#mongodubai

Data Processing and Aggregation

Norberto Leite

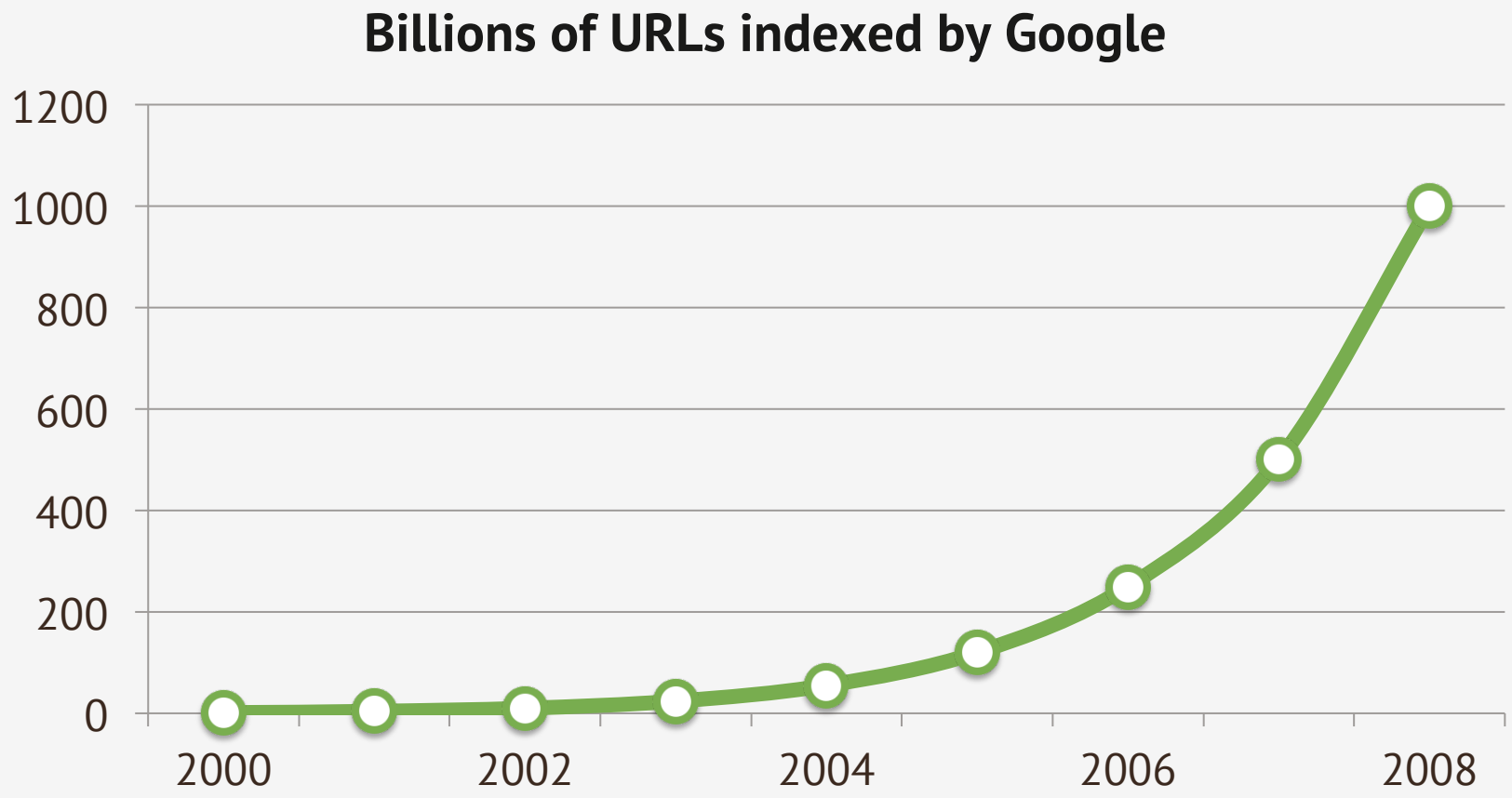
Senior Solutions Architect, MongoDB

mongoDB

Big Data

mongoDB

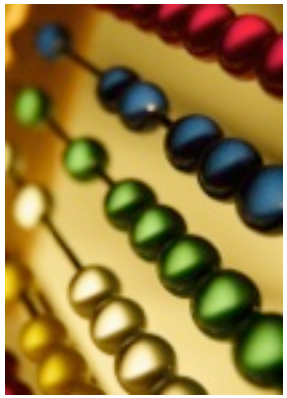
Big data is like teenage sex:
everyone talks about it, nobody
really knows how to do it, everyone
thinks everyone else is doing it, so
everyone claims they are doing it ...



Exponential data growth

For over a decade

Big Data == Custom Software



amazon web services

Amazon DynamoDB

Google Bigtable

In the past few years

Open source software has emerged enabling the rest of us to handle **Big Data**

How MongoDB solves our needs

- MongoDB is an ideal **operational** database
- MongoDB provides **high performance** for **storage and retrieval** at large scale
- MongoDB has a robust query interface permitting intelligent operations
- MongoDB is ***not*** a data processing engine, but provides processing functionality

MongoDB data processing options

http://www.flickr.com/photos/torek/4444673930/

Getting example data

mongoDB

The "hello world" of map reduce is counting words in a paragraph of text.

We could do that but lets do something a little more **interesting**...

What's the most popular pub name?

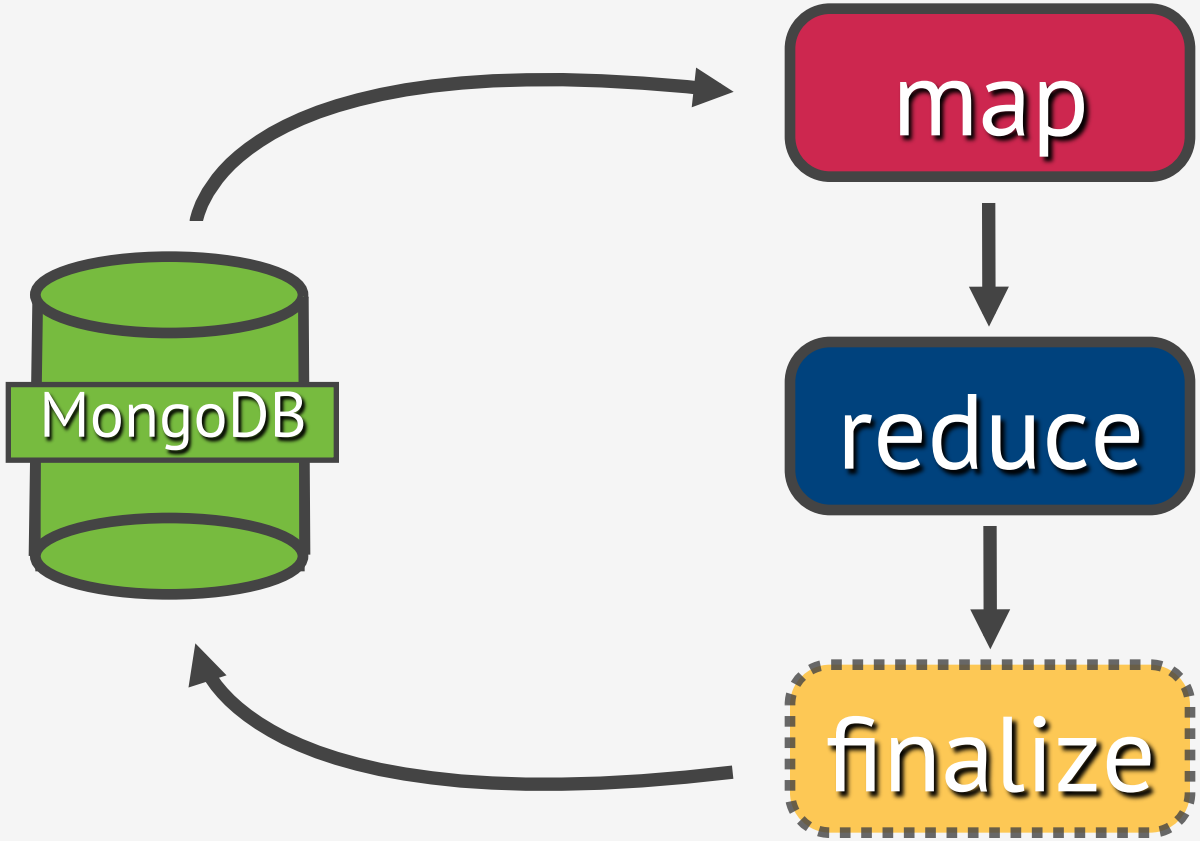
- http://www.flickr.com/photos/dayoff171/5670631538/

Open Street Map data

```
#!/usr/bin/env python

# Data Source
# http://www.overpass-api.de/api/xapi?*[amenity=pub][bbox=-10.5,49.78,1.78,59]

import re
import sys

from imposm.parser import OSMParser
import pymongo

class Handler(object):
    def nodes(self, nodes):
        if not nodes:
            return
        docs = []
        for node in nodes:
            osm_id, doc, (lon, lat) = node
            if "name" not in doc:
                node_points[osm_id] = (lon, lat)
                continue
            doc["name"] = doc["name"].title().lstrip("The ").replace("And", "&")
            doc["_id"] = osm_id
            doc["location"] = {"type": "Point", "coordinates": [lon, lat]}
            docs.append(doc)
        collection.insert(docs)
```

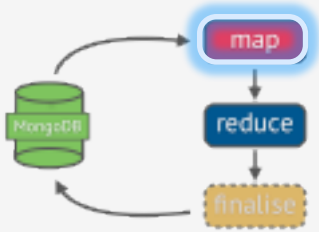
Example pub data


```
{
   "_id" : 451152,
   "amenity" : "pub",
   "name" : "The Dignity",
   "addr:housenumber" : "363",
   "addr:street" : "Regents Park Road",
   "addr:city" : "London",
   "addr:postcode" : "N3 1DH",
   "toilets" : "yes",
   "toilets:access" : "customers",
   "location" : {
      "type" : "Point",
      "coordinates" : [-0.1945732, 51.6008172]
   }
}
```



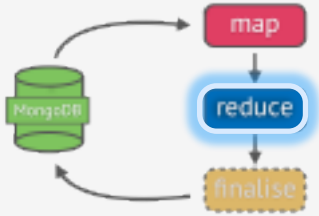
MongoDB Map/Reduce

Map Function




```
> var map = function() {
    emit(this.name, 1);
```

Reduce Function




```
> var reduce = function (key, values) {
    var sum = 0;
    values.forEach( function (val) {sum += val;} );
    return sum;
  }
```

Execute MongoDB Map Reduce

```
> db.pubs.mapReduce(map, reduce, {out: "pub_names"})

{
    "result" : "pub_names",
    "timeMillis" : 2042,
    "counts" : {
        "input" : 33142,
        "emit" : 33142,
        "reduce" : 5235,
        "output" : 16176
    },
    "ok" : 1,
}
```

Results

```
> db.pub_names.find().sort({value: -1}).limit(10)

{ "_id" : "The Red Lion", "value" : 407 }
{ "_id" : "The Royal Oak", "value" : 328 }
{ "_id" : "The Crown", "value" : 242 }
{ "_id" : "The White Hart", "value" : 214 }
{ "_id" : "The White Horse", "value" : 200 }
{ "_id" : "The New Inn", "value" : 187 }
{ "_id" : "The Plough", "value" : 185 }
{ "_id" : "The Rose & Crown", "value" : 164 }
{ "_id" : "The Wheatsheaf", "value" : 147 }
{ "_id" : "The Swan", "value" : 140 }
```


The Wheatsheaf
The Red Lion
The Royal Oak
The White Horse
The Coach And Horses
The Rising Sun
The Rose And Crown
The Crown Inn
The Crown
The White Hart
The New Inn
The Prince of Wales
The Plough
The Ship
The Bull
The Cricketers
The Woolpack
The Castle Inn
The Railway Tavern
The Half Moon
The Barley Mow
The Blue Bell
The Lord Nelson
The King'S Head
The Griffin
The Foresters Arms
The Globe
The Five Bells
The Fountain
The Albion
The Hare And Hounds
The Anchor
The Star Inn
The Golden Lion
The Victoria
The Castle
The Queens Head
The Kings Arms
The White Swan
The Plough Inn
The Masons Arms
The Green Dragon
The Dolphin
The Village Inn
The George & Dragon
The Black Lion
The Fox
The Greyhound
The Angel
The Bridge Inn
The George
The Sun
The Ship Inn
The Swan
The Cross Keys
The White Lion
The Fox And Hounds
The Windmill
The Railway Inn
The Station
The Queen'S Head
The Railway
The Swan Inn
The Bell Inn
The Duke Of Wellington
The Waggon And Horses
The Seven Stars
The Three Tuns
The Sun Inn
The King'S Arms
The George And Dragon
The Coach & Horses
The Black Swan
The Miners Arms
The Black Bull
The Beehive
The Lamb Inn
The Star
The Black Horse
The Railway House
The Bird In Hand
The George Inn
The Butchers Arms
The Kings Head
The Lamb
The Bell
The Cheuers
The Nags Head
The Green Man
The Buds Head
The Rose & Crown
The Anchor Inn
The Bay Horse
The Duke Of York
The Woodman
The Robin Hood
The Shoulder Of Mutton
The Three Horseshoes
The Horse And Jockey
Three Horseshoes

mongoDB

Pub names in the center of London

```
> db.pubs.mapReduce(map, reduce, { out: "pub_names",
    query: {
     location: {
      $within: { $centerSphere: [[-0.12, 51.516], 2 / 3959] }
     }}
  })

{
    "result" : "pub_names",
    "timeMillis" : 116,
    "counts" : {
        "input" : 643,
        "emit" : 643,
        "reduce" : 54,
        "output" : 537
    },
    "ok" : 1,
}
```

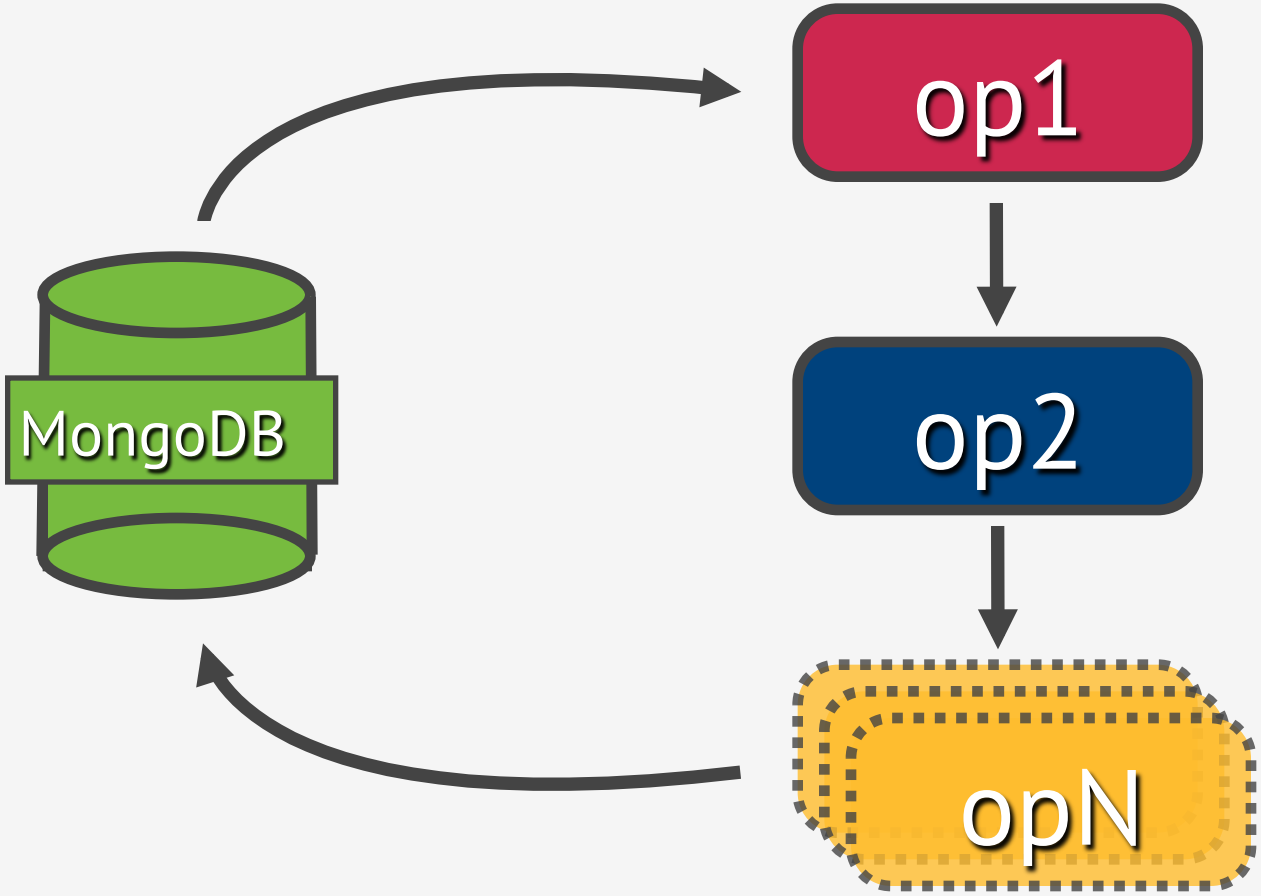
Results

```
> db.pub_names.find().sort({value: -1}).limit(10)

{ "_id" : "All Bar One", "value" : 11 }
{ "_id" : "The Slug & Lettuce", "value" : 7 }
{ "_id" : "The Coach & Horses", "value" : 6 }
{ "_id" : "The Green Man", "value" : 5 }
{ "_id" : "The Kings Arms", "value" : 5 }
{ "_id" : "The Red Lion", "value" : 5 }
{ "_id" : "Corney & Barrow", "value" : 4 }
{ "_id" : "O'Neills", "value" : 4 }
{ "_id" : "Pitcher & Piano", "value" : 4 }
{ "_id" : "The Crown", "value" : 4 }
```

MongoDB Map / Reduce

- Real-time
- Output directly to document or collection
- Runs inside MongoDB on local data

- Adds load to your DB
- In javascript - debugging can be a challenge
- Have to translate in and out of c++



Aggregation Framework

Aggregation Framework in 60 seconds

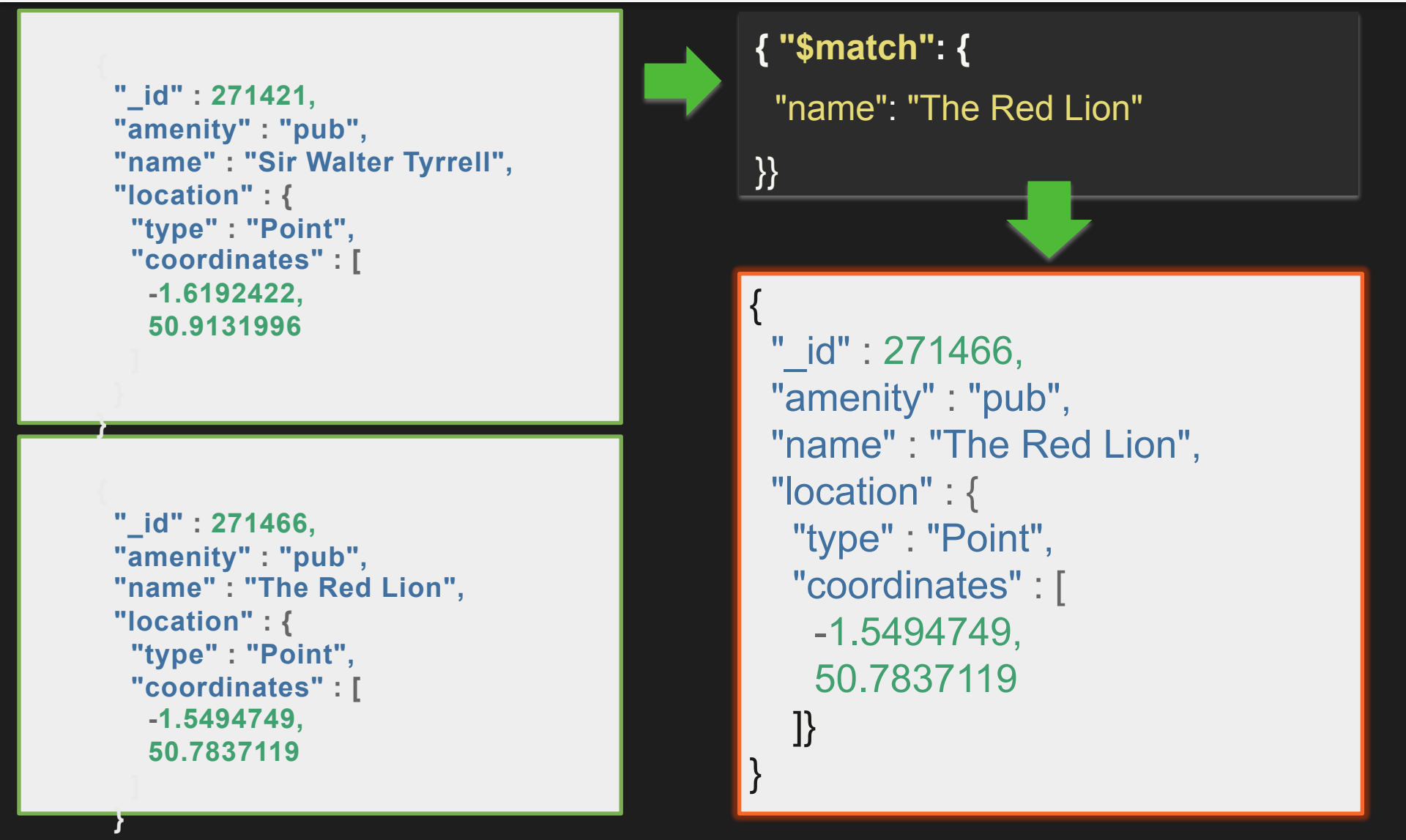
Aggregation framework operators

- $project
- $match
- $limit
- $skip
- $sort
- $unwind
- $group

$match

- Filter documents
- Uses existing query syntax
- If using $geoNear it has to be first in pipeline
- $where not supported

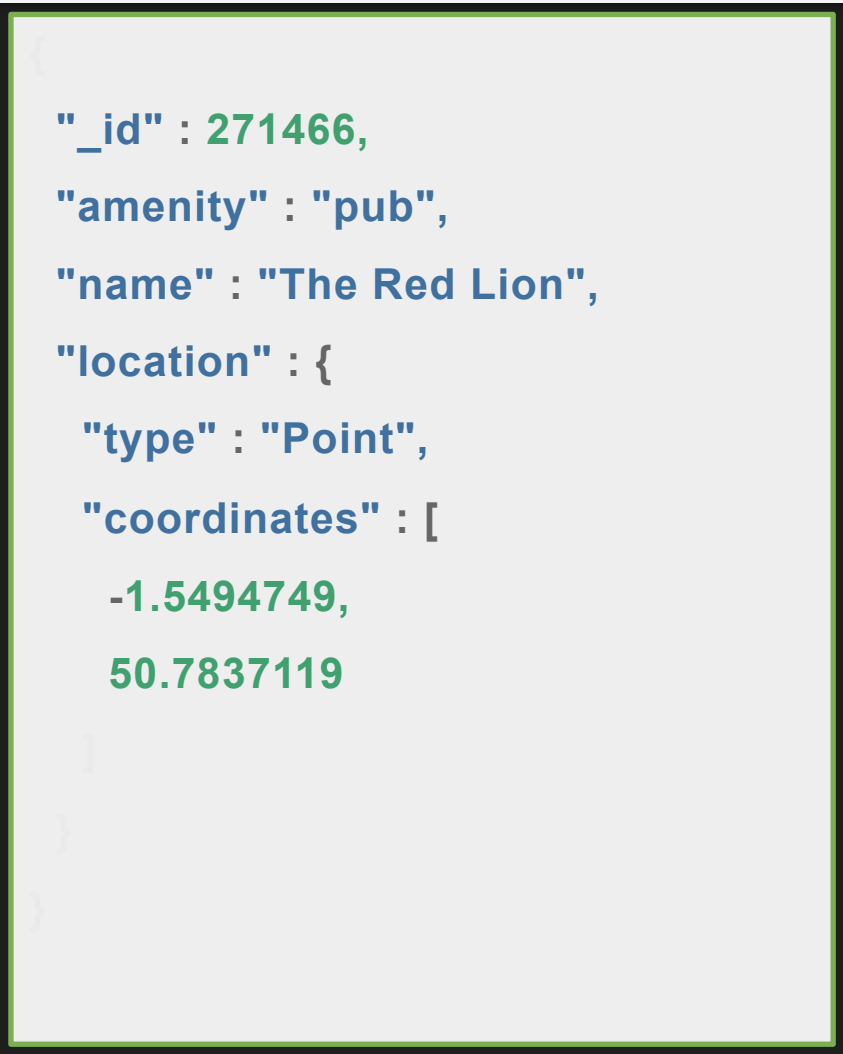
Matching Field Values



$project

- Reshape documents
- Include, exclude or rename fields
- Inject computed fields
- Create sub-document fields

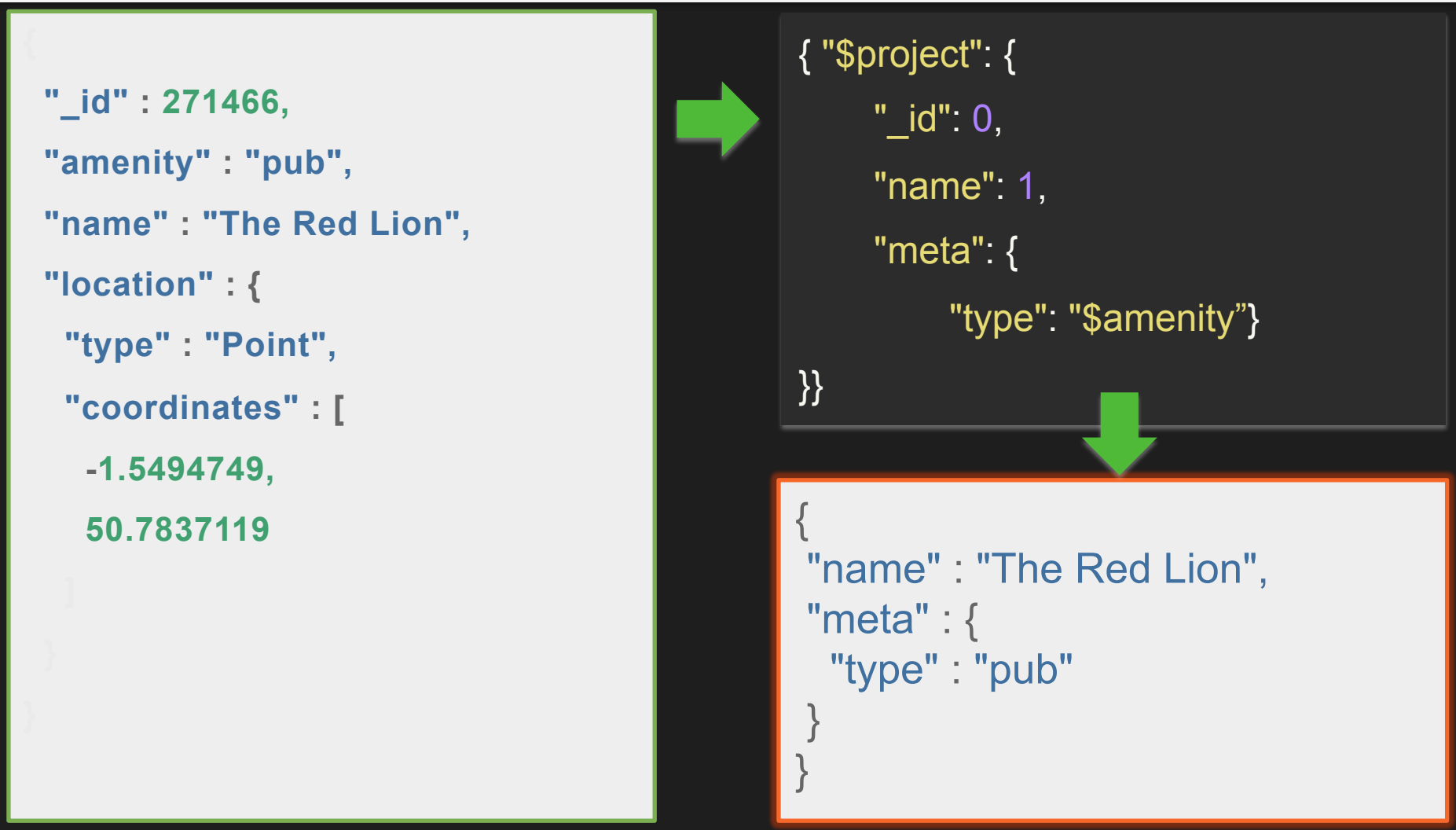
Including and Excluding Fields



```
{ "$project": {
  "_id": 0,
  "amenity": 1,
  "name": 1
}}
```

```
{
  "amenity" : "pub",
  "name" : "The Red Lion"
}
```

Reformatting documents



Dealing with arrays

```
{
"_id" : 271466,
"amenity" : "pub",
"name" : "The Red Lion",
"facilities" : [
  "toilets",
  "food"
],
}
```

```
{ "$project": {
        "_id": 0,
        "name": 1,
        "facility": "$facilities"
}},
{"$unwind": "$facility"}
```

```
{ "name" : "The Red Lion",
  "facility" : "toilets" },
```

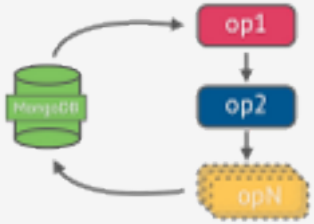
```
{ "name" : "The Red Lion",
  "facility" : "food" }
```

$group

- Group documents by an ID
- Field reference, object, constant
- Other output fields are computed
  $max, $min, $avg, $sum
  $addToSet, $push $first, $last
- Processes all data in memory

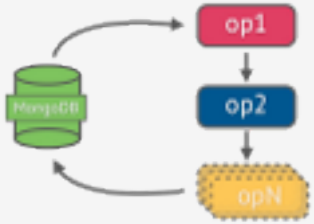
Back to the pub!

Popular pub names




```
> var popular_pub_names = [
    { $match : { location:
                 { $within: { $centerSphere:
                   [[-0.12, 51.516], 2 / 3959] }}}
    },
    { $group :
      { _id : "$name",
        value : { $sum : 1 } }
    },
    { $sort : { value : -1 } },
    { $limit : 10 }
  ]
```

Results




```
> db.pubs.aggregate(popular_pub_names)
{
    "result" : [
        { "_id" : "All Bar One", "value" : 11 }
        { "_id" : "The Slug & Lettuce", "value" : 7 }
        { "_id" : "The Coach & Horses", "value" : 6 }
        { "_id" : "The Green Man", "value" : 5 }
        { "_id" : "The Kings Arms", "value" : 5 }
        { "_id" : "The Red Lion", "value" : 5 }
        { "_id" : "Corney & Barrow", "value" : 4 }
        { "_id" : "O'Neills", "value" : 4 }
        { "_id" : "Pitcher & Piano", "value" : 4 }
        { "_id" : "The Crown", "value" : 4 }
    ],
    "ok" : 1
}
```

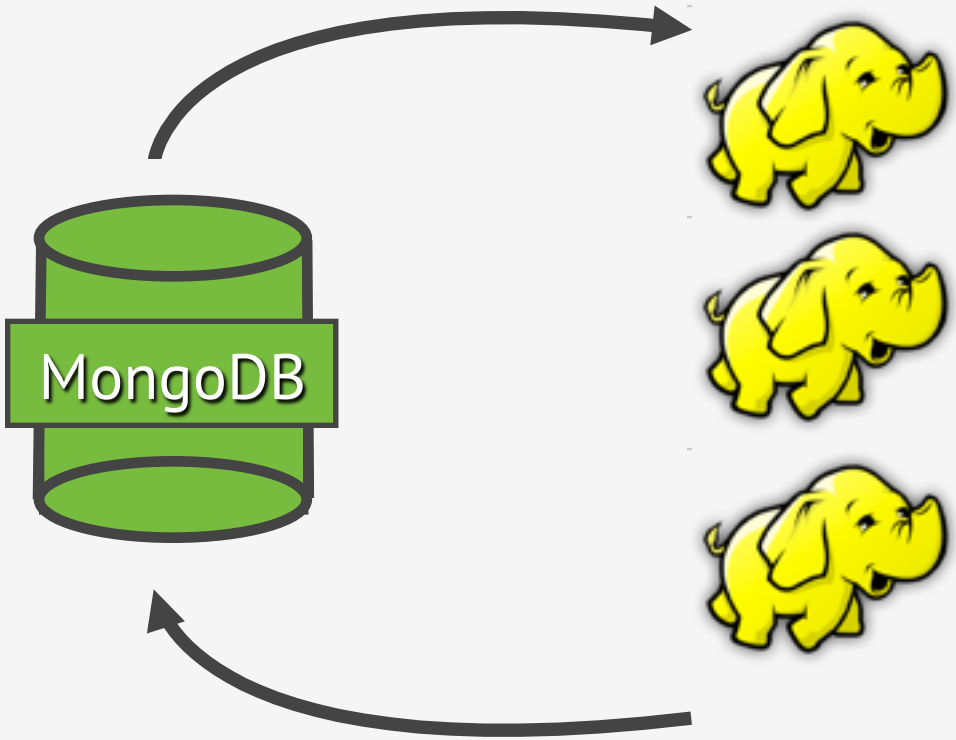
Aggregation Framework Benefits

- Real-time
- Simple yet powerful interface
- Declared in JSON, executes in C++
- Runs inside MongoDB on local data

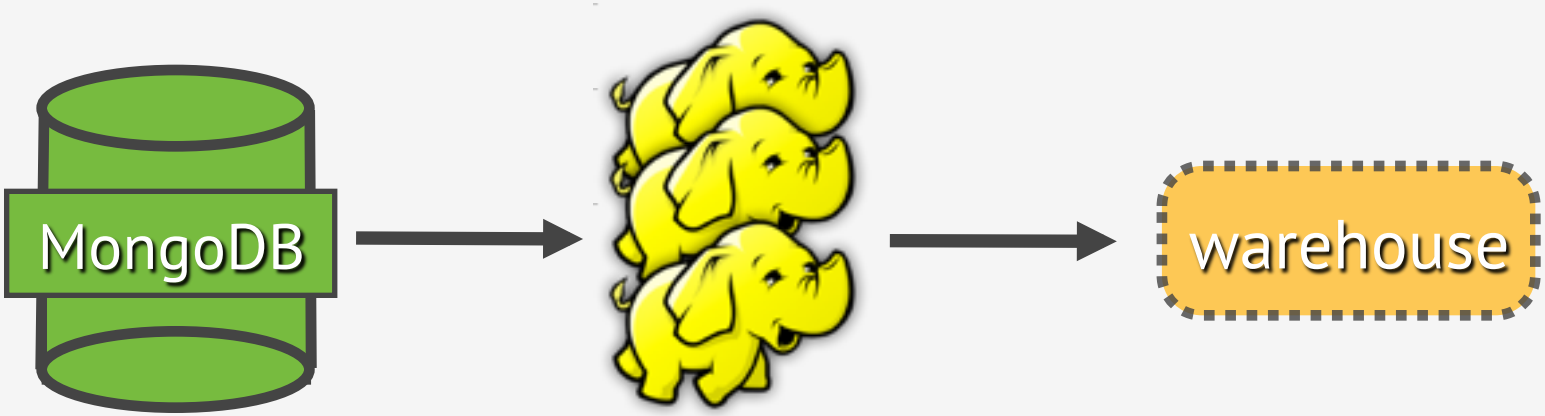
- Adds load to your DB
- Limited operators
- Limited how much data it can return

Analysing MongoDB Data in External Systems

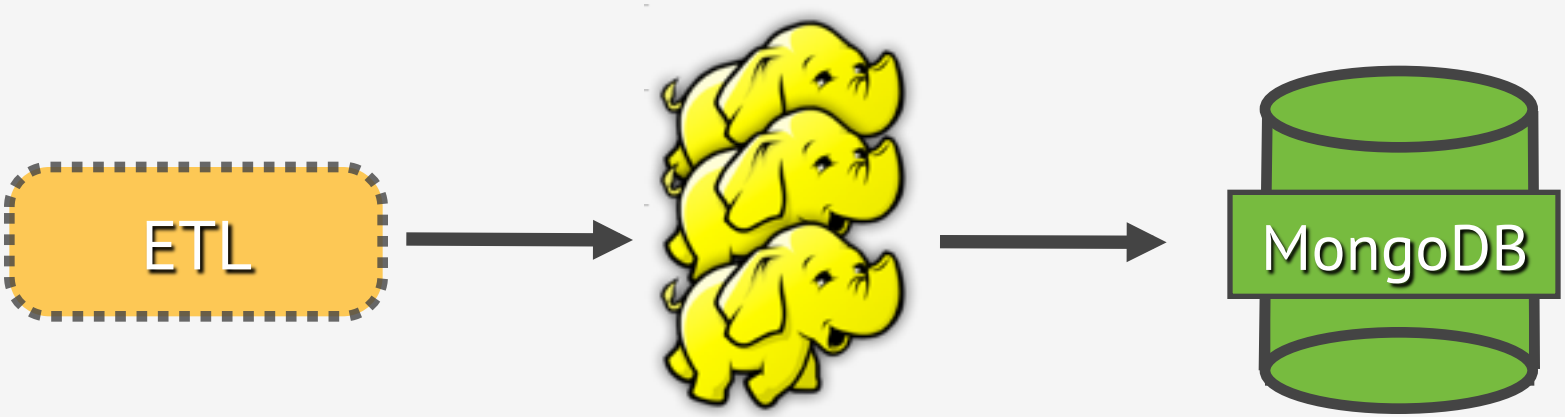
mongoDB



MongoDB with Hadoop



MongoDB with Hadoop



MongoDB with Hadoop

Map pub names in Python


```
#!/usr/bin/env python
from pymongo_hadoop import BSONMapper

def mapper(documents):
    bounds = get_bounds() # ~2 mile polygon
    for doc in documents:
        geo = get_geo(doc["location"]) # Convert the geo type
        if not geo:
            continue
        if bounds.intersects(geo):
            yield {'_id': doc['name'], 'count': 1}

BSONMapper(mapper)
print >> sys.stderr, "Done Mapping. "
```

Reduce pub names in Python


```
#!/usr/bin/env python

from pymongo_hadoop import BSONReducer

def reducer(key, values):
    _count = 0
    for v in values:
        _count += v['count']
    return {'_id': key, 'value': _count}

BSONReducer(reducer)
```

Execute M/R


```
hadoop jar target/mongo-hadoop-streaming-assembly-1.0.0-rc0.jar \
-mapper examples/pub/map.py \
-reducer examples/pub/reduce.py \
-mongo mongodb://127.0.0.1/demo.pubs \
-outputURI mongodb://127.0.0.1/demo.pub_names
```

Popular pub names nearby

```
> db.pub_names.find().sort({value: -1}).limit(10)

{ "_id" : "All Bar One", "value" : 11 }
{ "_id" : "The Slug & Lettuce", "value" : 7 }
{ "_id" : "The Coach & Horses", "value" : 6 }
{ "_id" : "The Kings Arms", "value" : 5 }
{ "_id" : "Corney & Barrow", "value" : 4 }
{ "_id" : "O'Neills", "value" : 4 }
{ "_id" : "Pitcher & Piano", "value" : 4 }
{ "_id" : "The Crown", "value" : 4 }
{ "_id" : "The George", "value" : 4 }
{ "_id" : "The Green Man", "value" : 4 }
```

MongoDB and Hadoop

- Away from data store
- Can leverage existing data processing infrastructure
- Can horizontally scale your data processing

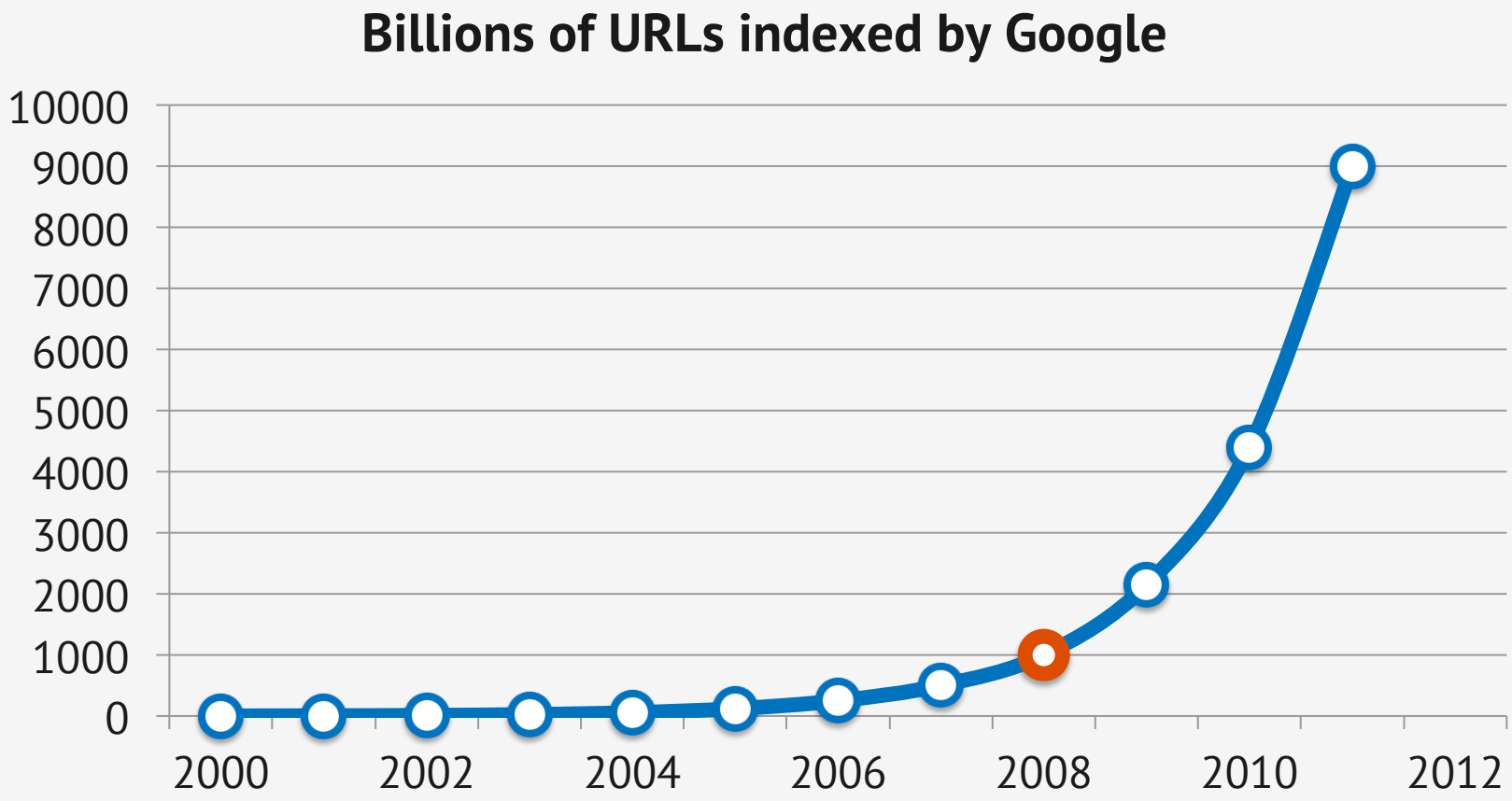
- Offline batch processing
- Requires synchronisation between store & processor
- Infrastructure is much more complex

The Future of Big Data and MongoDB

mongoDB

What is Big Data?

Big today is normal tomorrow



Exponential data growth

90% of the data in the world today has been created in the last two years

IBM - http://www-01.ibm.com/software/data/bigdata/

MongoDB enables

you to **scale** to the redefinition of **BIG**.

MongoDB is evolving

to enable you to process the new **BIG**.

Data Processing with MongoDB

- Process **in** MongoDB using **Map/Reduce**
- Process **in** MongoDB using **Aggregation Framework**
- Process **outside** MongoDB using **Hadoop** and other external tools

We are committed to working with the best data processing tools

- **Hadoop**
  https://github.com/mongodb/mongo-hadoop
- **Storm**
  https://github.com/christkv/mongo-storm
- **Disco**
  https://github.com/mongodb/mongo-disco
- **Spark**
  Coming soon!

#mongodubai

Obrigado!

Norberto Leite

Senior Solutions Architect, MongoDB

@nleite

norberto@mongodb.com

mongoDB