mongoDB®

MongoDB 1

Fundamentals

Contents

- Introduction to MongoDB
- Accessing MongoDB Atlas
- Storage and Retrieval

Introduction to MongoDB

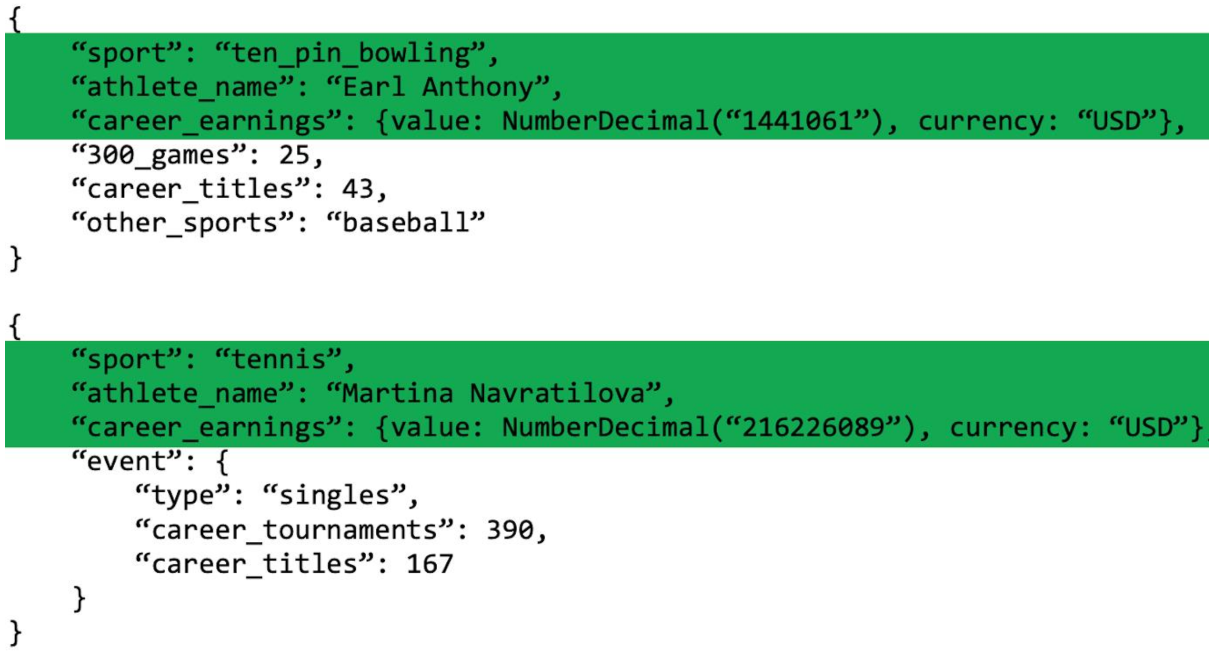
- Modern Document-model database.
- Designed to back the modern-day business applications:
  - Developer and Operations oriented
  - Easy to scale horizontally
  - Business Critical
  - Lessons learned from 50 years of RDBMS
- What two RDBMS functions are hard to do efficiently in a distributed system?

Terminology

SQL Terms/Concepts	MongoDB Terms/Concepts
database	database
table	collection
row	document or BSON document
column	field
index	index

https://www.mongodb.com/docs/manual/reference/sql-comparison/

MongoDB – Polymorphic



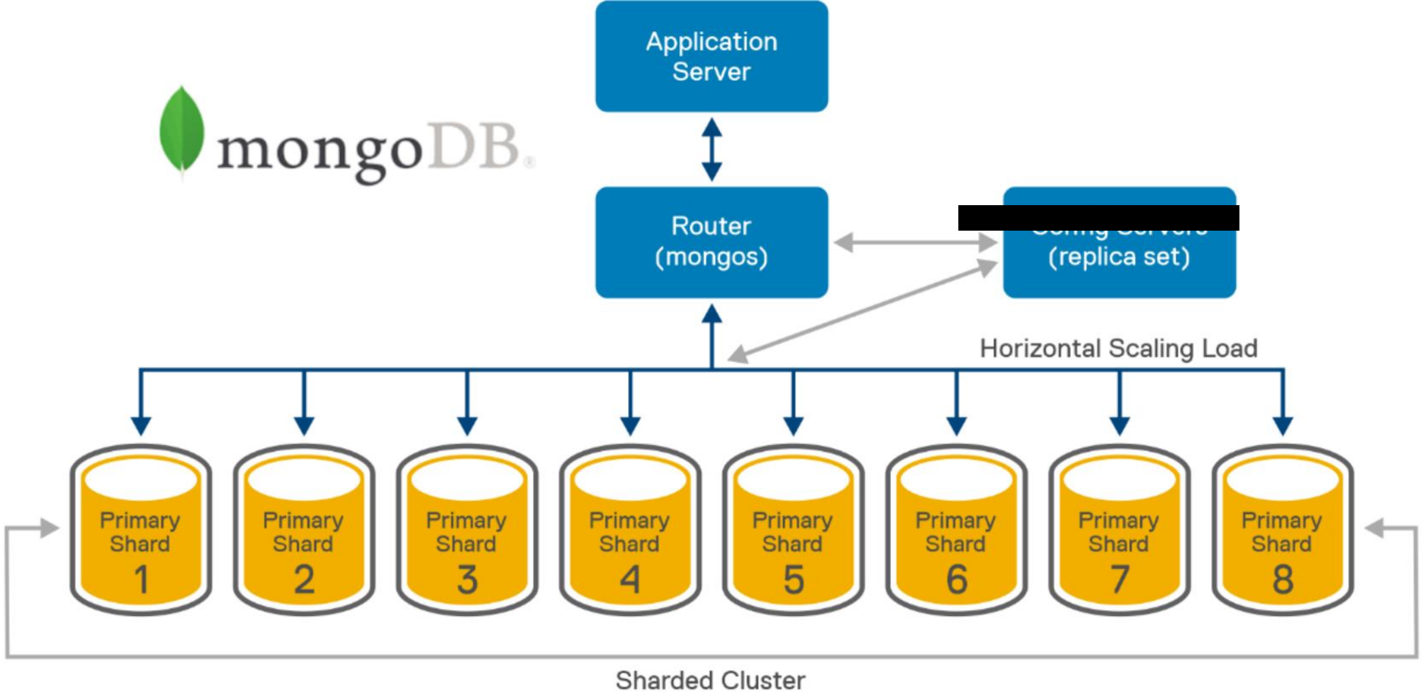
```
{
    “sport”: “ten_pin_bowling”,
    “athlete_name”: “Earl Anthony”,
    “career_earnings”: {value: NumberDecimal(“1441061”), currency: “USD”},
    “300_games”: 25,
    “career_titles”: 43,
    “other_sports”: “baseball”
}

{
    “sport”: “tennis”,
    “athlete_name”: “Martina Navratilova”,
    “career_earnings”: {value: NumberDecimal(“216226089”), currency: “USD”},
    “event”: {
        “type”: “singles”,
        “career_tournaments”: 390,
        “career_titles”: 167
    }
}
```

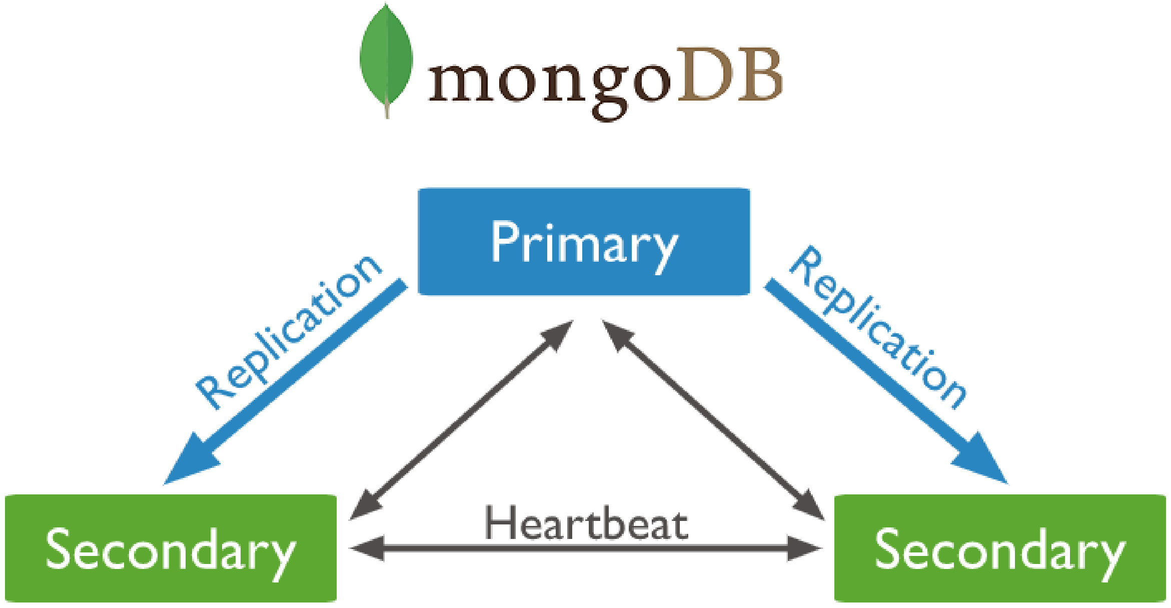
MongoDB – Availability and Scalability

– High Availability (Replica Sets)

– High Scalability (Sharding)



+

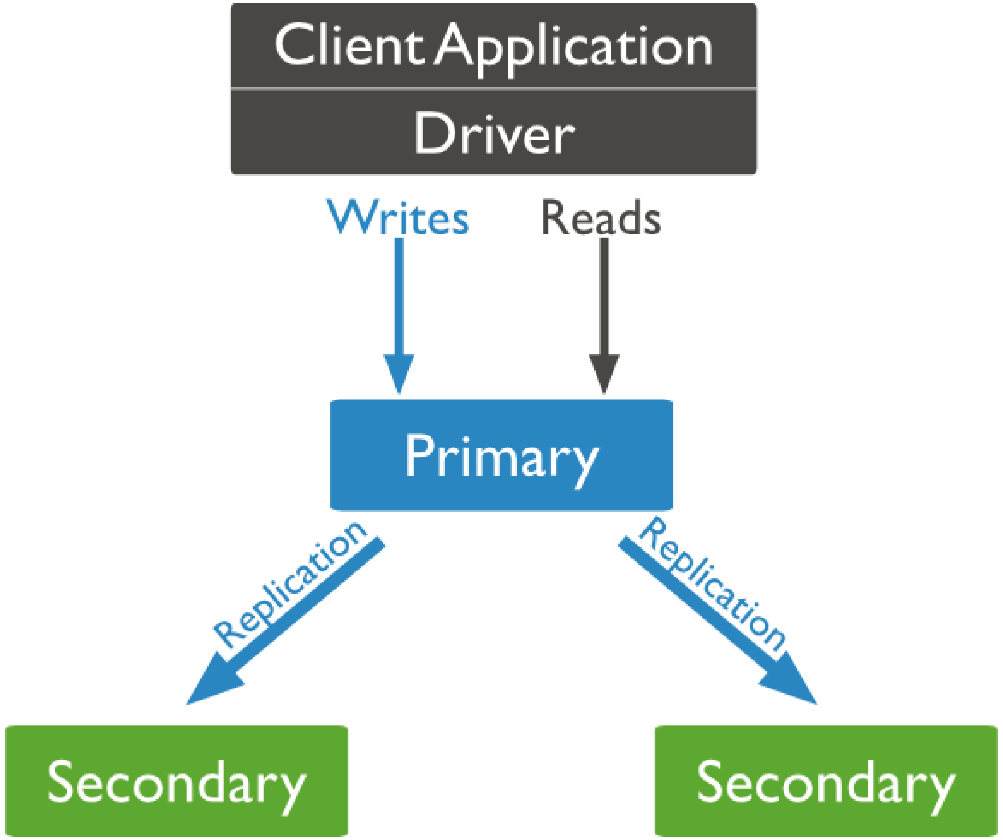


Benefits of Replication

– Increases data availability

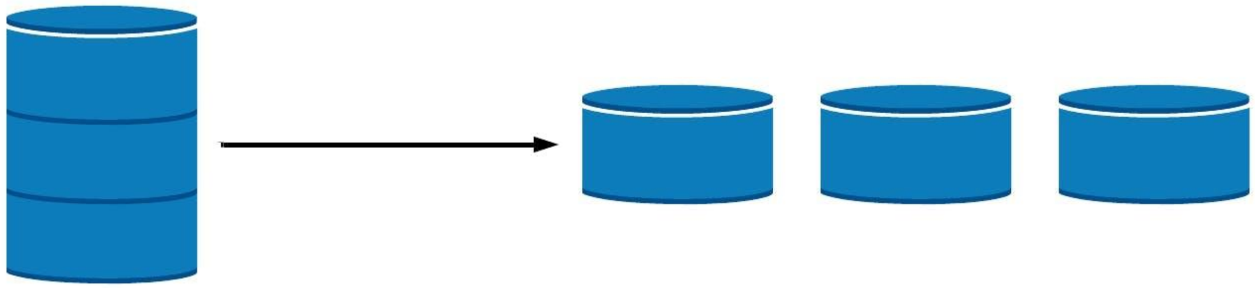
– Increases data reliability

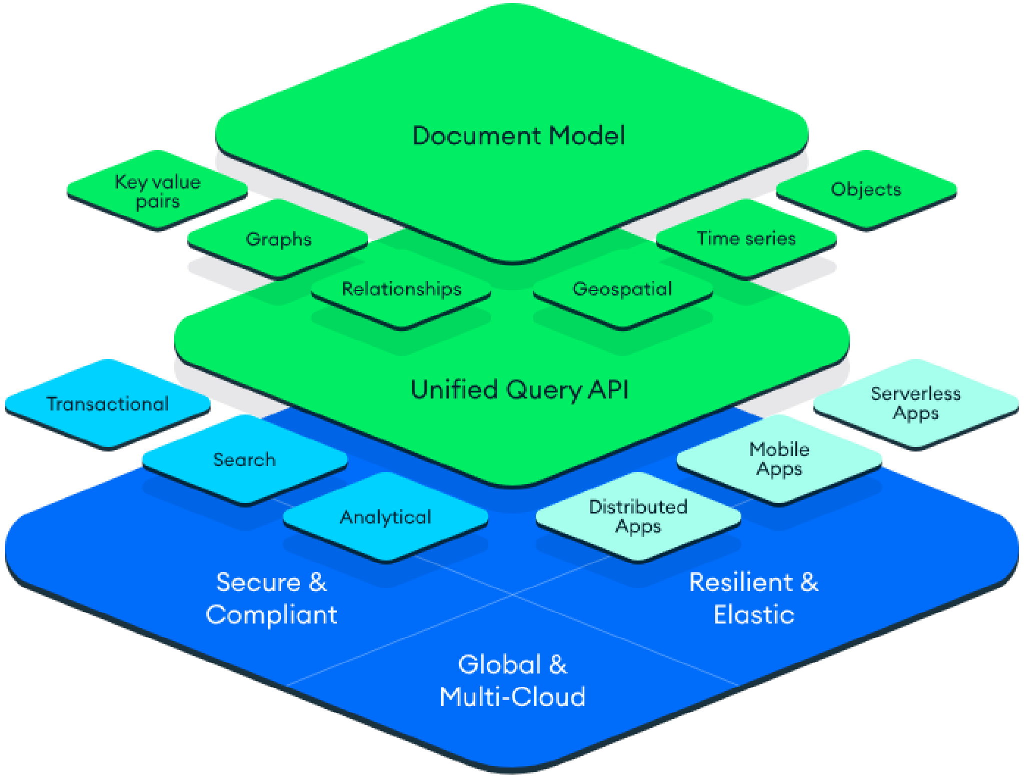
– Helpful in case of an event like hardware failure or a server crash



Benefits of Sharding

- Increased read/write throughput
- Increased storage capacity
- High availability





Accessing MongoDB Atlas

MongoDB Atlas is MongoDB as a Service.

Setup a Cluster

– Follow the guide

Download Mongo shell

https://www.mongodb.com/docs/mongodb-shell/

Essential commands

1. Show all databases (**show dbs**)

2. Check the current database (**db**)

3. Change current database (**use <DBNAME>**)

4. Show all collections in the database (**show collections**)

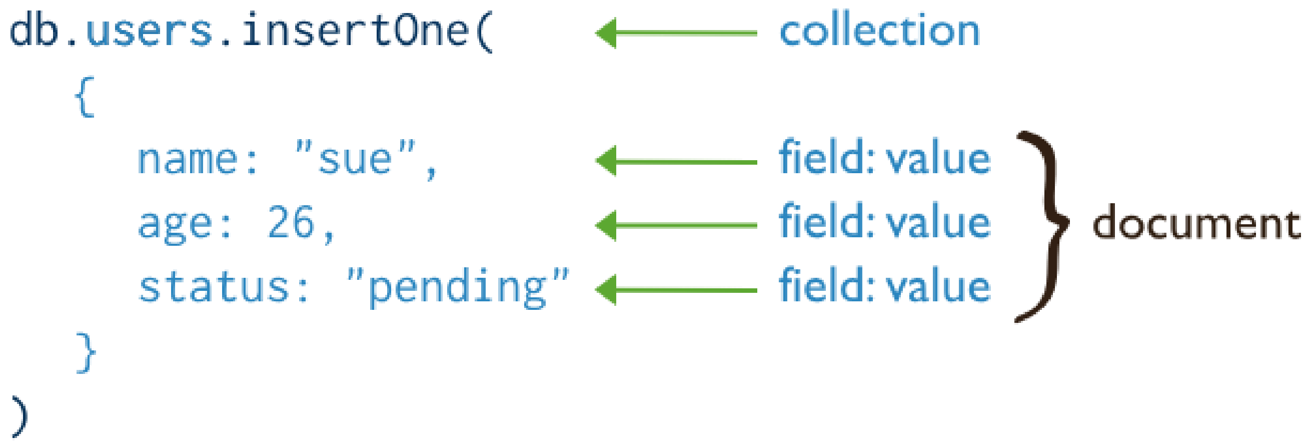
Storage and Retrieval

Basic Database CRUD Interactions

- **Create**
- **Read**
- **Update**
- **Delete**

Creating New Documents - insertOne

- Add one document to the collection
- **_id** field must be unique
- **_id** is Auto-generated if not supplied

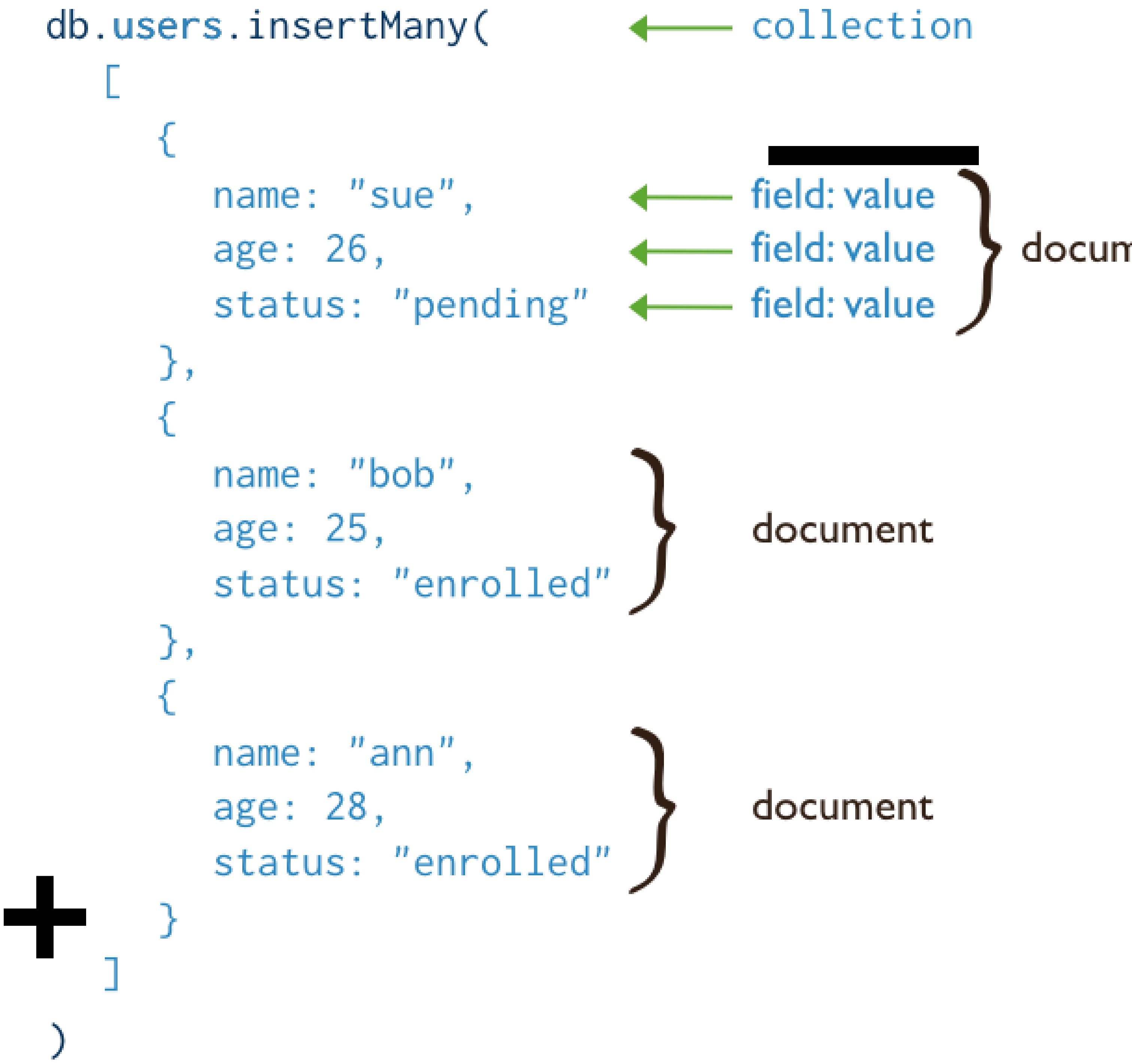


```
db.users.insertOne(
   {
      name: "sue",
      age: 26,
      status: "pending"
   }
)
```

Creating New Documents - insertMany

– Add array of documents to the collection

– Faster than multiple insertOne calls



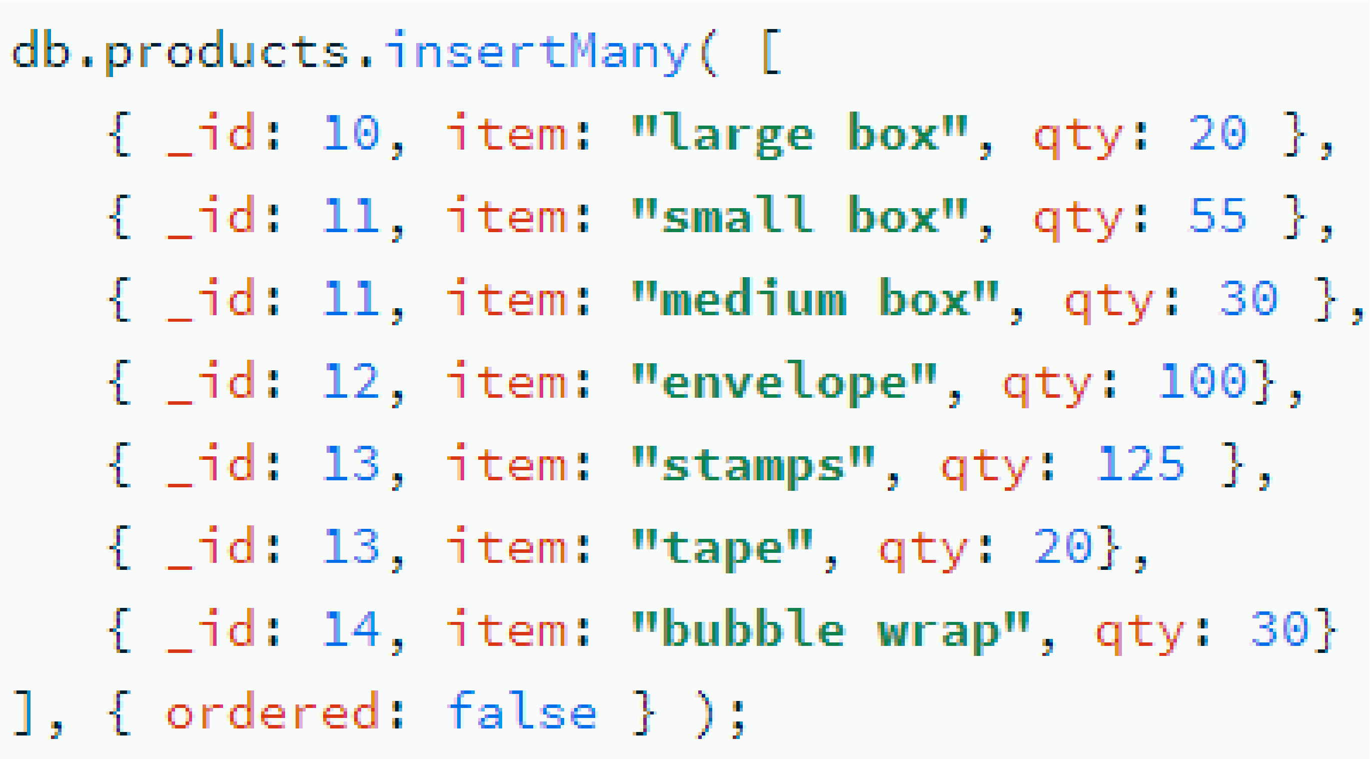
```
db.users.insertMany(
   [
     {
       name: "sue",
       age: 26,
       status: "pending"
     },
     {
       name: "bob",
       age: 25,
       status: "enrolled"
     },
     {
       name: "ann",
       age: 28,
       status: "enrolled"
     }
   ]
)
```

[redacted]

Creating New Documents - insertMany

– Ordered (**default**)

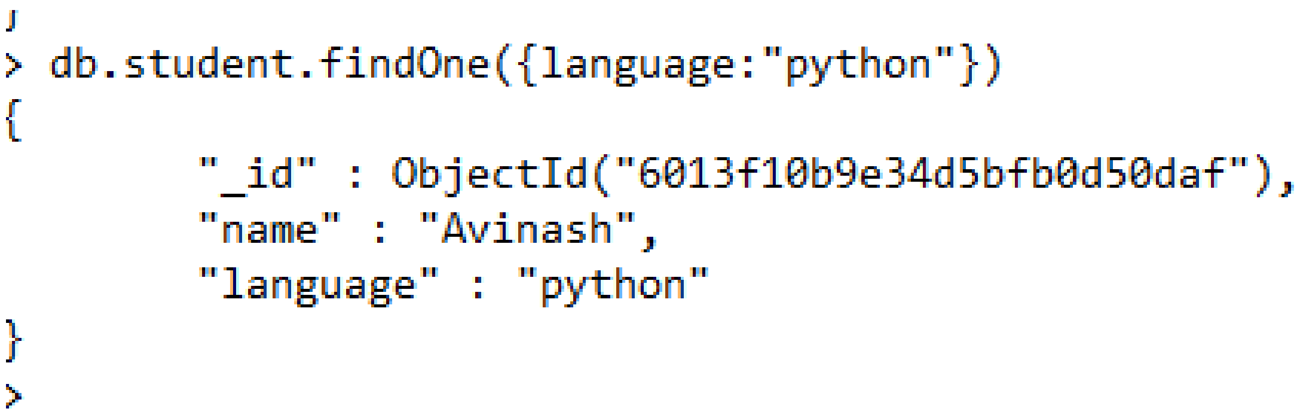
– Unordered



```
db.products.insertMany( [
   { _id: 10, item: "large box", qty: 20 },
   { _id: 11, item: "small box", qty: 55 },
   { _id: 11, item: "medium box", qty: 30 },
   { _id: 12, item: "envelope", qty: 100},
   { _id: 13, item: "stamps", qty: 125 },
   { _id: 13, item: "tape", qty: 20},
   { _id: 14, item: "bubble wrap", qty: 30}
], { ordered: false } );
```

Reading Documents - findOne

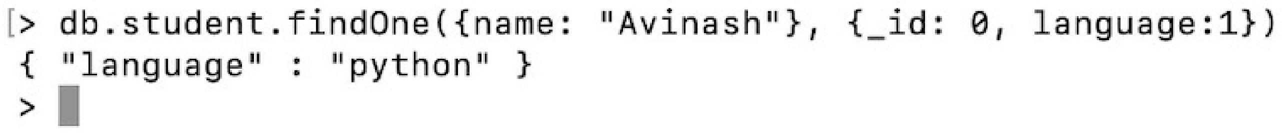
- Retrieves a single document
- Query by example



```
> db.student.findOne({language:"python"})
{
        "_id" : ObjectId("6013f10b9e34d5bfb0d50daf"),
        "name" : "Avinash",
        "language" : "python"
}
>
```

Reading Documents - projection

- Include/Exclude
- Query by example



```
[> db.student.findOne({name: "Avinash"}, {_id: 0, language:1})
 { "language" : "python" }
 >
```

Reading Documents - find

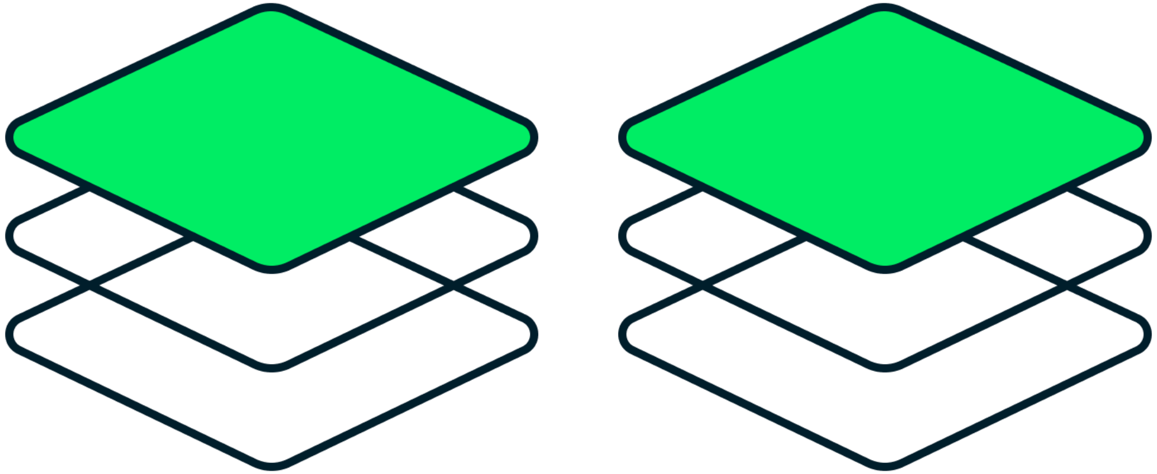
– Returns cursor

```
db.collection.find()
```

```
{ "_id": "apples", "qty": 5 }
{ "_id": "bananas", "qty": 7 }
```

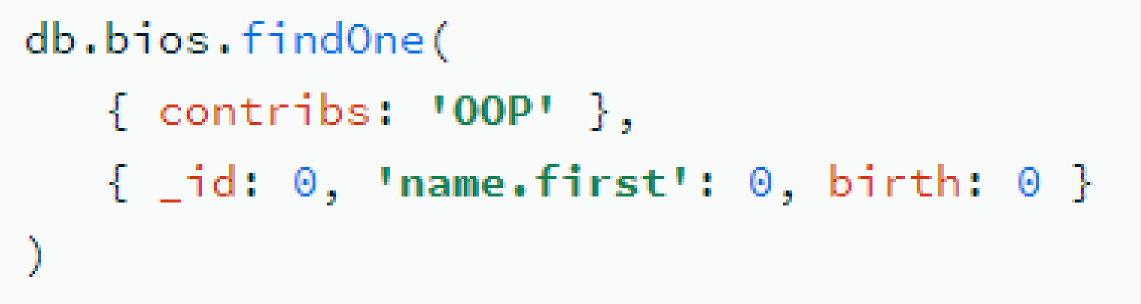
Cursors

- Store results of find to a variable
- Append skip, limit, sort
- Work in batches



Reading Documents – find in nested documents

– Dot notation “address.city”



```
db.bios.findOne(
   { contribs: 'OOP' },
   { _id: 0, 'name.first': 0, birth: 0 }
)
```

Reading documents – Range of values

$lt

$lte

$gt

$gte

$in

$nin

Reading documents – Boolean Logic Operators

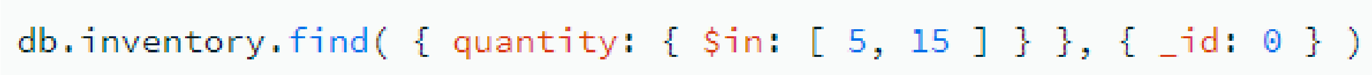
$and

$or

$not

$nor

Reading documents – Querying arrays

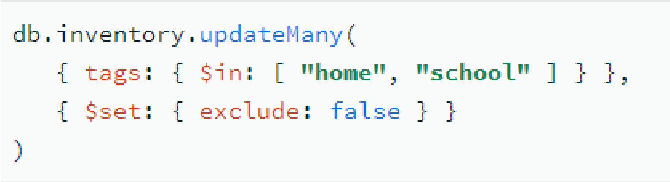


```
db.inventory.find( { quantity: { $in: [ 5, 15 ] } }, { _id: 0 } )
```

- $all
- $size
- $elemMatch

Updating Documents

- updateOne - Update first matching document
- updateMany – Update all matching Documents



```
db.inventory.updateMany(
   { tags: { $in: [ "home", "school" ] } },
   { $set: { exclude: false } }
)
```

Update Document - Mutation

Mutation is an object describing the changes to make to each record.

Values can be explicitly set or changed relative to the current value or external values

- $set
- $unset
- $inc
- $mul
- $max
- $push
- $pop
- $pull
- $pullAll
- $addToSet

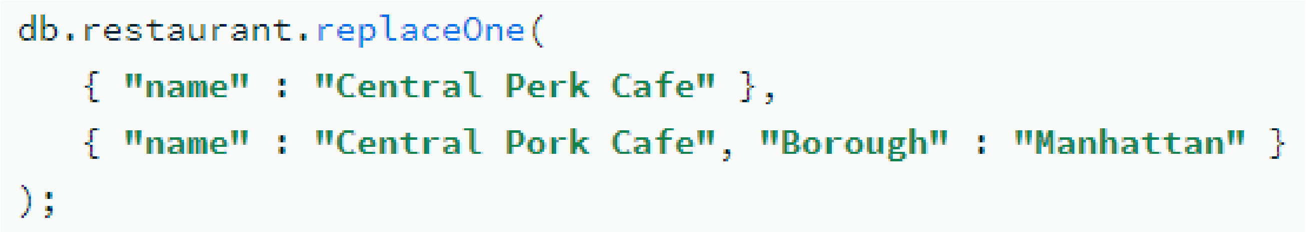
Delete document

deleteOne

deleteMany

Overwriting a document

– replaceOne()



```
db.restaurant.replaceOne(
   { "name" : "Central Perk Cafe" },
   { "name" : "Central Pork Cafe", "Borough" : "Manhattan" }
);
```

Questions