CSC 261/461 – Database Systems
Lecture 22

Spring 2018

Announcements

- Project 3 (MongoDB) is out
  - Due on April 19
  - Please go to workshop this week!
- Project 1 Milestone 4 is out too...
  - Due on last day of lecture
  - Extra credit for best projects
  - Demo during exam week.
- Term Paper:
  - Due date is approaching
  - Final submission is due on April 18, 2018 @ 11:59 pm. Please submit on time.
- Optional Project 4:
  - Spark
  - On bluehive
  - I will give you a very brief introduction
  - Will provide you a few Jupyter Notebook to practice.

Topics for Today

- MongoDb

MONGODB

What is MongoDB

- Scalable High-Performance Open-source, Document-orientated database.
- Built for Speed
- Rich Document based queries for Easy readability.
- Full Index Support for High Performance.
- Map / Reduce for Aggregation.

Why use MongoDB?

- SQL was invented in the 70's to store data.
- MongoDB stores documents (or) objects
- Embedded documents and arrays reduce need for joins

Why will we use Mongodb?

- Semi-Structured Content Management

XML -> Tables

- For Project 2 we have done:

Items -> User, Item, Category, Bid

XML Format

```
<Item ItemID="1043374545">
  <Name>christopher radko | fritz n_ frosty sledding</Name>
  <Category>Collectibles</Category>
  <Category>Decorative &amp;amp; Holiday</Category>
  <Category>Decorative by Brand</Category>
  <Category>Christopher Radko</Category>
  <Currently>$30.00</Currently>
  <First_Bid>$30.00</First_Bid>
  <Number_of_Bids>0</Number_of_Bids>
  <Bids/>
  <Location>its a dry heat</Location>
  <Country>USA</Country>
  <Started>Dec-03-01 18:10:40</Started>
  <Ends>Dec-13-01 18:10:40</Ends>
  <Seller UserID="rulabula" Rating="1035"/>
  <Description>brand new beautiful handmade european blown glass ornament
</Item>
```

JSON Format


```
{
    "_id": "1043374545",
    "Name": "christopher radko | fritz n_ frosty sl
    "Category": [
        "Collectibles",
        "Decorative & Holiday",
        "Decorative by Brand",
        "Christopher Radko"
    ],
    "Currently": 30.00,
    "First_Bid": 30.00,
    "Number_of_Bids": 0,
    "Bids": null,
    "Location": "its a dry heat",
    "Country": "USA",
    "Started": "Dec-03-01 18:10:40",
    "Ends": "Dec-13-01 18:10:40",
    "Seller": {
        "_id": "rulabula",
        "_Rating": 1035
    },
    "Description": "brand new beautiful handmade eu
 a snowman paired with a little girl bundled up in h
 ornament is approximately 5_ tall and 4_ wide. bran
ature black radko gift box. PLEASE READ CAREFULLY!!!
ear before shipping. the hold period will be a minim
 shipping rate is dependent on both the weight of th
arge is $6 and will increase with distance and weigh
will apply for all USPS shipments if you cannot have
and I_will furnish quotes on availability. the BUY-I
laska and hawaii receive a credit of like value appl
 utilize the feature. paypal is not accepted if you
buy it now feature is utilized. thank you for your u
See item description and Payment Instructions, or co
ct seller for more information."
 },
```

Object-relational impedance mismatch

- A set of conceptual and technical difficulties that are often encountered:
  - when a relational database management system (RDBMS) is being served by an application program (or multiple application programs) written in an object-oriented programming language
- Objects or class definitions must be mapped to database tables defined by relational schema.

MongoDB: No Impedance Mismatch

```
// your application code
class Foo { int x; string [] tags;}

// mongo document for Foo
{ x: 1, tags: [‘abc’,’xyz’] }
```

When I say

Database

Think

Database

- Made up of Multiple Collections.
- Created on-the-fly when referenced for the first time.

When I say

Collection

Think

Table

- Schema-less, and contains Documents.
- Indexable by one/more keys.
- Created on-the-fly when referenced for the first time.
- Capped Collections: Fixed size, older records get dropped after reaching the limit.

When I say

Document

Think

Record/Row

- Stored in a Collection.
- Have _id key – works like Primary keys in MySQL.
- Supported Relationships – Embedded (or) References.
- Document storage in BSON (Binary form of JSON).

The Document Model

```
var post = {
        '_id': ObjectId('3432'),
        'author': ObjectId('2311'),
        'title': 'Introduction to MongoDB',
        'body': 'MongoDB is an open sources.. ',
        'timestamp': Date('01-04-12'),
        'tags': ['MongoDB', 'NoSQL'],
        'comments': [{'author': ObjectId('5331'),
                                'date': Date('02-04-12'),
                                'text': 'Did you see.. ',
                                'upvotes': 7} ]
}

> db.posts.insert(post);
```

Find

// find posts which has ‘MongoDB’ tag.

> db.posts.find({tags: ‘MongoDB’});

// find posts by author’s comments.

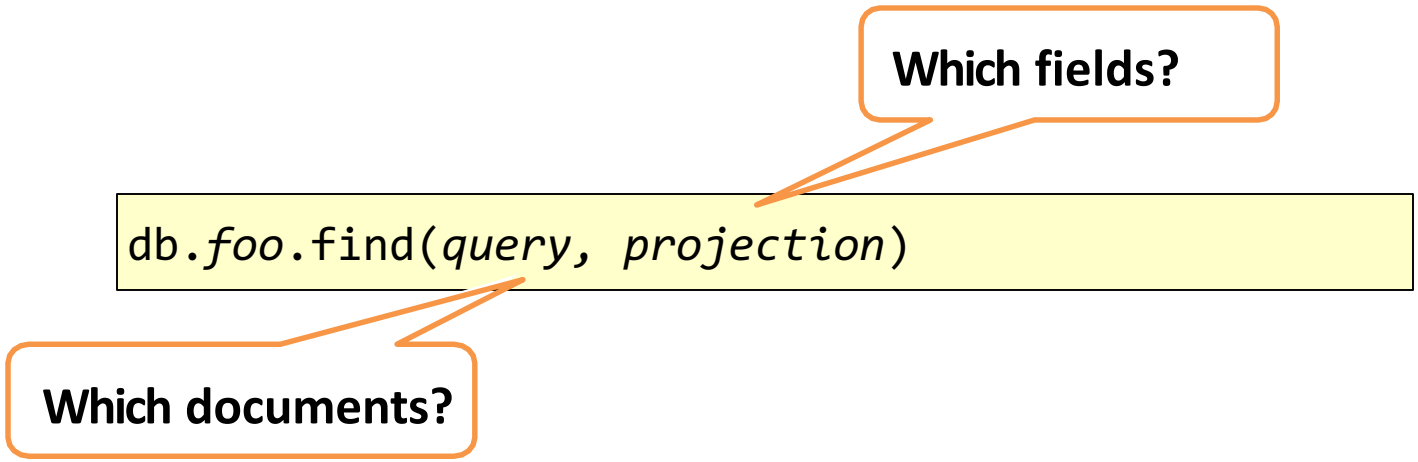
> db.posts.find({‘comments.author’: ‘Johnson’}).count();

// find posts written after 31st March.

> db.posts.find({‘timestamp’: {‘$gte’: Date(’31-03-12’)}});

$gt, $lt, $gte, $lte, $ne, $all, $in, $nin…

Find



Find: Projection

> db.posts.find({}, {title:1})

```
{ "_id" : ObjectId("5654381f37f63ffc4ebf1964"),
        "title" : "NodeJS server" }
{ "_id" : ObjectId("5654385c37f63ffc4ebf1965"),
        "title" : "Introduction to MongoDB" }
```

Like

select **title** from **posts**

Empty projection like

select * from **posts**

Find

Find

- Query criteria
  - Single value field
  - Array field
  - Sub-document /dot notation

Projection

- Field inclusion and exclusion

Cursor

- Sort
- Limit
- Skip

Update

```
> db.posts.update(
        {"_id" : ObjectId("5654381f37f63ffc4ebf1964")},
        {
                title:"NodeJS server"
        });
```

This will **replace** the document by {title:"NodeJS server"}

Update: Change part of the document

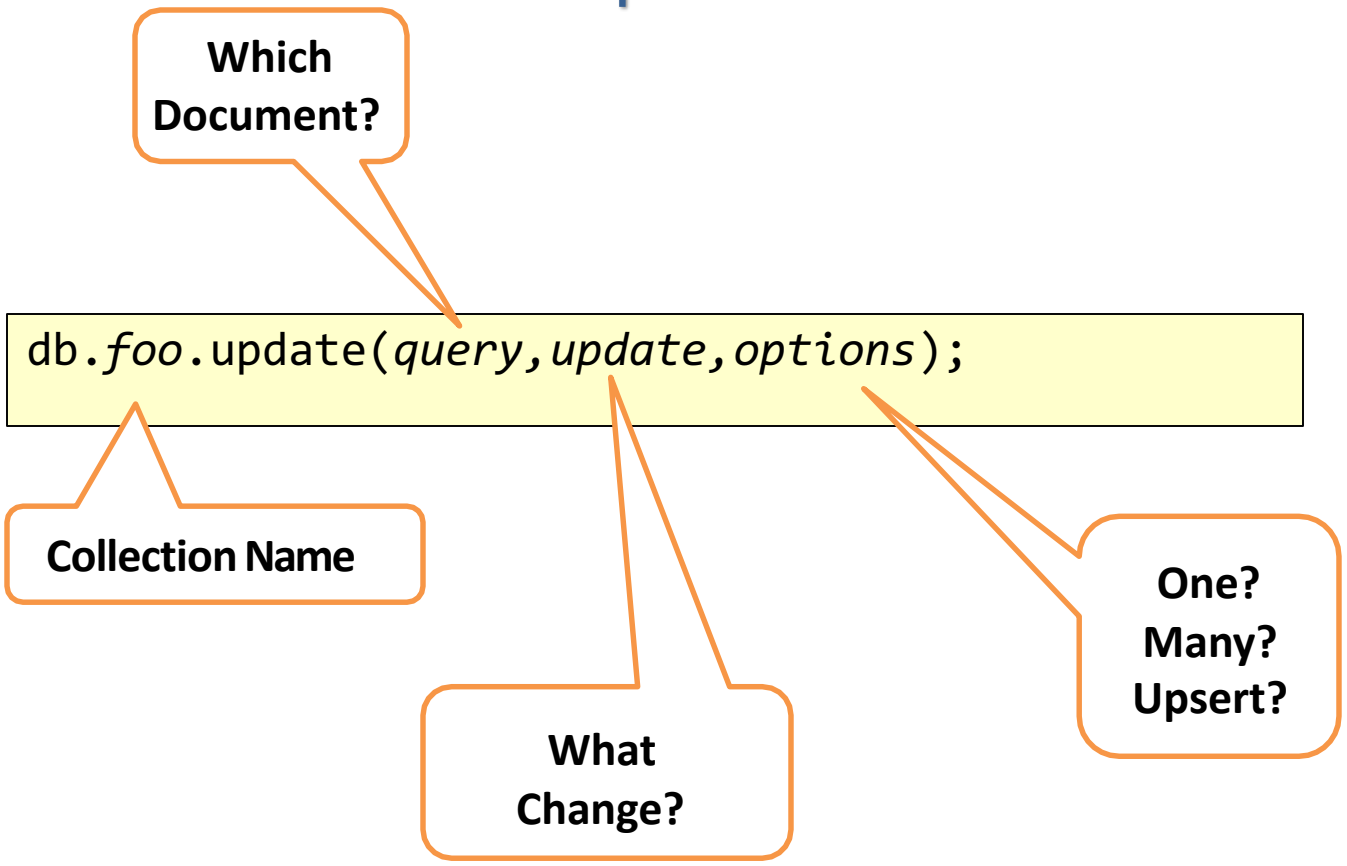
```
> db.posts.update(
        {"_id" : ObjectId("5654381f37f63ffc4ebf1964")},
        {
                $addToSet: {tags:"JS"},
                $set: {title:"NodeJS server"},
                $unset: { comments: 1}
        });
```

$set, $unset

$push, $pull, $pop, $addToSet

$inc, $decr, many more…

Update



Options:

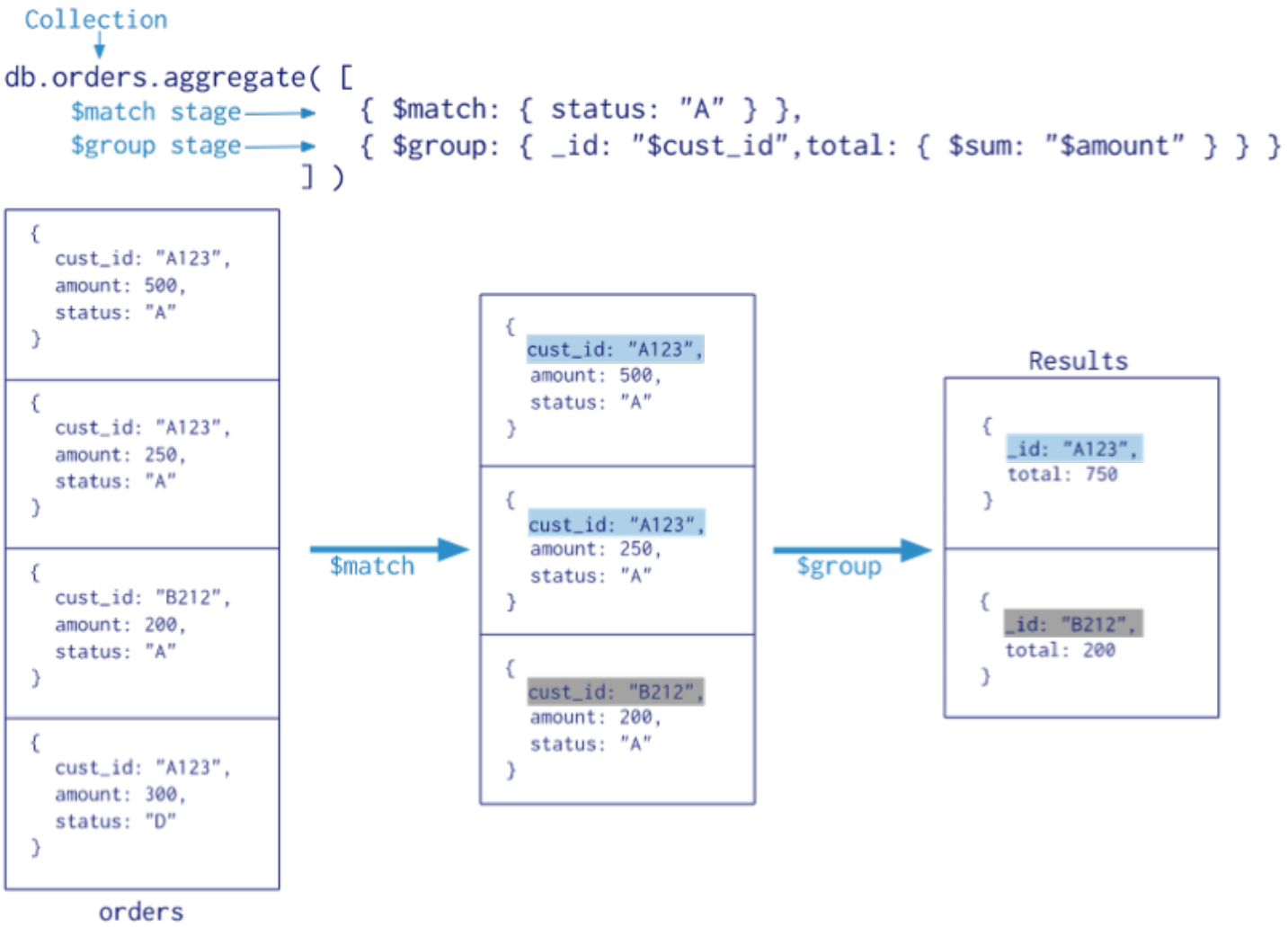
{multi: true} – will change all found documents;
by default only first found will be updated

{upsert: true} – will insert document if it was not found

Remove

- db.collection.remove(<query>, <justOne>)
- db.items.remove({Currently: { $gt: 20 } })

Aggregation



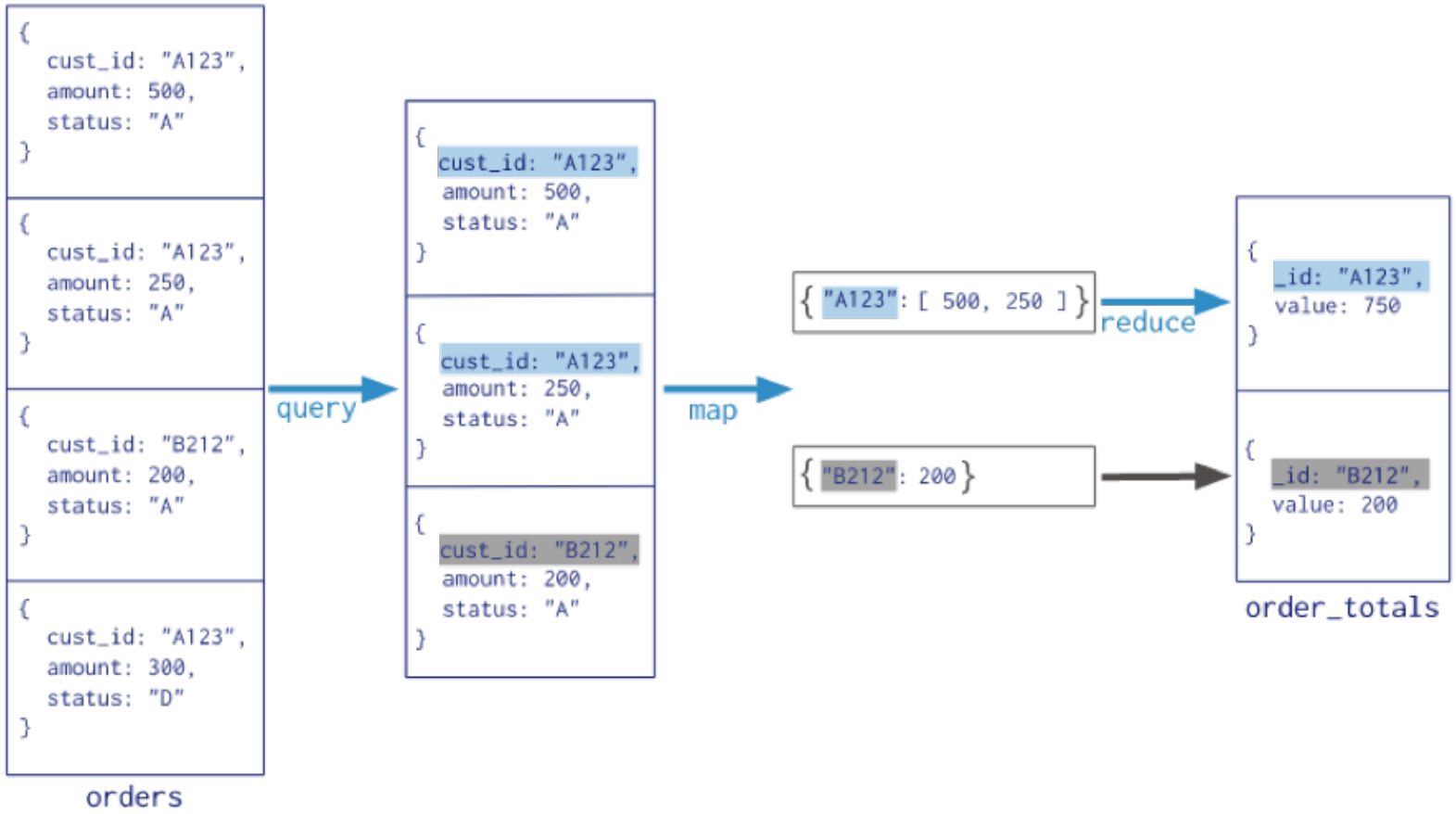
Aggregation

- https://docs.mongodb.com/v3.0/applications/aggregation/
- https://www.safaribooksonline.com/blog/2013/06/21/aggregation-in-mongodb/

MapReduce

Collection

```
db.orders.mapReduce(
        map      →   function() { emit( this.cust_id, this.amount ); },
        reduce   →   function(key, values) { return Array.sum( values ) },
                     {
        query    →      query: { status: "A" },
        output   →      out: "order_totals"
                     }
                 )
```



Acknowledgement

- Some of the slides in this presentation are taken from the slides provided by the authors.
- Many of these slides are taken from cs145 course offered by Stanford University.