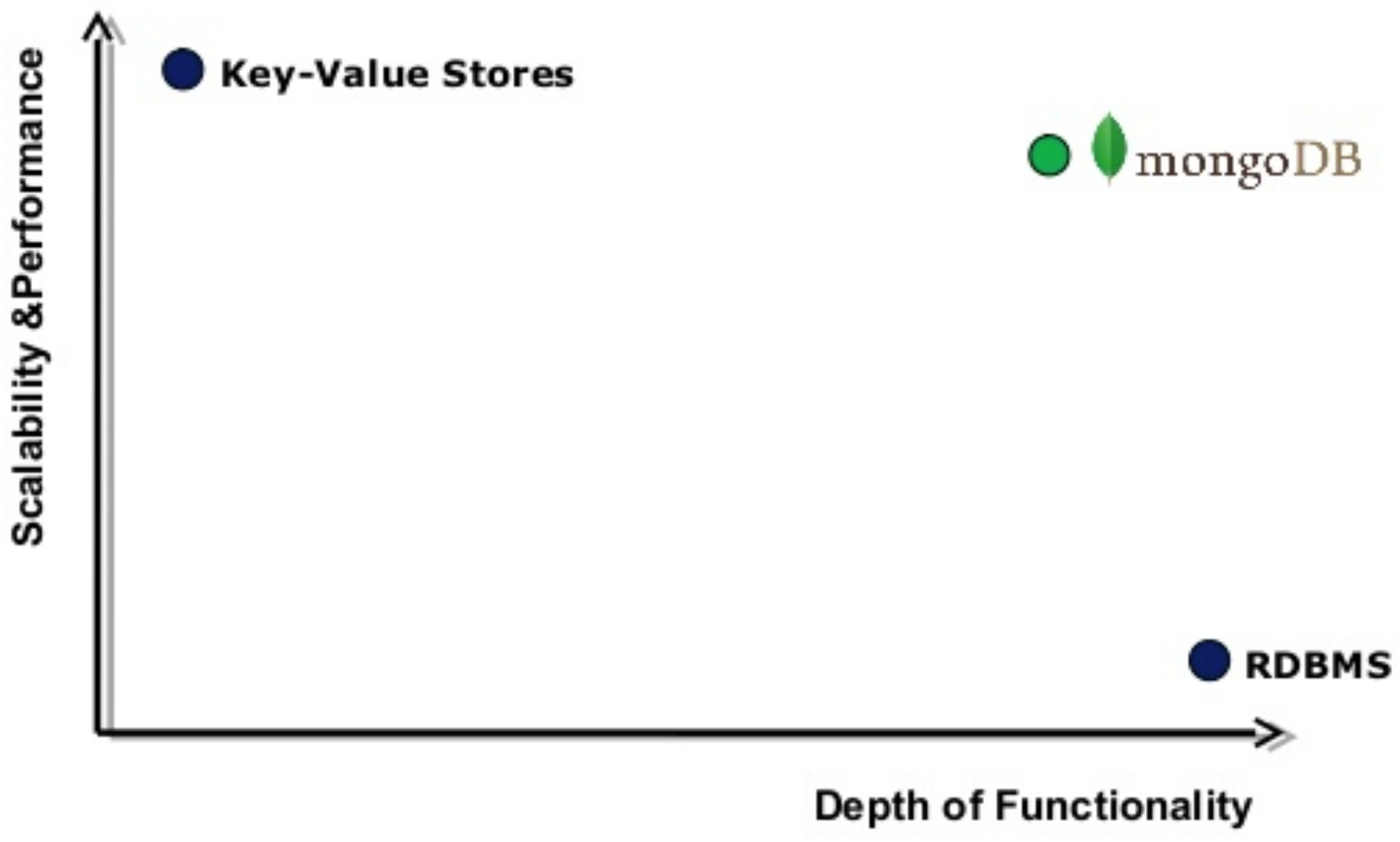
*MongoDB (from "hu**mongo**us") is a scalable, high-performance, open source, schema-free, document-oriented database.*

-- Mongodb.org

Background

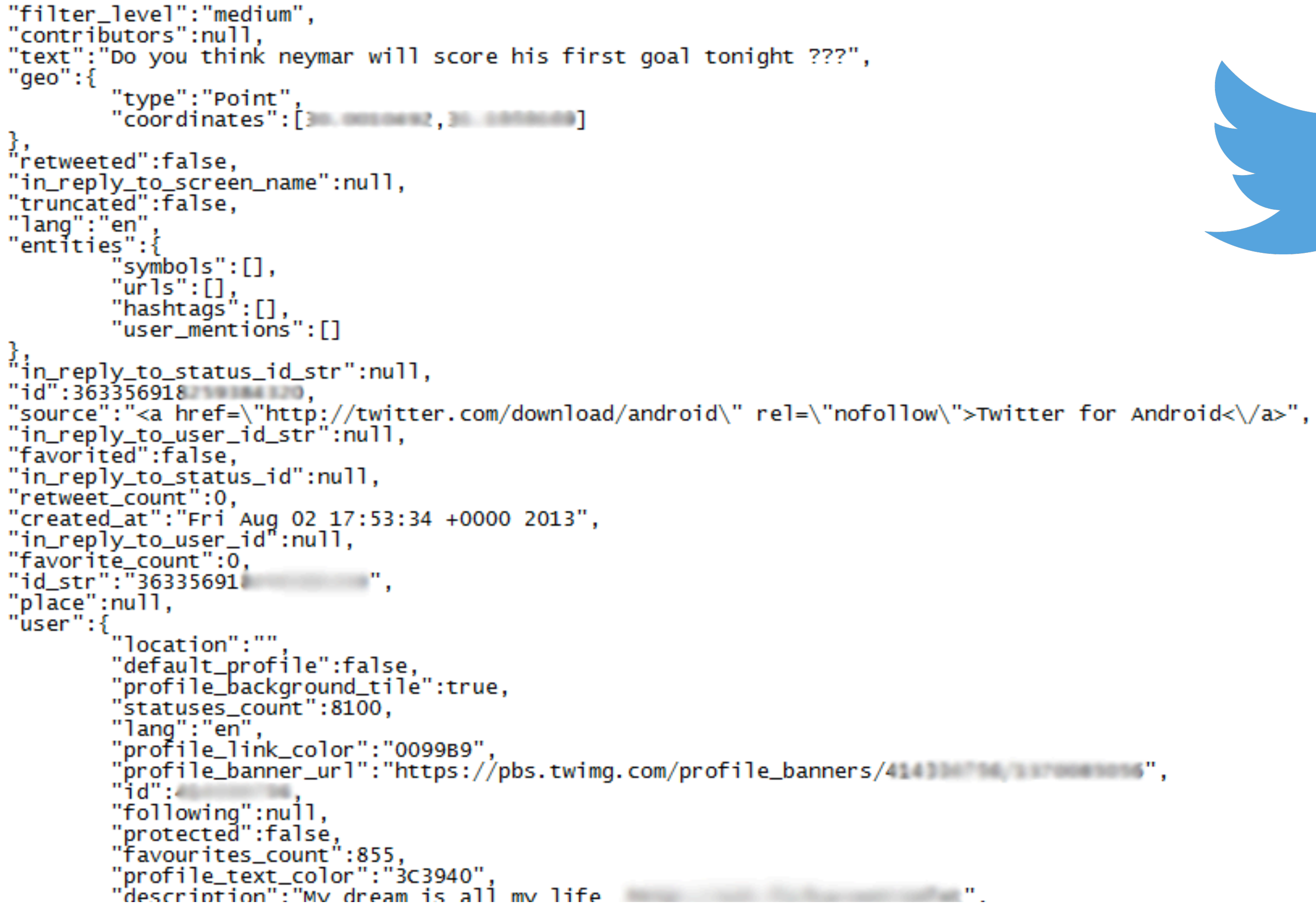
- A NoSQL database of type document oriented
- Eschews the traditional table-based relational database structure in favor of JSON-like documents with dynamic schemas (MongoDB calls the format BSON)
- First developed by MongoDB Inc in October 2007
- Shifted to open source in 2009

Where MongoDB Stands?



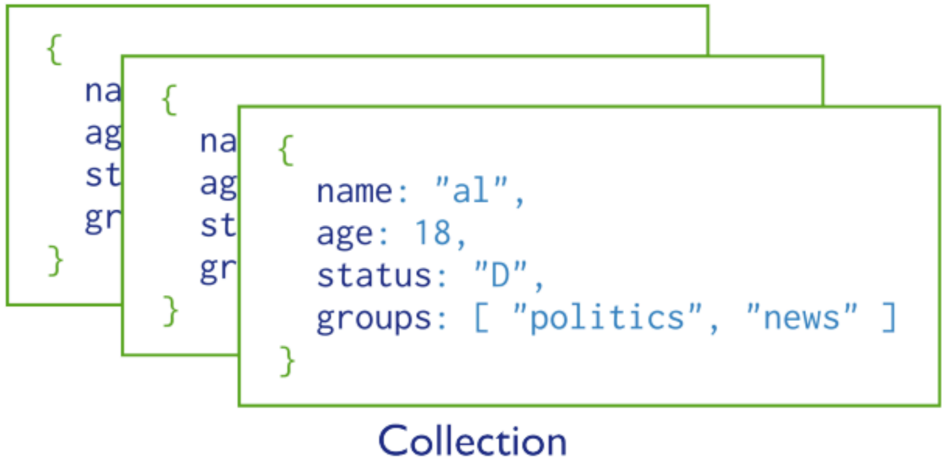
http://www.slideshare.net/cuelogic/introduction-to-mongodb-33683242

Examples of JSON Format



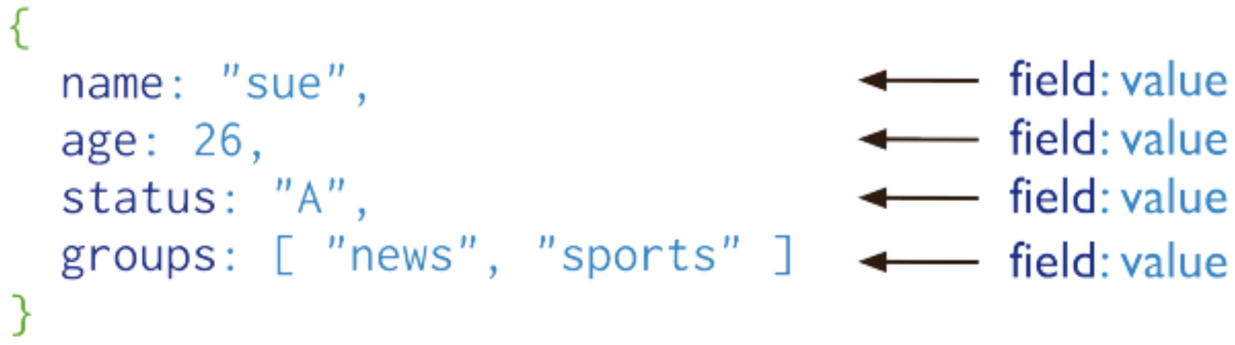
```
"filter_level":"medium",
"contributors":null,
"text":"Do you think neymar will score his first goal tonight ???",
"geo":{
        "type":"Point",
        "coordinates":[[redacted],[redacted]]
},
"retweeted":false,
"in_reply_to_screen_name":null,
"truncated":false,
"lang":"en",
"entities":{
        "symbols":[],
        "urls":[],
        "hashtags":[],
        "user_mentions":[]
},
"in_reply_to_status_id_str":null,
"id":36335691[redacted],
"source":"<a href=\"http:\/\/twitter.com\/download\/android\" rel=\"nofollow\">Twitter for Android<\/a>",
"in_reply_to_user_id_str":null,
"favorited":false,
"in_reply_to_status_id":null,
"retweet_count":0,
"created_at":"Fri Aug 02 17:53:34 +0000 2013",
"in_reply_to_user_id":null,
"favorite_count":0,
"id_str":"36335691[redacted]",
"place":null,
"user":{
        "location":"",
        "default_profile":false,
        "profile_background_tile":true,
        "statuses_count":8100,
        "lang":"en",
        "profile_link_color":"0099B9",
        "profile_banner_url":"https://pbs.twimg.com/profile_banners/[redacted]",
        "id":[redacted],
        "following":null,
        "protected":false,
        "favourites_count":855,
        "profile_text_color":"3C3940",
        "description":"My dream is all my life [redacted]",
```

MongoDB Basics (cont.)



```
{
   name: "al",
   age: 18,
   status: "D",
   groups: [ "politics", "news" ]
}
```

Each document within a collection can have its own unique set of fields



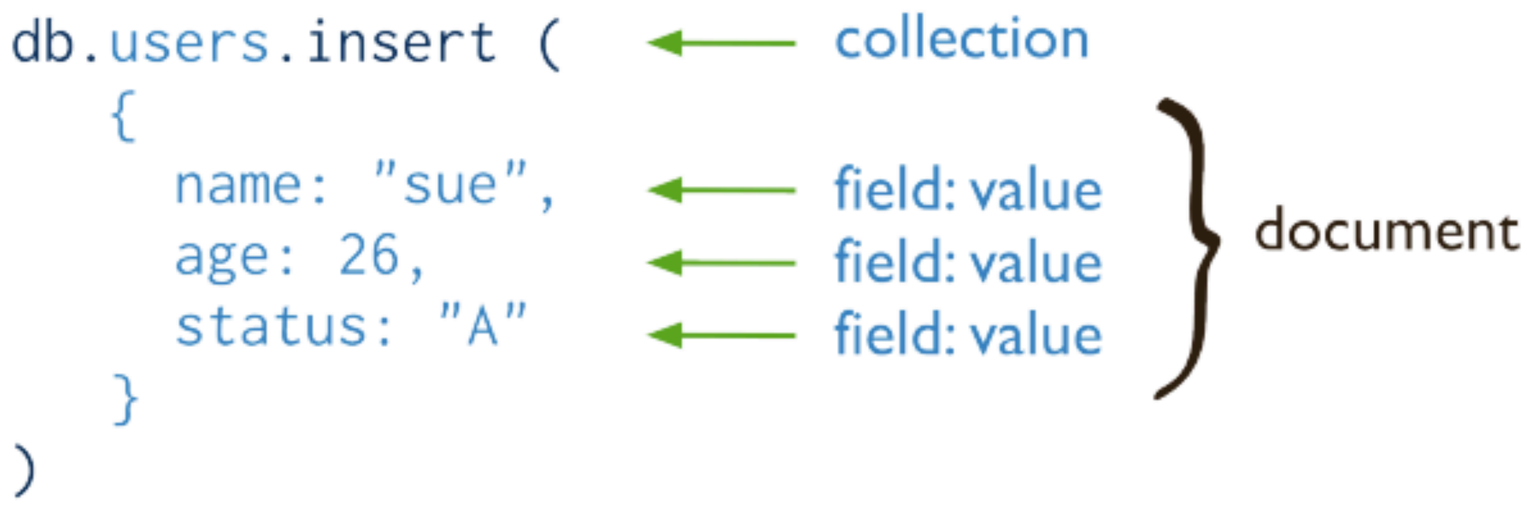
```
{
   name: "sue",
   age: 26,
   status: "A",
   groups: [ "news", "sports" ]
}
```

MongoDB CRUD

- MongoDB provides rich semantics for reading and manipulating data
- CRUD = Create, Read, Update, and Delete

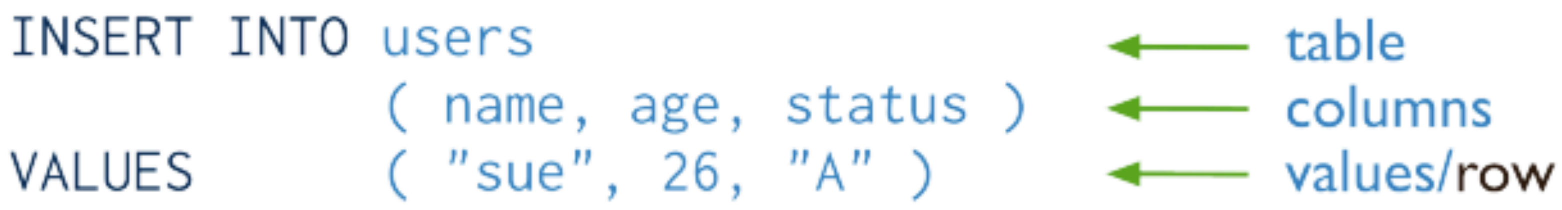
CRUD: Create

In MongoDB



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

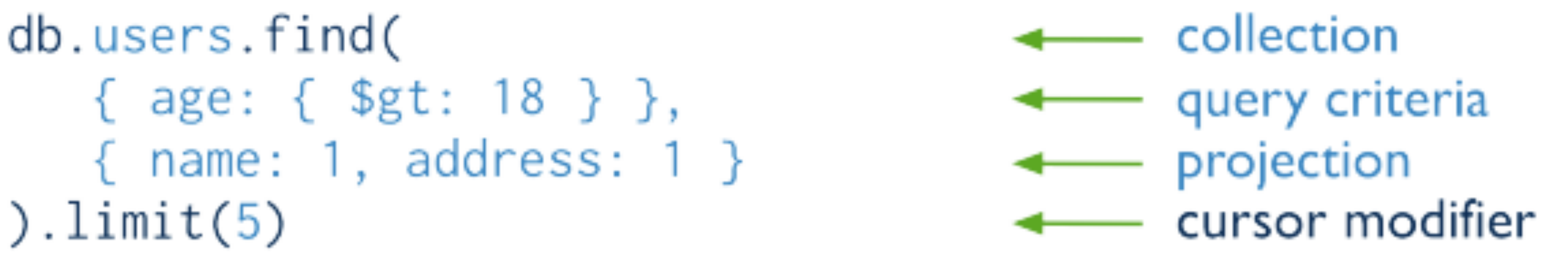
In SQL



```
INSERT INTO users
            ( name, age, status )
VALUES      ( "sue", 26, "A" )
```

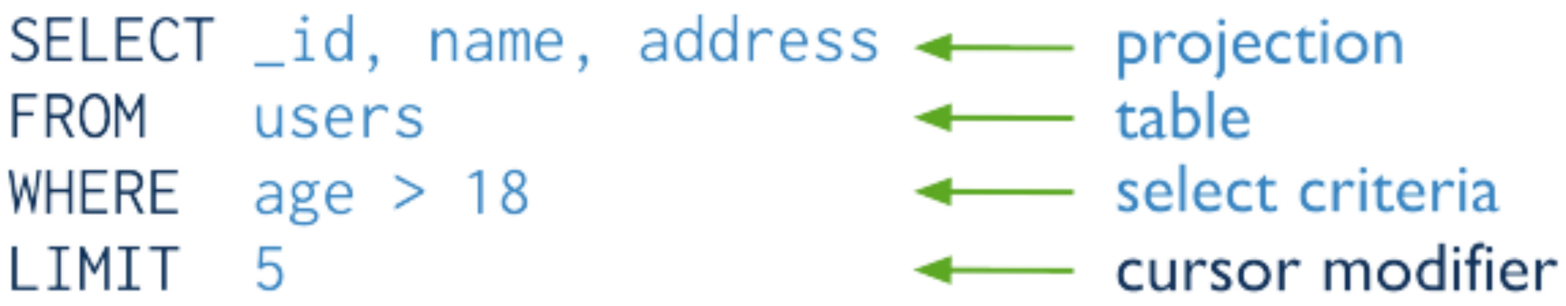
CRUD: Read

In MongoDB



```
db.users.find(
   { age: { $gt: 18 } },
   { name: 1, address: 1 }
).limit(5)
```

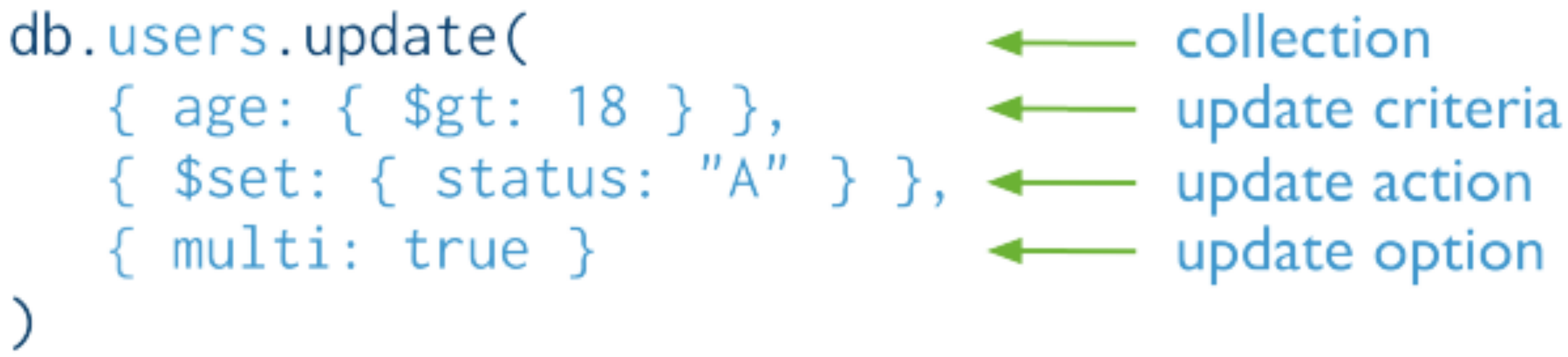
In SQL



```
SELECT _id, name, address
FROM   users
WHERE  age > 18
LIMIT  5
```

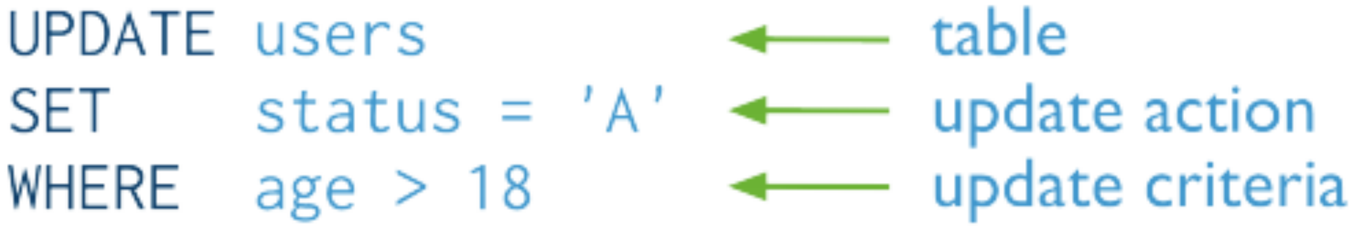
CRUD: Update

In MongoDB



```
db.users.update(
   { age: { $gt: 18 } },
   { $set: { status: "A" } },
   { multi: true }
)
```

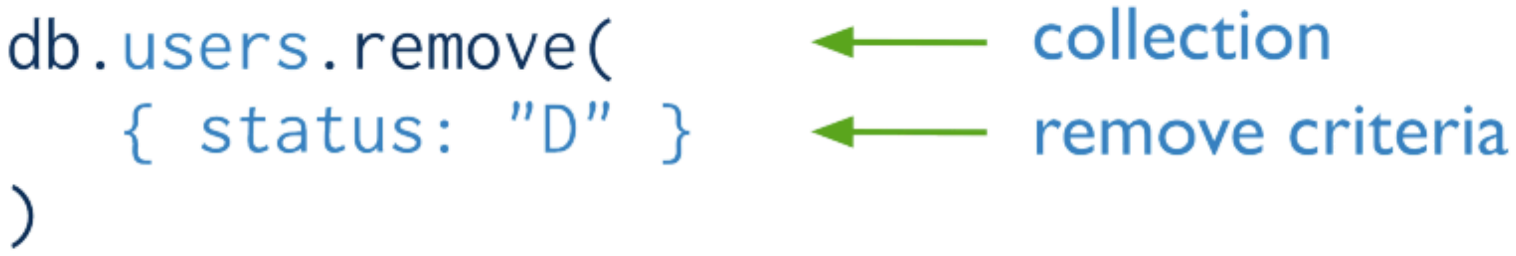
In SQL



```
UPDATE users
SET    status = 'A'
WHERE  age > 18
```

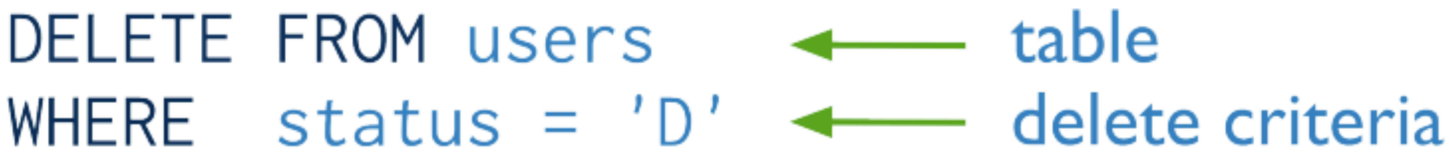
CRUD: Delete

In MongoDB



```
db.users.remove(
    { status: "D" }
)
```

In SQL



```
DELETE FROM users
WHERE  status = 'D'
```

Using MongoDB

- You can either install MongoDB on your machine, or visit: http://www.tutorialspoint.com/mongodb_terminal_online.php

To Get Started…

- Global commands: `help`, `exit`, etc.
- Commands execute against the current database are executed against the db object, for example:
  - `db.help()`: returns a list of commands that you can use against db object
  - Note: `db.help` without `()` gives you the method body

Create Database

- To create a wonderland database:

```
use wonderland
```

* creates the database and switches to it

- To get the collections in the current database:

```
db.getCollectionNames()
```

Insert Data

- To insert a document into the collection:

```
db.unicorns.insert(
    {name: ‘Aurora’,
    gender: ‘f’,
    weight: 450}
)
```

* Try out db.getCollectionNames() now, you’ll see:

List Documents in a Collection

- Try out:

```
db.unicorns.find()
```

- One more field is added: `_id`
  - Every document must have a unique `_id` field
  - Can generate your own or have MongoDB generate automatically for you

Remove all data: db.unicorns.remove({}).
Get the data from: http://bit.ly/iiitsdbms

Query Selector (cont.)

- Use

```
{field: value}
```

  to select documents that match the condition.

- If matching multiple conditions is desired:

```
{field1: value1, field2: value2…}
```

  * This implies the ______ statement

Comparison Operators in MongoDB

`$lt`	less than
`$lte`	less than or equal to
`$gt`	greater than
`$gte`	greater than or equal to
`$ne`	not equal to

(Q1) Find the male unicorns weigh more than 700 pounds

Ans1:

```
db.unicorns.find({gender: 'm', weight:
{$gt: 700}})
```

(Q2) Find the unicorns that have no vampire field

Ans 2:

```
db.unicorns.find({ vampires: {$exists: false}})
```

(Q3) Find the unicorns that like apples or oranges

Ans 3:

```
db.unicorns.find({ loves: {$in:
[‘apple','orange']}})
```

(Q4) Find the female unicorns that either love apples or weigh less than 500 pounds

Ans 4:

```
db.unicorns.find({gender: 'f', $or: [{loves: 'apple'}, {weight:
{$lt: 500}}]})
```

CRUD: Update

- Intuitively, updating unicorn Roooooodles' weight to 590 can be:

```
db.unicorns.update(
    {name: "Roooooodles"},
    {weight: 590})
```

- But if you try:

```
db.unicorns.find({name: "Roooooodles"})
```

the result will be: ____________________

CRUD: Update (cont.)

- The reason that no document was found was because the second parameter we supplied didn't have any update operators
- Therefore, the original document was replaced
- Try the following command to see:

```
db.unicorns.find({weight: 590})
```

CRUD: Update (cont.)

- To fix the problem, we should do:

```
db.unicorns.update({weight: 590},
 {$set: {name: “Roooooodles”,
         dob: new Date(1979, 7, 18, 18, 44),
         loves: [“apple”],
         gender: “m”,
         vampires: 99}})
```

CRUD: Update (cont.)

- The correct way to update at the beginning should therefore be:

```
db.unicorns.update({name: “Rooooodles”},
                  {$set: {weight: 590}})
```

More Update Operators

- `$inc`: increment a field by a certain positive or negative amount
- `$push`: add a value to the existing field

(Q5) Decrease unicorn Pilot’s number of vampires by 2

Ans 5:

```
db.unicorns.update({name: 'Pilot'}, {$inc: {vampires: -2}})
```

(Q6) Add “sugar” to the list of food unicorn Aurora loves to eat

Ans 6:

```
db.unicorns.update({name: 'Aurora'}, {$push: {loves: ‘sugar'}})
```

Projection

- find() can take a second argument, which is the project list
- Example:

  db.unicorns.find({}, {name:1, _id:0})

  – The values following field names are boolean:
    - 1 means including the field
    - 0 means excluding the field

  – Note that except excluding _id, the list cannot have a mixture of exclusion and inclusion

Upserts

- An `upsert` updates the document if found or inserts it if not
- To enable upserting we pass a third parameter to update `{upsert: true}`

Upserts (cont.)

- This will not do anything:

```
db.unicorns.update({name: "Walala"},
    {$inc: {vampires: 1}})
```

- Instead, do this:

```
db.unicorns.update({name: "Walala"},
    {$inc: {vampires: 1}},
    {upsert: true})
```

Multiple Updates

- By default, update will only update a single document. Passing the third parameter `{multi: true}` will enable the multiple update

(Q7) Give all of the unicorns vaccine (set vaccinated to be true)

Ans 7:

```
db.unicorns.update({}, {$set: {vaccinated: true }},
{multi:true});

db.unicorns.find({vaccinated: true});
```

(Q8) Sort the unicorns based on weights decreasingly

Ans 8:

```
db.unicorns.find().sort({weight: -1})
```

(Q9) Sort the unicorns based on the names increasingly, then the number of vampires decreasingly

Ans 9:

```
db.unicorns.find().sort({name: 1, vampires: -1})
```

(Q10) Get the second and third heaviest unicorns

Ans: 10

```
db.unicorns.find() .sort({weight: -1}) .limit(2) .skip(1)
```

(Q11) Count the number of unicorns who have more than 50 vampires

Ans 11:

```
db.unicorns.count({vampires: {$gt: 50}})
```

References

- Karl Seguin, *The Little MongoDB Book*, http://openmymind.net/mongodb.pdf
- Kristina Chodorow, *MongoDB: The Definite Guide*, O'Reilly
- MongoDB CRUD Operations, https://docs.mongodb.org/master/MongoDB-crud-guide-master.pdf