).count()
```

0.02 sec.

465

Con la siguiente instrucción se obtienen los 365 nombres distintos que empiezan por W:

```
db.restaurants.distinct("name", {"name": { $regex: "^W" }})
```

Mongo Master localhost:17017 test

```
db.restaurants.distinct("name", {"name": { $regex: "^W" }})
```

0.024 sec.

Key	Value	Type
(1)	[365 elements]	Array
[0]	Wendy'S	String
[1]	Wilken'S Fine Food	String
[2]	Wild Asia	String
[3]	White Castle	String
[4]	White Horse Tavern	String
[5]	Whitestone Lanes	String
[6]	West Bank Cafe	String
[7]	Water Front Crab House	String
[8]	War Memorial Ice Skating Rink	String
[9]	Water'S Edge Club	String
[10]	Western Bakery	String
[11]	Windjammers Bar	String
[12]	Winnie'S Bar	String
[13]	Westside Restaurant	String
[14]	Walnut Bus Stop	String
[15]	Westway Diner	String
[16]	Walker'S Restaurant	String
[17]	Wharf Bar And Grill	String
[18]	Walter'S Bar	String
[19]	Wembley Athletic Club	String
[20]	Winchesters Pub	String
[21]	Weiss Cafe Rockefeller University	String
[22]	World Cup Cafe	String

5. (0,5 puntos) Obtener los cinco primeros restaurantes cuyo campo “borough” es igual a Bronx.

`db.restaurants.find({"borough" : "Bronx"}).limit(5)`

Key	Value	Type
(1) ObjectId("59bb8617b8d752b489d7c417")	{ 7 fields }	Object
_id	ObjectId("59bb8617b8d752b489d7c417")	ObjectId
address	{ 4 fields }	Object
borough	Bronx	String
cuisine	Bakery	String
grades	[5 elements]	Array
name	Morris Park Bake Shop	String
restaurant_id	30075445	String
(2) ObjectId("59bb8617b8d752b489d7c421")	{ 7 fields }	Object
(3) ObjectId("59bb8617b8d752b489d7c43...")	{ 7 fields }	Object
(4) ObjectId("59bb8617b8d752b489d7c43...")	{ 7 fields }	Object
(5) ObjectId("59bb8617b8d752b489d7c44...")	{ 7 fields }	Object

6. (1 punto) Listar aquellos restaurantes que han alcanzado una puntuación mayor de 90 (campo “score”).

Con la siguiente instrucción se obtienen los restaurantes que tienen algún valor de score mayor a 90:

`db.restaurants.find({"grades.score": {"$gt":90}})`

Key	Value	Type
(1) ObjectId("59bb8617b8d752b489d7c575")	{ 7 fields }	Object
(2) ObjectId("59bb8617b8d752b489d7c61...")	{ 7 fields }	Object
(3) ObjectId("59bb8617b8d752b489d7c77...")	{ 7 fields }	Object
(4) ObjectId("59bb8617b8d752b489d7f7ad")	{ 7 fields }	Object

7. (1,5 puntos) ¿Qué consulta deberíamos lanzar para seleccionar todos los restaurantes que el tipo del campo “coord” es Double? ¿Todos los documentos de la colección tienen dicho campo a Double? Si la respuesta es no, ¿Cuántos no lo tienen? ¿Por qué?.

`db.restaurants.find({"address.coord": {"$type": "double"}})`

Mongo Master localhost:17017 test

```
db.restaurants.find({"address.coord": {"$type": "double"}})
```

restaurants 0.003 sec. 0 50

Key	Value	Type
(1) ObjectId("59bb8617b8d752b489d7c41...")	{ 7 fields }	Object
(2) ObjectId("59bb8617b8d752b489d7c41...")	{ 7 fields }	Object
(3) ObjectId("59bb8617b8d752b489d7c41...")	{ 7 fields }	Object
(4) ObjectId("59bb8617b8d752b489d7c41...")	{ 7 fields }	Object
(5) ObjectId("59bb8617b8d752b489d7c41...")	{ 7 fields }	Object
(6) ObjectId("59bb8617b8d752b489d7c41...")	{ 7 fields }	Object
(7) ObjectId("59bb8617b8d752b489d7c41...")	{ 7 fields }	Object
(8) ObjectId("59bb8617b8d752b489d7c41...")	{ 7 fields }	Object
(9) ObjectId("59bb8617b8d752b489d7c41...")	{ 7 fields }	Object
(10) ObjectId("59bb8617b8d752b489d7c41...")	{ 7 fields }	Object
(11) ObjectId("59bb8617b8d752b489d7c41...")	{ 7 fields }	Object
(12) ObjectId("59bb8617b8d752b489d7c41...")	{ 7 fields }	Object
(13) ObjectId("59bb8617b8d752b489d7c41...")	{ 7 fields }	Object
(14) ObjectId("59bb8617b8d752b489d7c41...")	{ 7 fields }	Object
(15) ObjectId("59bb8617b8d752b489d813...")	{ 7 fields }	Object
(16) ObjectId("59bb8617b8d752b489d7c41...")	{ 7 fields }	Object
(17) ObjectId("59bb8617b8d752b489d7c41...")	{ 7 fields }	Object

Al contar los elementos obtenidos, vemos que es menor al total de objetos de la colección.

Ejecutamos la siguiente instrucción, y vemos como dos objetos de la colección no cumplen con la condición de que las coordenadas sean de tipo double. Como se puede apreciar en la siguiente imagen es porque el array coord está vacío.

`db.restaurants.find({"address.coord": {"$not": {"$type": "double"}}})`

Mongo Master localhost:17017 test

```
db.restaurants.find({"address.coord": {"$not": {"$type": "double"}}})
```

restaurants 0.016 sec. 0 50

Key	Value	Type
(1) ObjectId("59bb8617b8d752b489d7d49...")	{ 7 fields }	Object
_id	ObjectId("59bb8617b8d752b489d7d49b")	ObjectId
address	{ 4 fields }	Object
building	0	String
coord	[0 elements]	Array
street	Wards Island/2FI	String
zipcode	10057	String
borough	Manhattan	String
cuisine	American	String
grades	[5 elements]	Array
name	Fratelli'S Market Place	String
restaurant_id	40959339	String
(2) ObjectId("59bb8618b8d752b489d8244...")	{ 7 fields }	Object
_id	ObjectId("59bb8618b8d752b489d8244d")	ObjectId
address	{ 4 fields }	Object
building	397	String
coord	[0 elements]	Array
street	Tompkins Ave	String
zipcode	11216	String
borough	Brooklyn	String
cuisine	American	String
grades	[1 element]	Array
name	Eugene & Co	String
restaurant_id	50017256	String

8. (1,5 puntos) Obtener todos los restaurantes que tengan siete o más puntuaciones (campo "grades") y que además, el valor del campo "cuisine" tenga el valor *Hamburgers*. Listar únicamente de cada uno de estos restaurantes el nombre y dirección, que no salga el id y que lo ordene alfabéticamente de forma descendente por el campo "name".

Se obtienen los restaurantes donde el campo grades sea igual o mayor a 7 elementos:

`db.restaurants.find({ "$where": "this.grades.length >= 7" })`

Mongo Master localhost:17017 test

```
db.restaurants.find({ "$where": "this.grades.length >= 7" })
```

restaurants 0.072 sec. 0 50

Key	Value	Type
(1) ObjectId("59bb8617b8d752b489d7c43...")	{ 7 fields }	Object
_id	ObjectId("59bb8617b8d752b489d7c432")	ObjectId
address	{ 4 fields }	Object
borough	Queens	String
cuisine	Chinese	String
grades	[8 elements]	Array
name	Ho Mei Restaurant	String
restaurant_id	40362432	String
(2) ObjectId("59bb8617b8d752b489d7c48...")	{ 7 fields }	Object
_id	ObjectId("59bb8617b8d752b489d7c480")	ObjectId
address	{ 4 fields }	Object
borough	Manhattan	String
cuisine	American	String
grades	[7 elements]	Array
name	Reynold'S Bar	String
restaurant_id	40365423	String
(3) ObjectId("59bb8617b8d752b489d7c48...")	{ 7 fields }	Object
(4) ObjectId("59bb8617b8d752b489d7c4...")	{ 7 fields }	Object
(5) ObjectId("59bb8617b8d752b489d7c49...")	{ 7 fields }	Object

Se han obtenido 916 restaurantes con esta condición.

Ahora se obtienen los restaurantes de tipo de cocina *Hamburgers*:

`db.restaurants.find({ "cuisine" : "Hamburgers" })`

Mongo Master localhost:17017 test

```
db.restaurants.find({ "cuisine" : "Hamburgers" })
```

restaurants 0.005 sec. 0 50

Key	Value	Type
(1) ObjectId("59bb8617b8d752b489d7c41...")	{ 7 fields }	Object
_id	ObjectId("59bb8617b8d752b489d7c418")	ObjectId
address	{ 4 fields }	Object
borough	Brooklyn	String
cuisine	Hamburgers	String
grades	[4 elements]	Array
name	Wendy'S	String
restaurant_id	30112340	String
(2) ObjectId("59bb8617b8d752b489d7c43...")	{ 7 fields }	Object
_id	ObjectId("59bb8617b8d752b489d7c431")	ObjectId
address	{ 4 fields }	Object
borough	Brooklyn	String
cuisine	Hamburgers	String
grades	[4 elements]	Array
name	White Castle	String
restaurant_id	40362344	String
(3) ObjectId("59bb8617b8d752b489d7c4e...")	{ 7 fields }	Object
(4) ObjectId("59bb8617b8d752b489d7c4e...")	{ 7 fields }	Object
(5) ObjectId("59bb8617b8d752b489d7c52...")	{ 7 fields }	Object

Se obtienen 433 restaurantes con este criterio.

Para obtener los restaurantes que cumplen con todo el enunciado, se unen las dos consultas anteriores con un “and”, y además se restringen los campos a mostrar en la consulta y se ordenan por el nombre:

```
db.restaurants.find( { "$and": [ { "$where": "this.grades.length >= 7" }, { "cuisine" : "Hamburgers" } ] }, { "name":1, "address":1, "_id":0 }).sort({"name":-1})
```



Mongo Master localhost:17017 test

db.restaurants.find({ "\$and": [{ "\$where": "this.grades.length >= 7" }, { "cuisine" : "Hamburgers" }] }, { "name":1, "address":1, "_id":0 }).sort({"name":-1})

restaurants 0.067 sec. 0 50

Key	Value	Type
(1)	{ 2 fields }	Object
address	{ 4 fields }	Object
building	7002	String
coord	[2 elements]	Array
street	Copper Avenue	String
zipcode	11385	String
name	McDonald'S	String
(2)	{ 2 fields }	Object
(3)	{ 2 fields }	Object
(4)	{ 2 fields }	Object
(5)	{ 2 fields }	Object
(6)	{ 2 fields }	Object
address	{ 4 fields }	Object
name	Lucky'S Famous Burgers	String
(7)	{ 2 fields }	Object
(8)	{ 2 fields }	Object
(9)	{ 2 fields }	Object
address	{ 4 fields }	Object
name	Burger King, Popeye'S Chicken & Biscuits	String
(10)	{ 2 fields }	Object
address	{ 4 fields }	Object
name	Burger King	String

Al final se obtienen 10 restaurantes de cocina de tipo hamburguesería y además donde el campo grades tenga al menos 7 elementos. Se muestran en la salida de la consulta sólo los campos address y name, y se ordenan por el name de manera descendente.