

Monary

Scientific analysis with MongoDB and Numpy

OSDC 2015

Imagine a world where...

- You are a data scientist
- You have taken a cab in New York

... and you want to analyze data

How can you do that?

- Need a tool to run analysis
- Need a place to store it

What to use for analysis?

To take an average...

- List of Python Dictionaries
~12 million numbers/sec

```
[{"a": 1}, {"a": 2}, {"a": 3}]
```

What to use for analysis?

To take an average...

- List of Python Dictionaries
~12 million numbers/sec



What to use for analysis?

To take an average...

- List of Python Dictionaries
~12 million numbers/sec
- Python List
110 million numbers/sec



[1,2,3]

What to use for analysis?

To take an average...

- List of Python Dictionaries
~12 million numbers/sec
- Python List
110 million numbers/sec



What to use for analysis?

To take an average...

- List of Python Dictionaries
~12 million numbers/sec
- Python List
110 million numbers/sec
- `numpy.ndarray`
500 million numbers/sec [1,2,3]



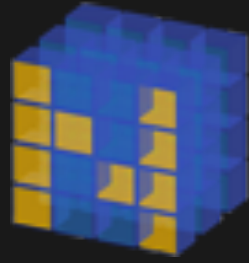
What to use for analysis?

To take an average...

- List of Python Dictionaries
~12 million numbers/sec
- Python List
110 million numbers/sec
- `numpy.ndarray`
500 million numbers/sec



Numpy



Numerical Python:

Scientific computing with Python

- **ndarray:** n-dimensional array. High performance, C-style arrays
- extensive built-in math libraries

Storage

Where to store our data?

MongoDB



Open-source document database

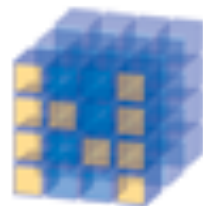
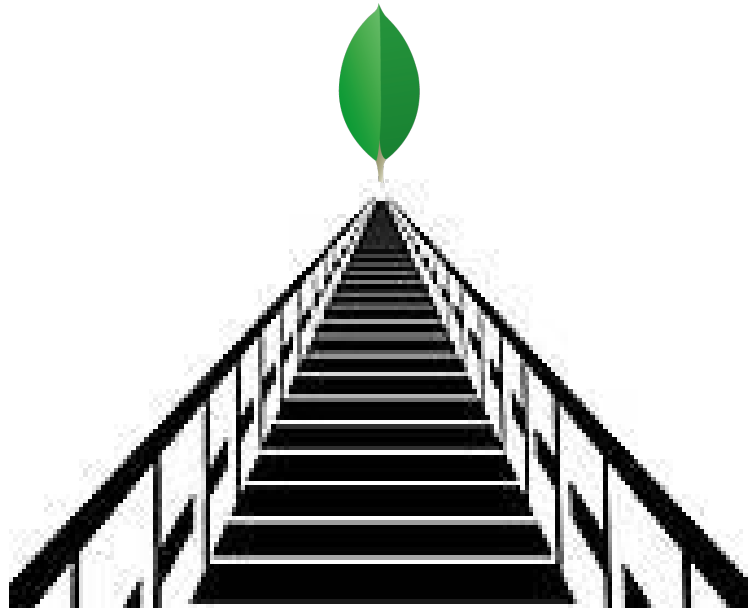
- data stored as BSON

```
{  
  "name": "anna",  
  "height": 67,  
}
```

Why MongoDB?

- **Easy to use!**
- Powerful query language
 - built-in geo queries
- Sophisticated aggregations

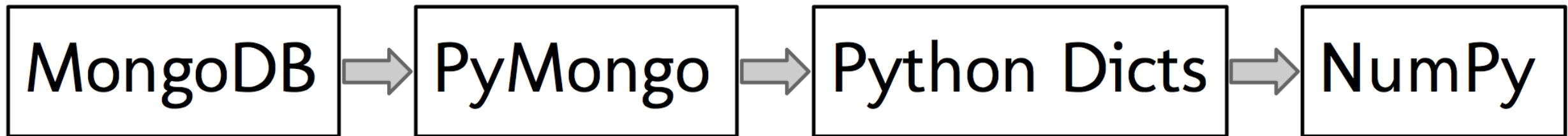
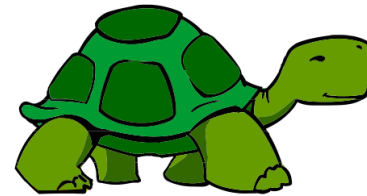
Now what?



NumPy

PyMongo

Official MongoDB Python Driver



About 150,000 documents read per second

Can we do better?

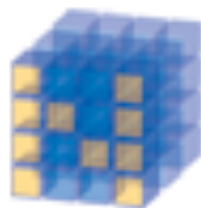
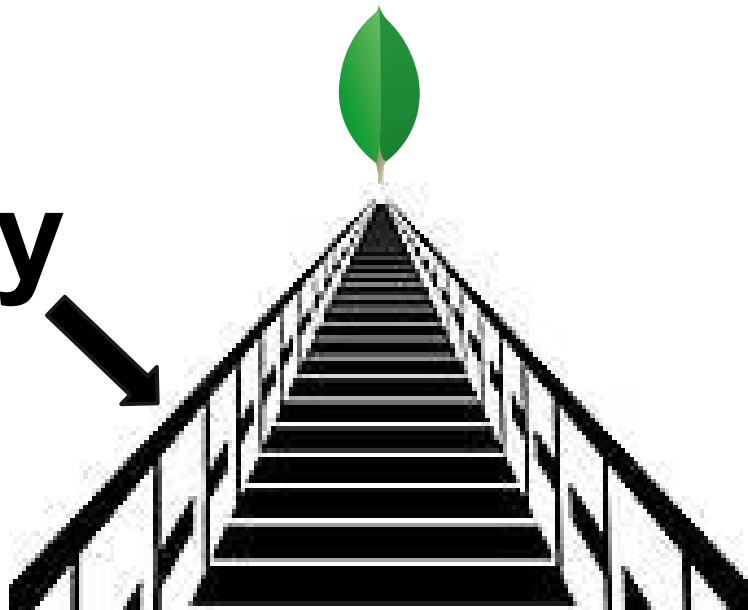


Monary



Over **1,700,000** reads per second

Monary



NumPy

Monary Overview

- author: David J. C. Beach
- operations: CRUD
- dependencies: NumPy, C Driver 1.0

Monary Features

- Insert
- Remove
- Queries
- Aggregation Framework
- **DB Commands**

Monary Queries

Say you have a DB, “test”, with documents:

```
{ "a": 5,  
  "b": "hello!",  
  "c": [1.2, 2.5, 3.1, 4.6, 5.7] }
```

...and you want to get a closer look at all documents for which $a = 5$



Monary Queries

```
with Monary() as m:
```

```
    m.query("test",  Database)
```




Monary Queries

```
with Monary() as m:
```

```
    m.query("test",  Database  
               "collection",  Collection
```

Monary Queries

```
with Monary() as m:
```

```
    m.query("test",  Database  
        "collection",  Collection  
        {"a": 5},  Query
```


Monary Queries


```
with Monary() as m:
```

```
    m.query("test",  Database
```

```
        "collection",  Collection
```

```
        {"a": 5},  Query
```

```
        ["b"],  Field Name
```

```
        ["string"])  Return Type
```

Multiple Field Queries

```
with Monary() as m:
```

```
    m.query("test",  Database
```

```
        "collection",  Collection
```

```
        {"a": 5},  Query
```

```
        ["b", "c"],  Field Name
```

```
        ["string", "5float64"] )  Return  
                                Type
```

How Does Monary Work?

Python

- **numpy**: allocates ndarrays
- **Ctypes**: passes pointers to ndarrays into C
- *Passes query to CMongo and receives data back*

C

- **CMongo**: MongoDB C Driver
- **Libbson**: BSON Library
- *Populates arrays allocated by numpy with data retrieved from MongoDB*

How Does Monary Work?

Scientist

query +
ndarray
pointer

Monary
Python

CMonary

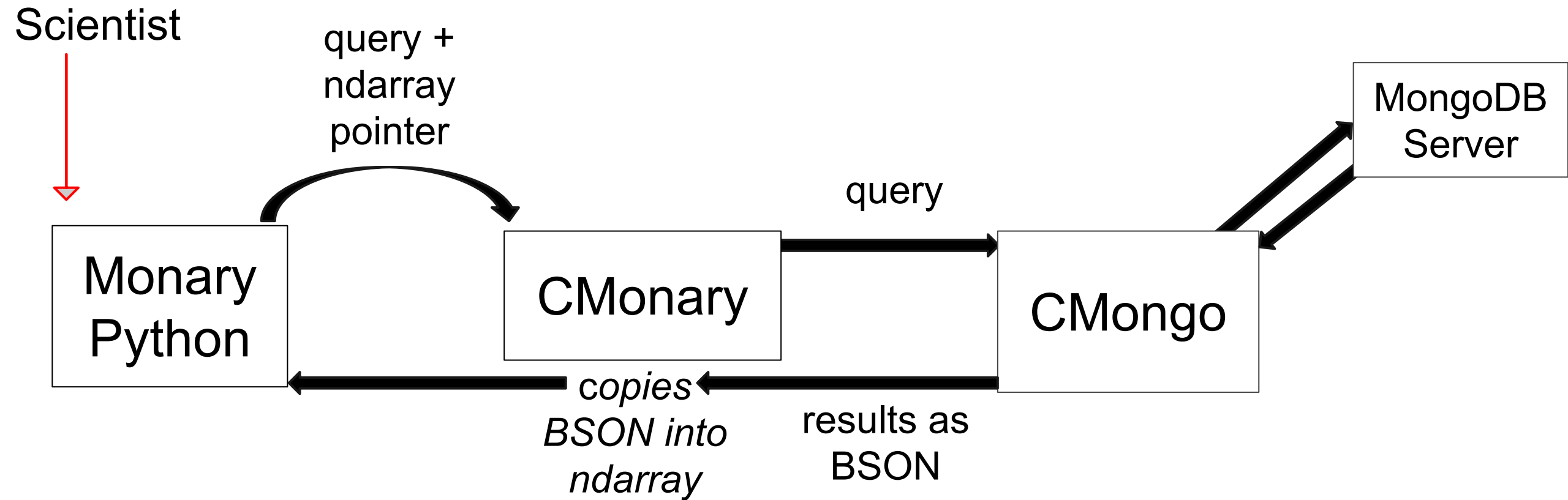
query

CMongo

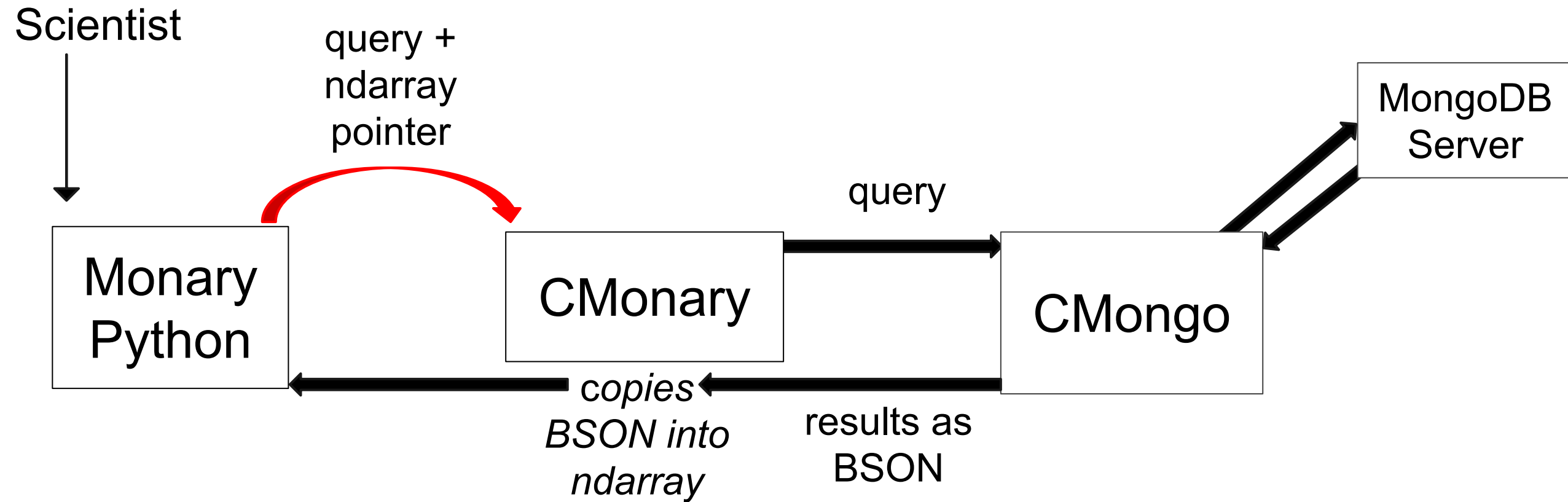
MongoDB
Server

*copies
BSON into
ndarray*

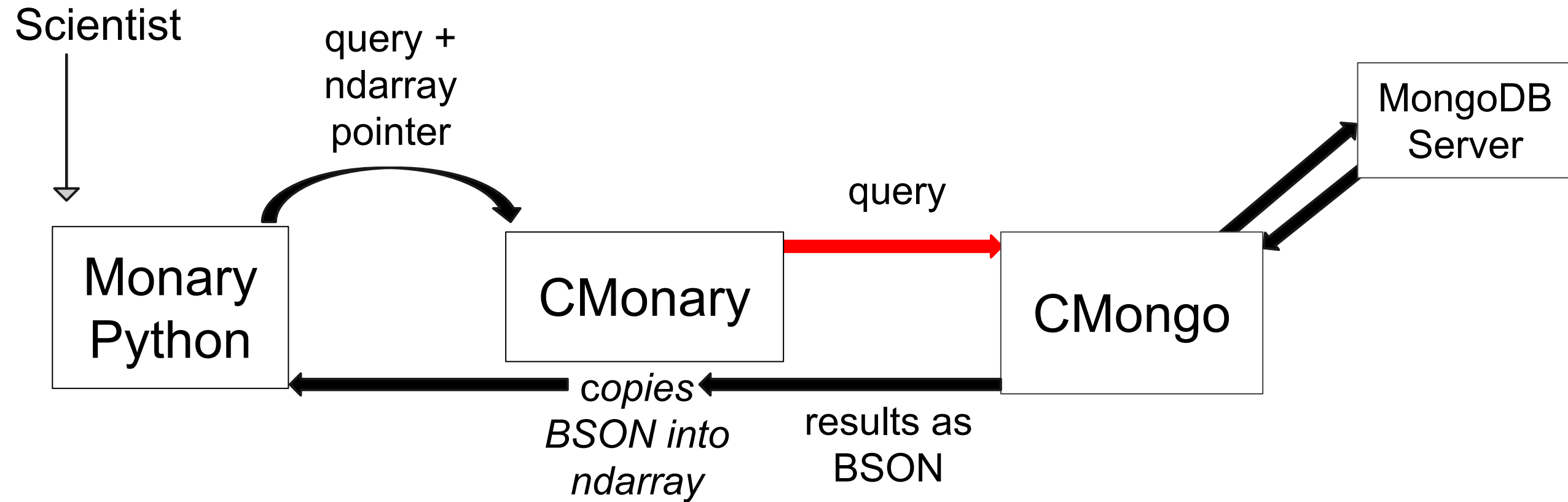
*results as
BSON*



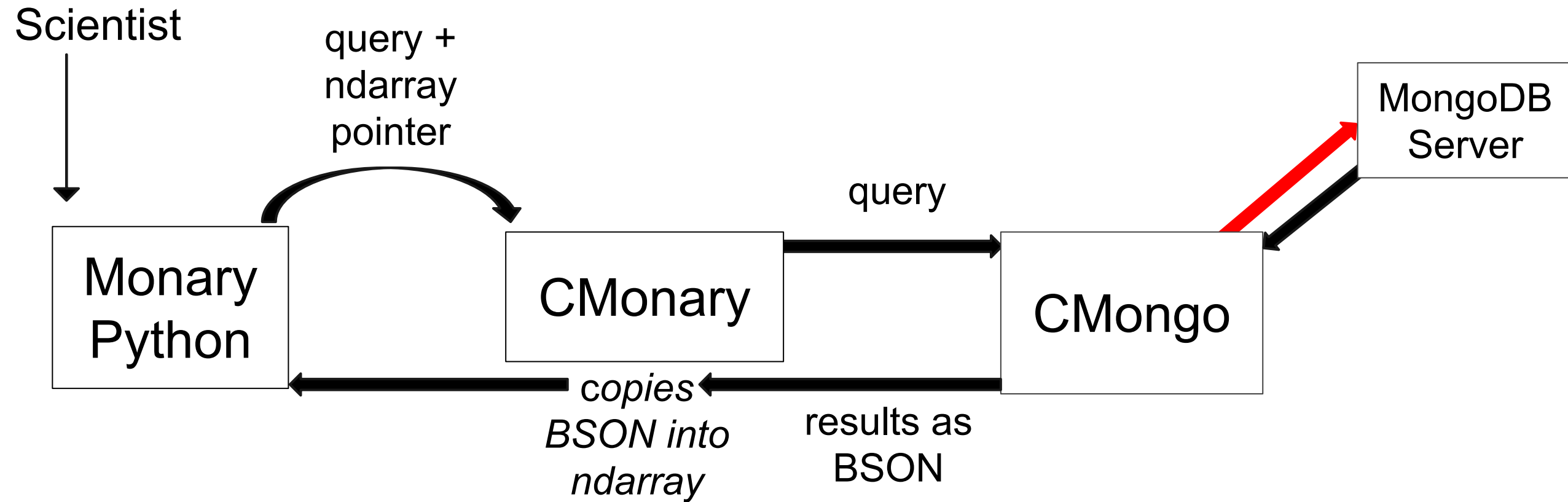
How Does Monary Work?



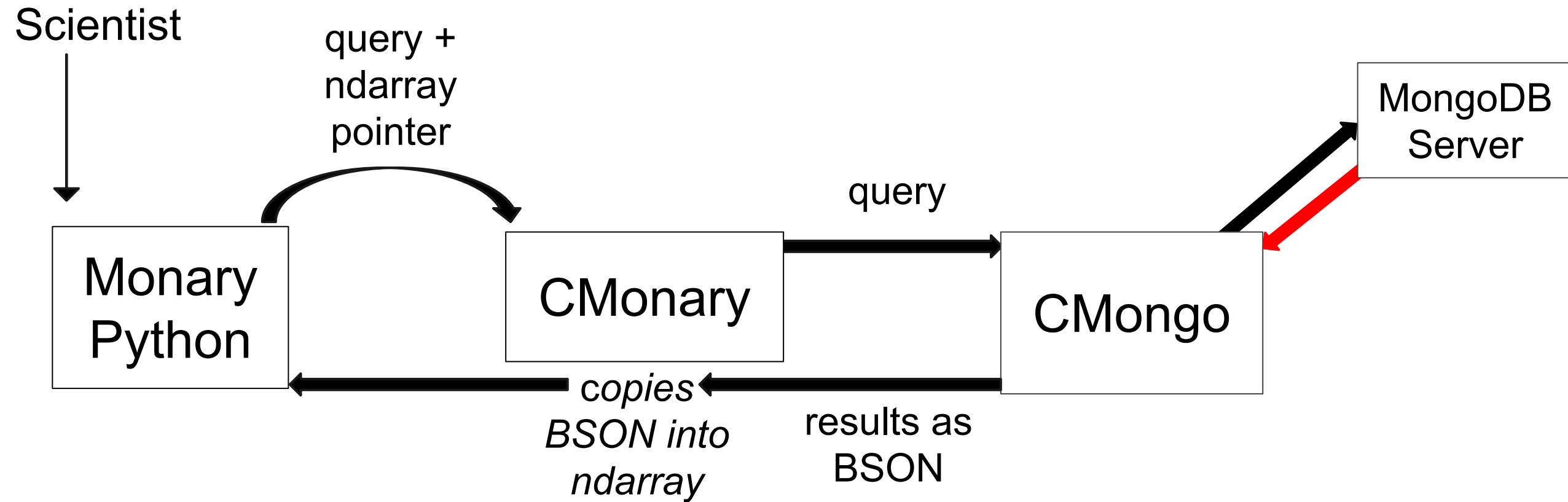
How Does Monary Work?



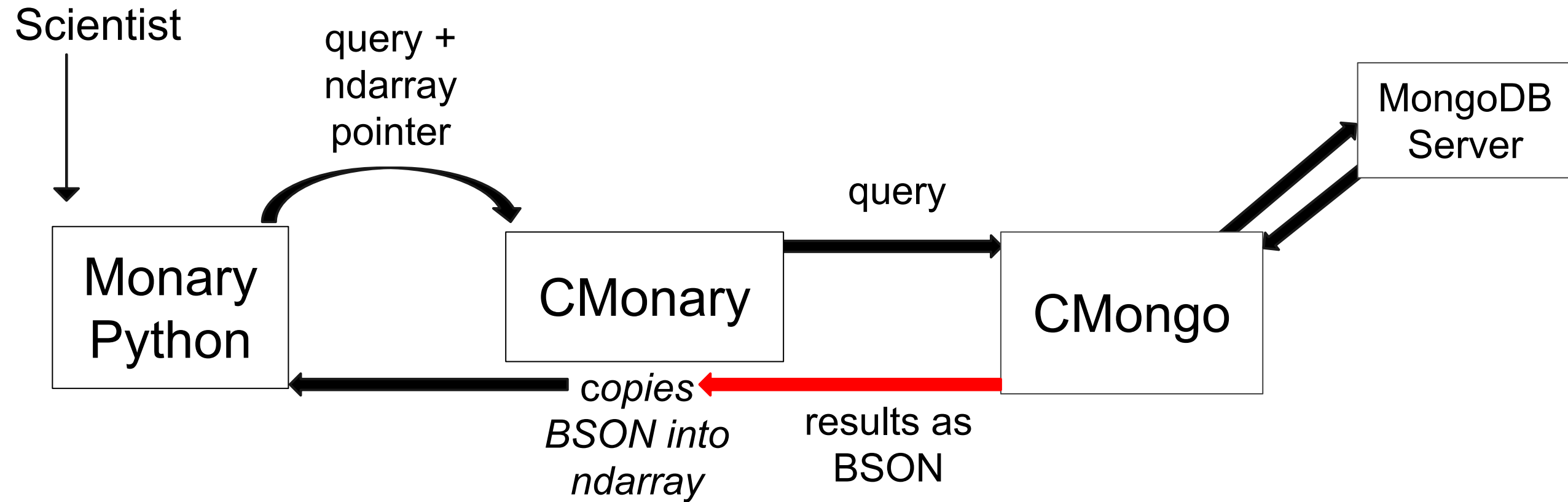
How Does Monary Work?



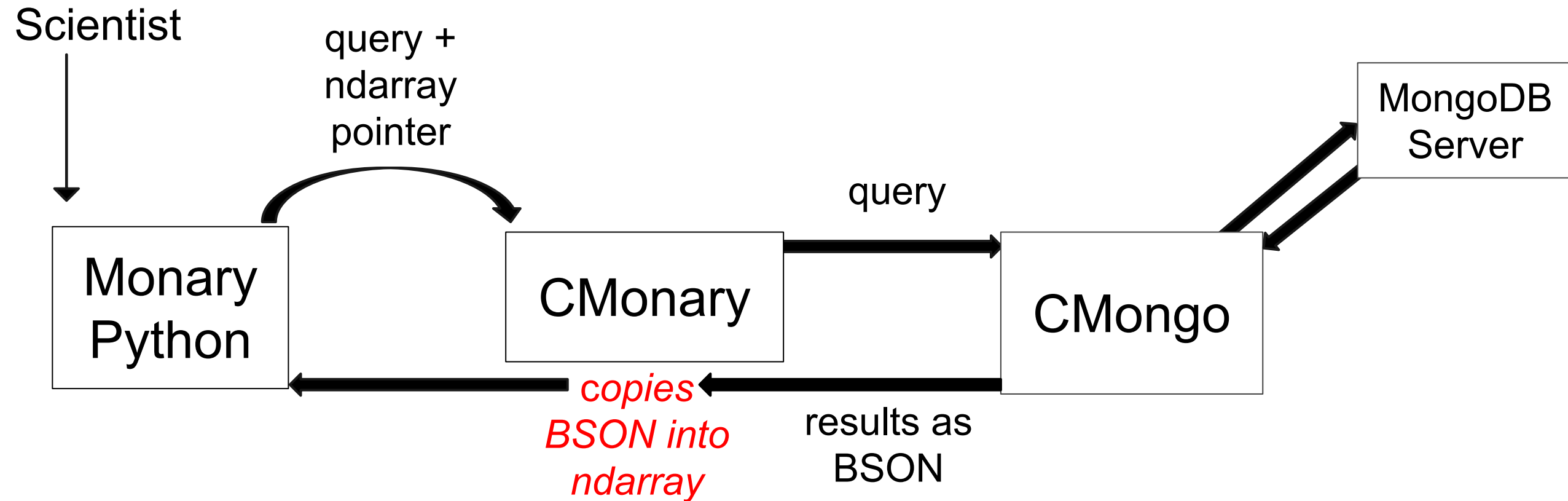
How Does Monary Work?



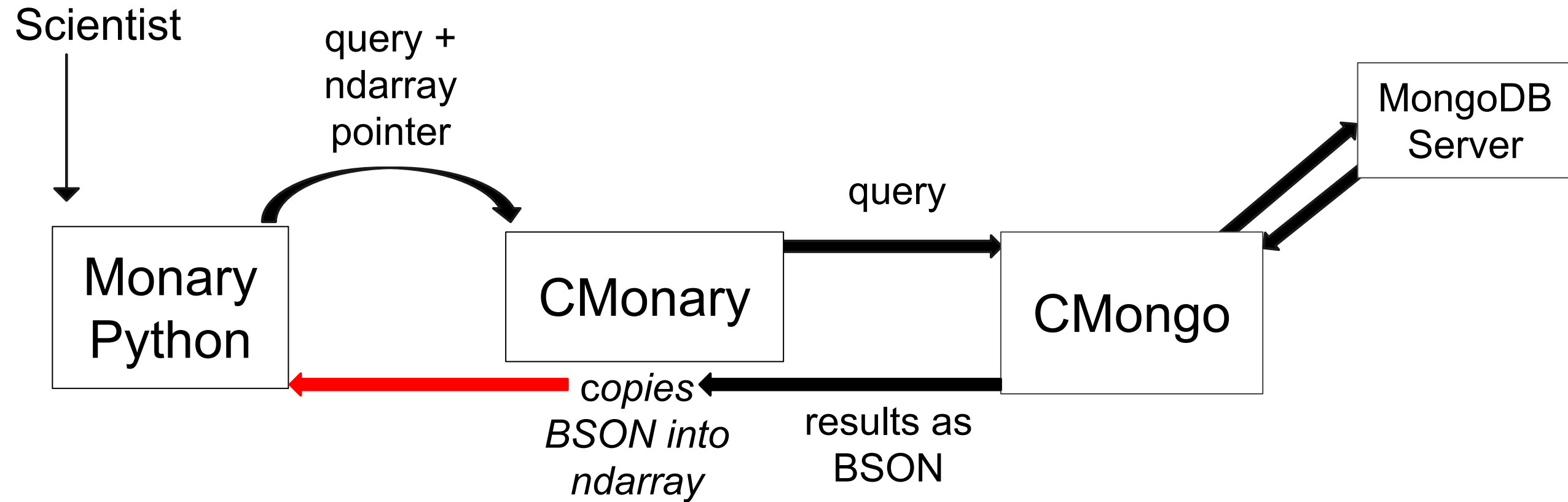
How Does Monary Work?



How Does Monary Work?



How Does Monary Work?



Monary Benefits

We can now combine the computational power of NumPy with the simplicity of MongoDB!

Demo

- NYC Taxi Data 2013 (Chris Whong)
- ~168 million rides
- What to do with it?



Collections

- “**pickups**”: set of trips that start in Times Square
- “**drops**”: set of trips that end in Times Square
- “**both**”: set of trips that start and end in the 5-block radius around times square (yes, there were 140,829)

Example: Query

Filtering by Neighborhood

- How can I further simplify the data?
 - Primarily interested in location points
- PediaCities' NYC neighborhoods GeoJSON
<http://nyc.pediacities.com/nycpedia>
- MongoDB's Geo Queries

My Query

```
with Monary() as m:
    m.count("taxi",
            "drop",
            {pickup_loc:
              $geoWithin: {
                neighborhoods["soho"]
              }
            })
```

Trips Starting in Times Square

Latitude

40.9

40.8

40.7

40.6

40.5

-74.3

-74.2

-74.1

-74.0

-73.9

-73.8

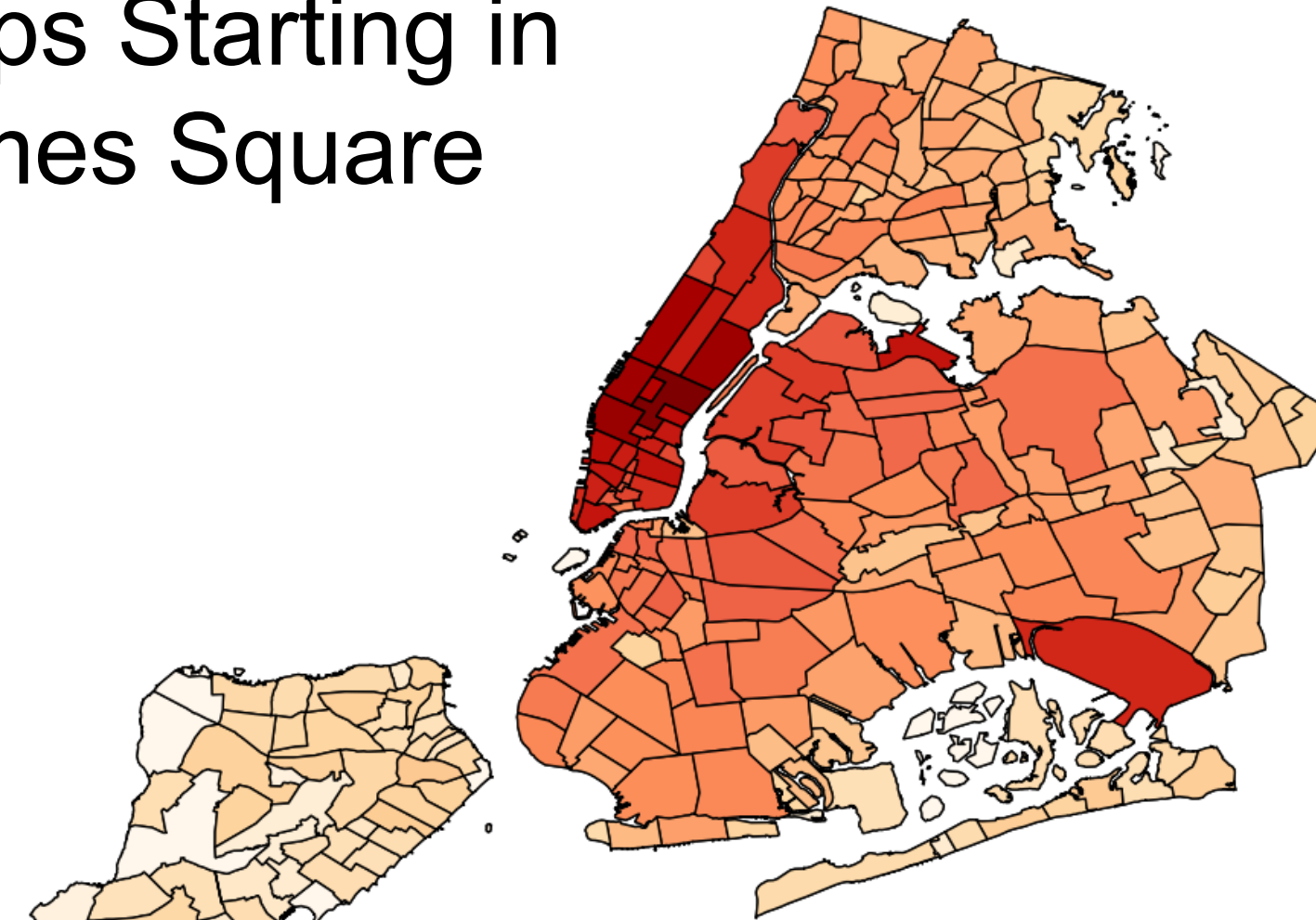
-73.7

-73.6

Longitude

Max: 1129284 (Midtown)

Min: 0 (about 80 places)



Trips Ending in Times Square

Latitude

40.9
40.8
40.7
40.6
40.5

-74.3

-74.2

-74.1

-74.0

-73.9

-73.8

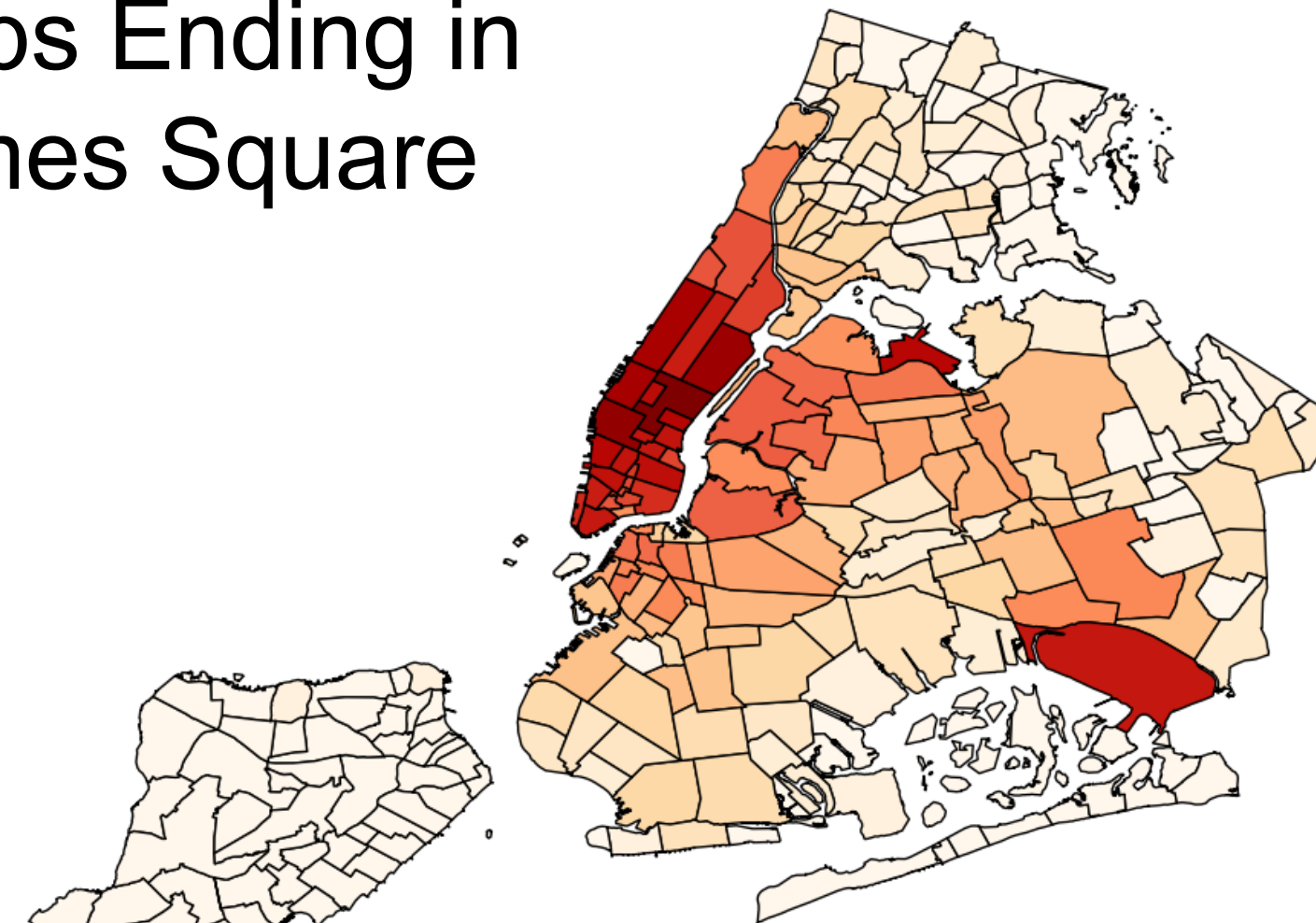
-73.7

-73.6

Longitude

Max: 1055572 (Midtown)

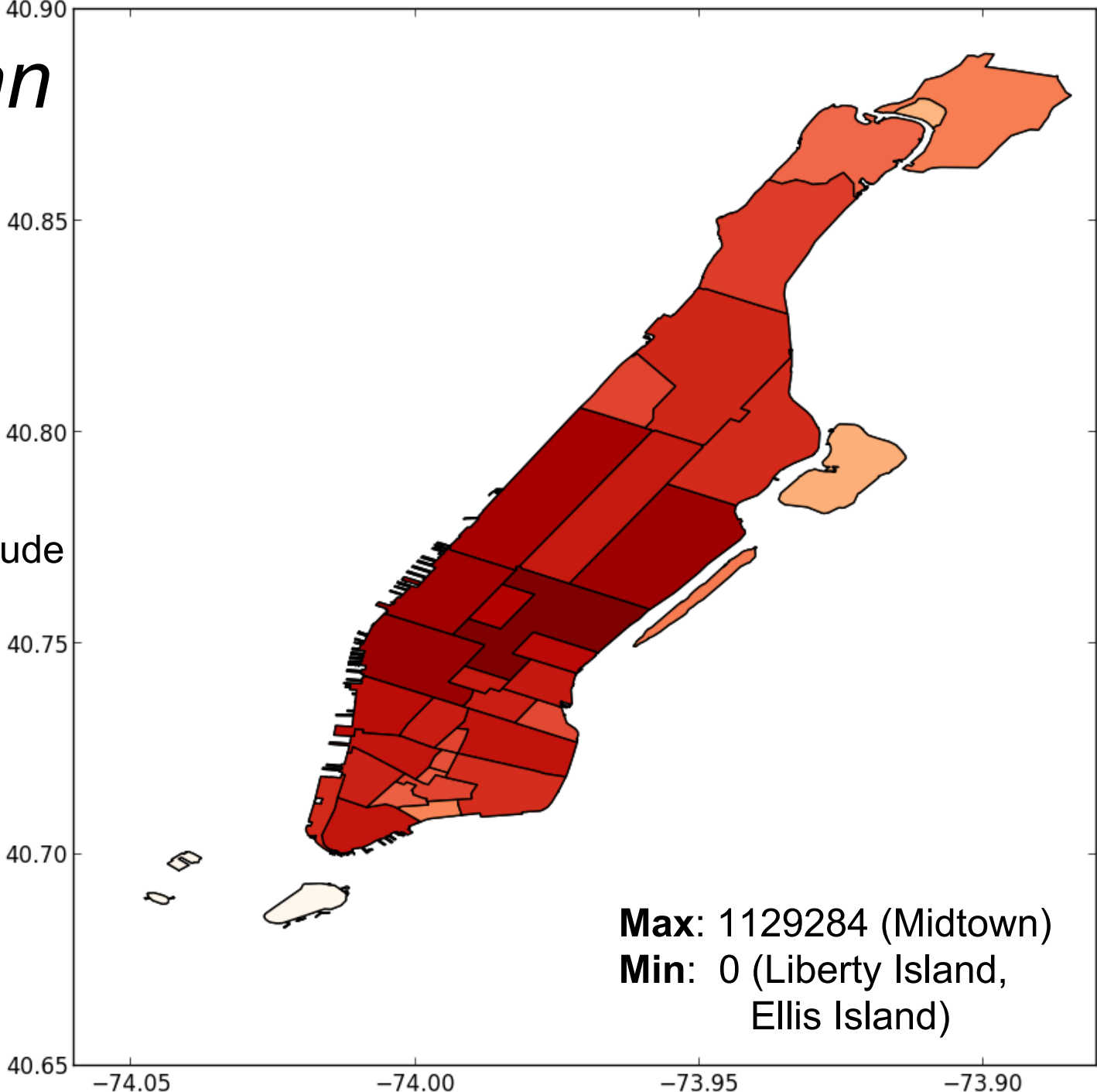
Min: 0 (about 40 places)



Manhattan

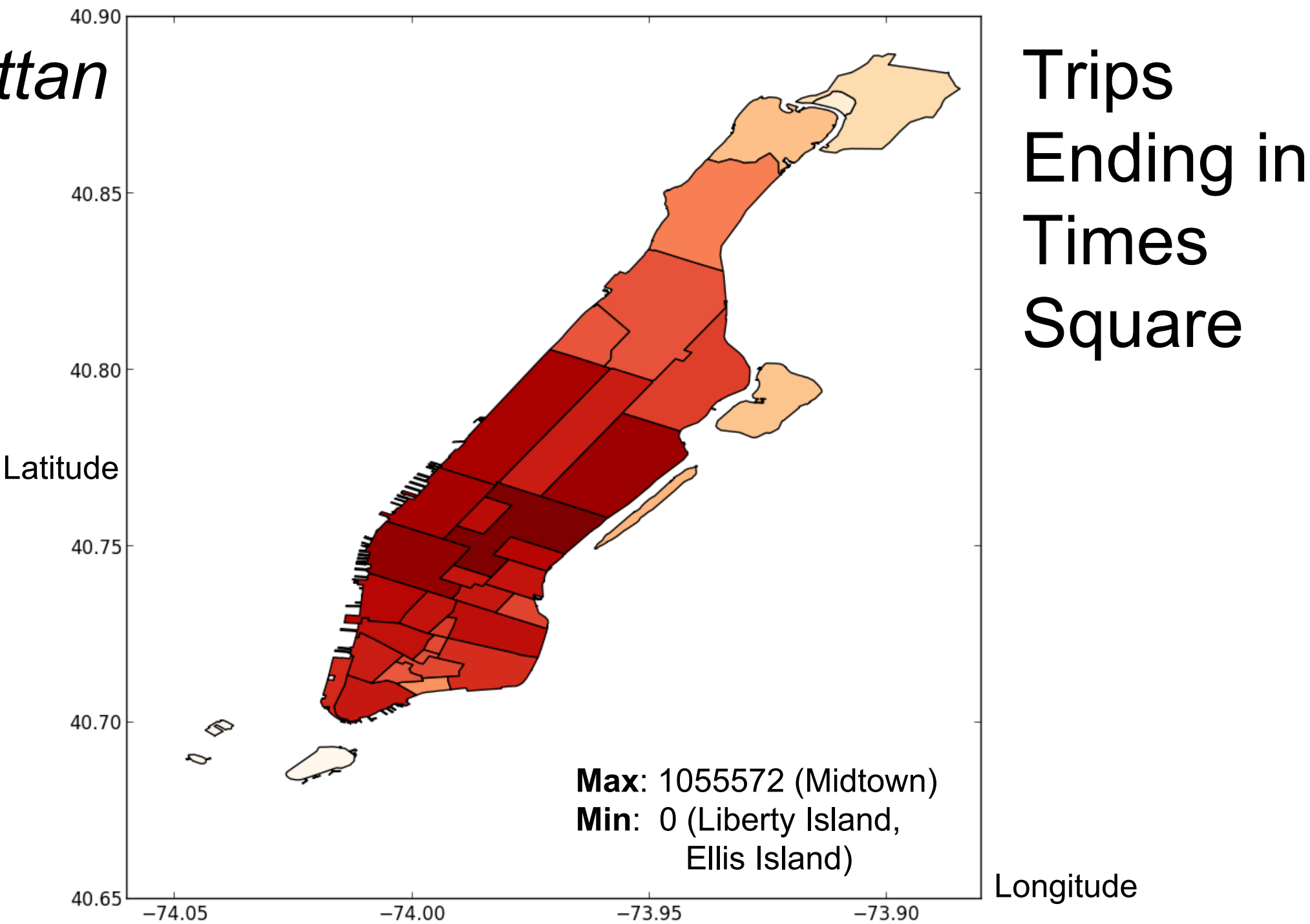
Trips
Starting in
Times
Square

Latitude



Longitude

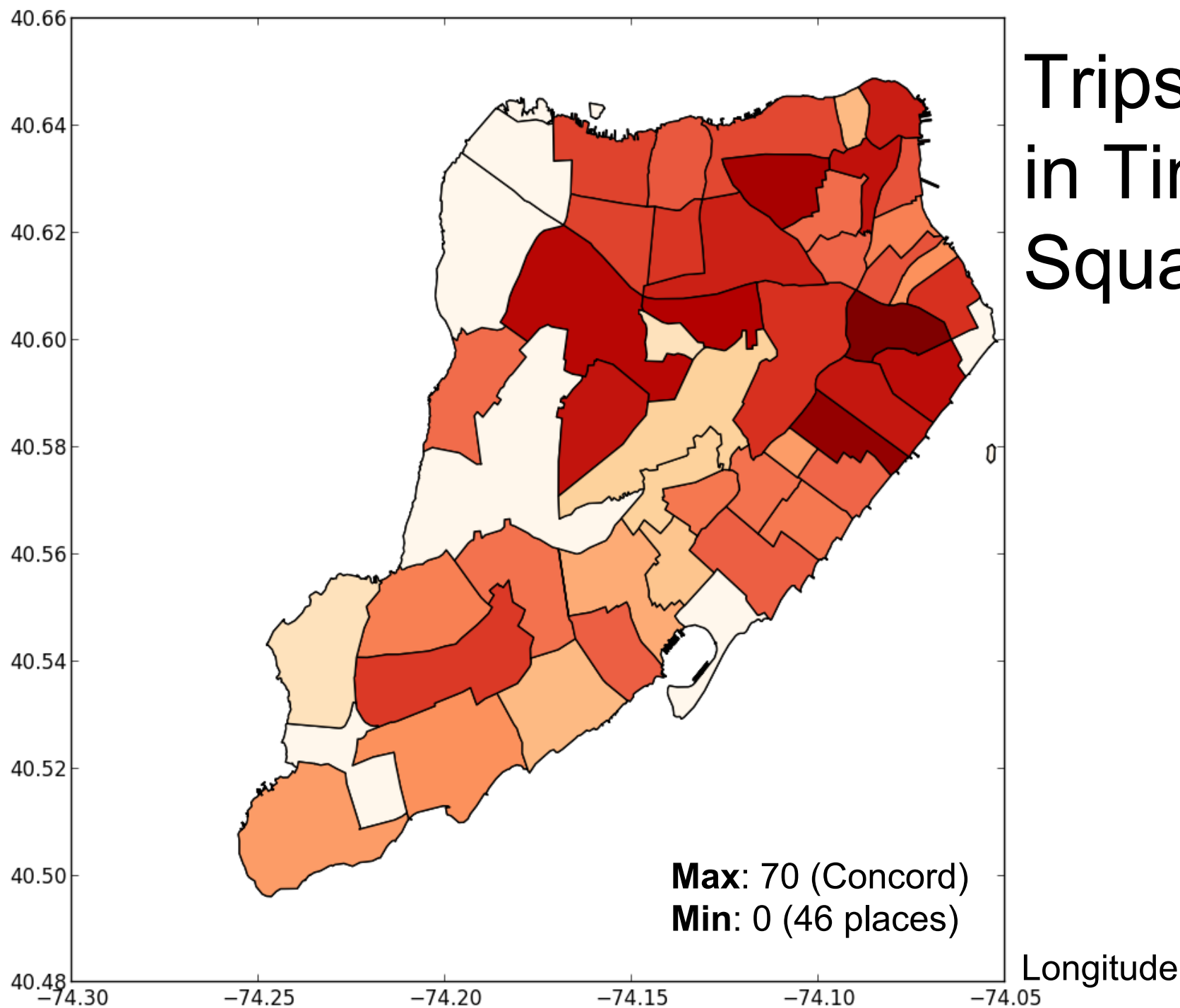
Manhattan



*Staten
Island*

Trips Starting
in Times
Square

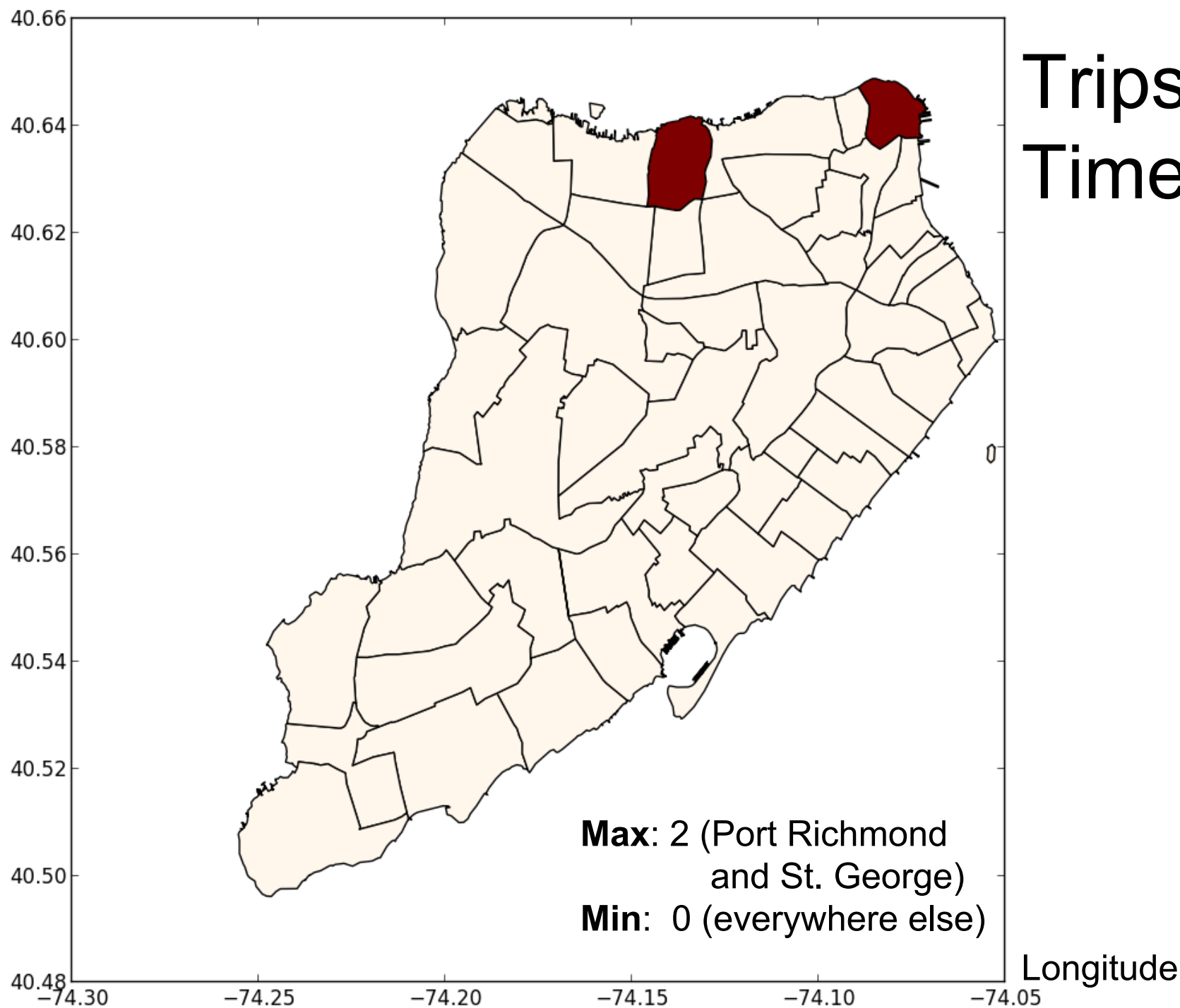
Latitude



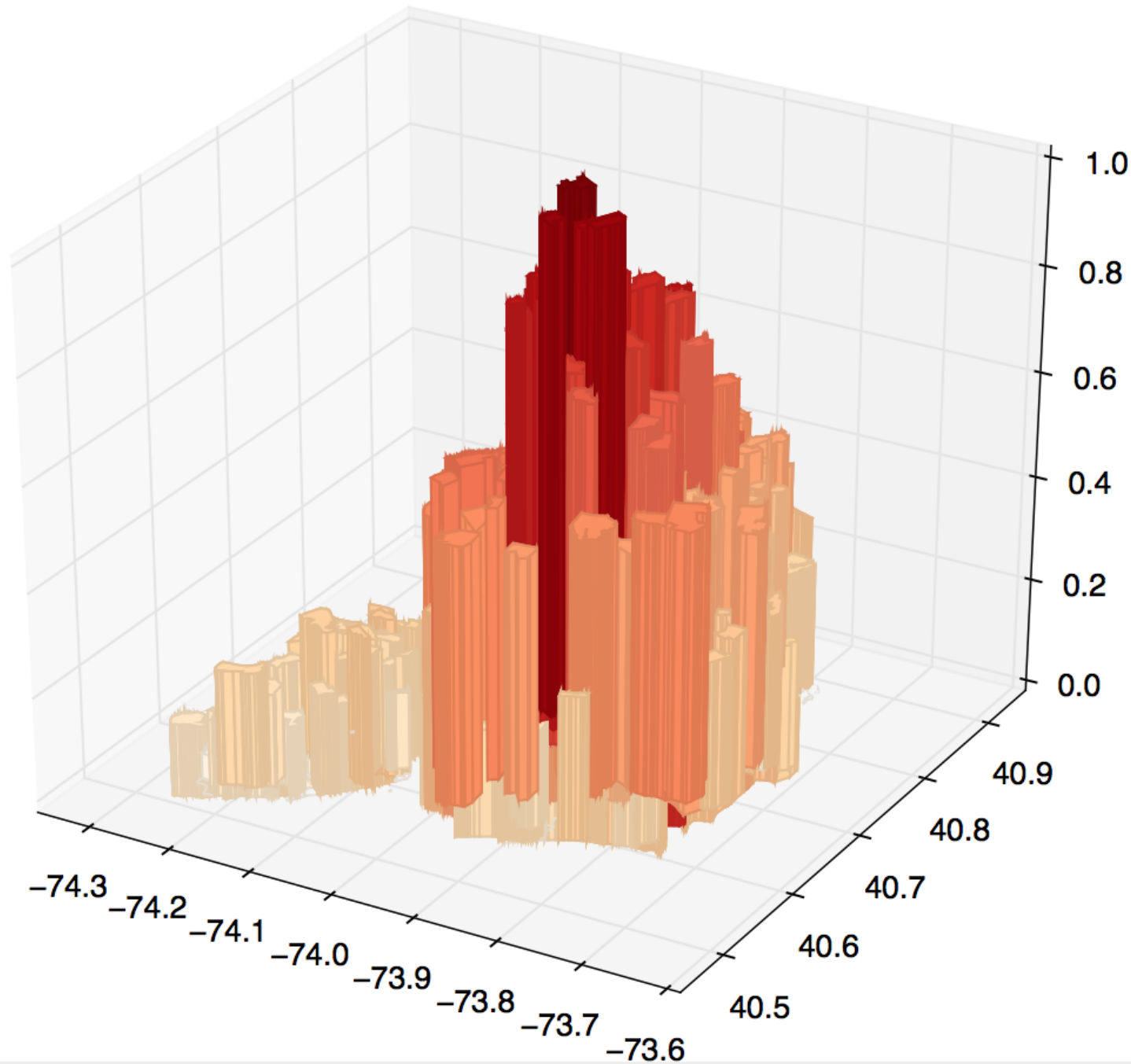
Staten Island

Trips Ending in Times Square

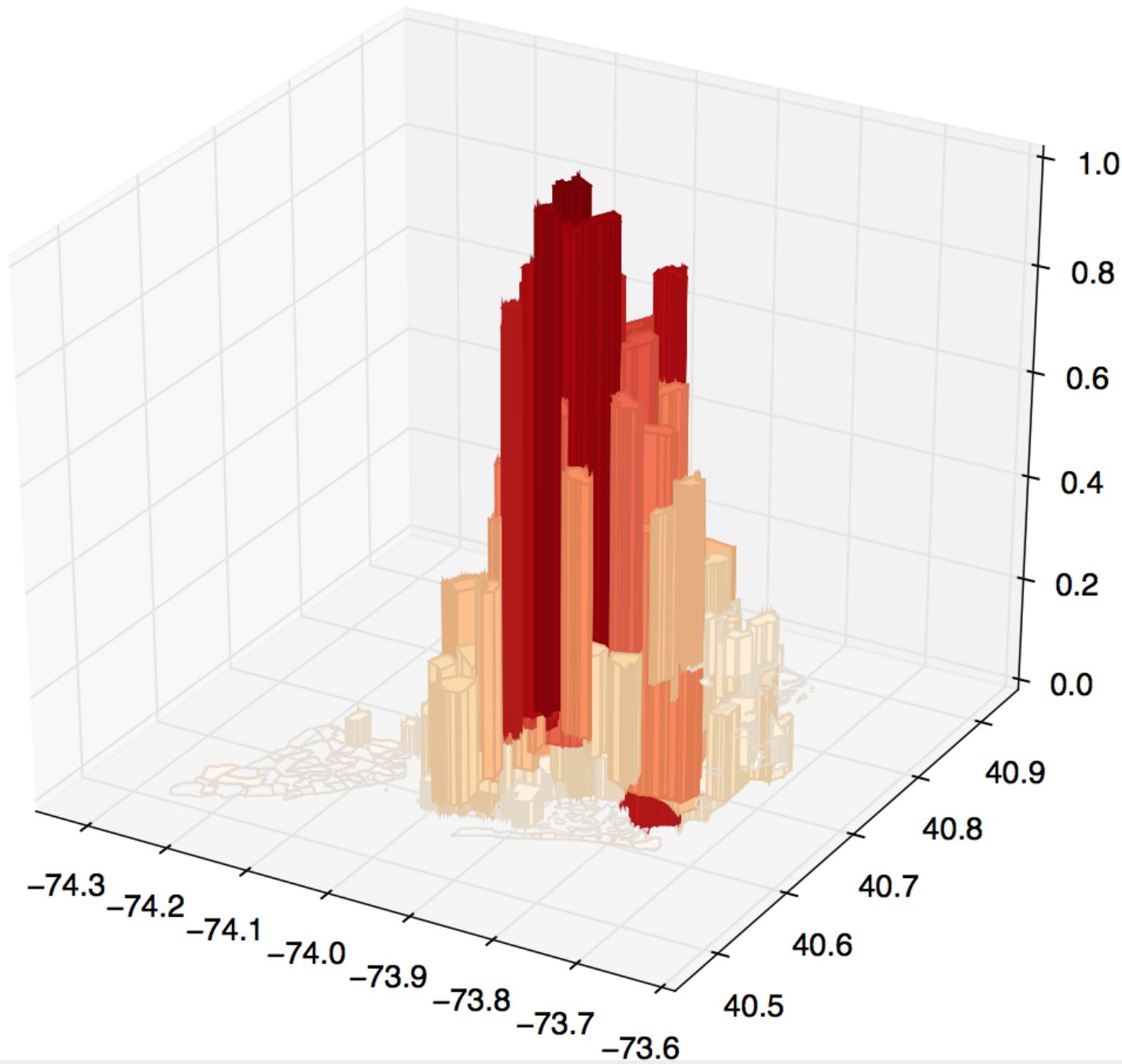
Latitude



Trips Starting in Times Square



Trips Ending in Times Square



Example: Aggregation

Aggregating Data by Date

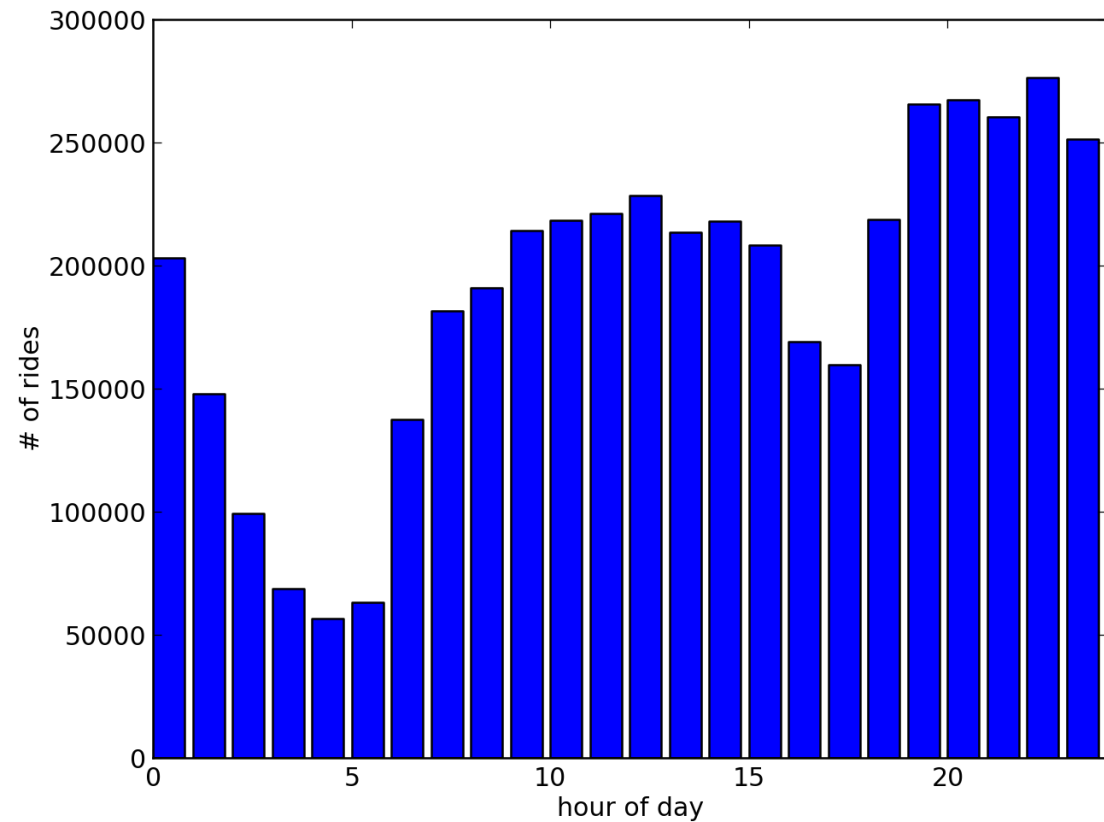
- MongoDB aggregation pipeline sets up various filters to run the data through
- \$group, \$count, \$sort
- Date Operators: \$hour, \$dayOfWeek, \$dayOfMonth, \$dayOfYear

Aggregation

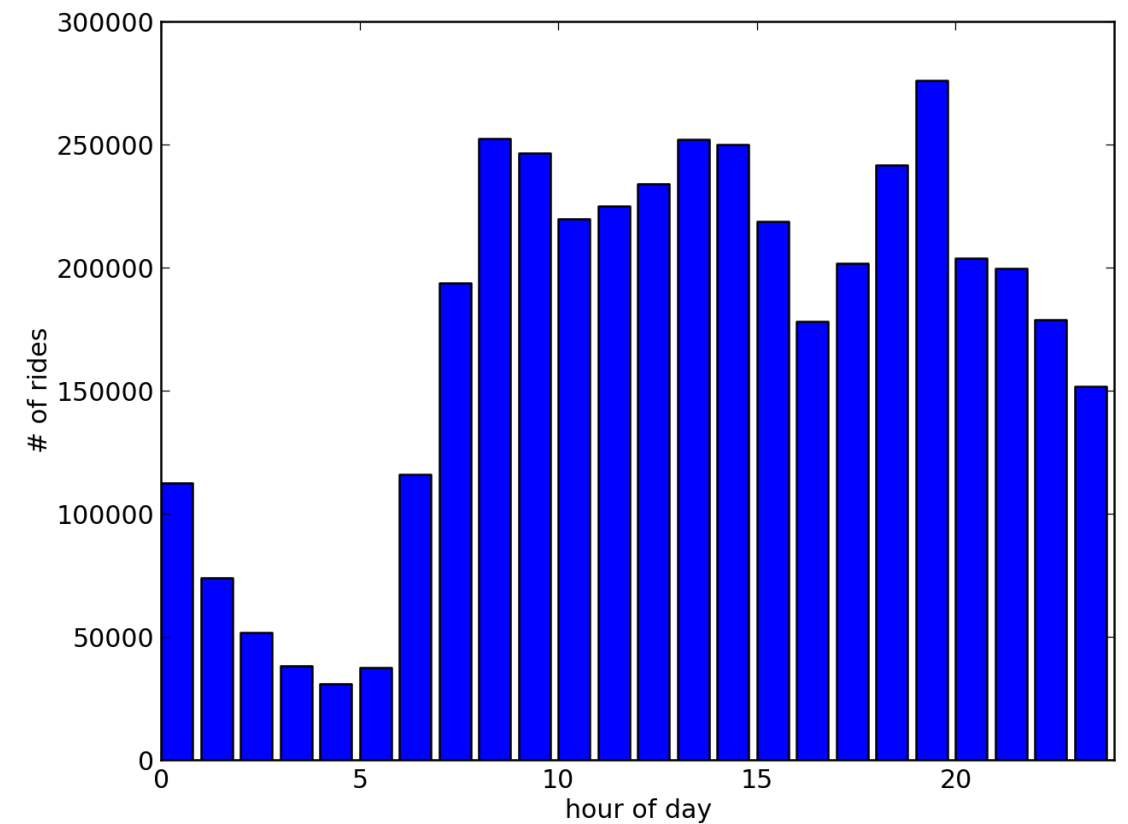
```
db.both.aggregate(  
  [{ $group: { _id:  
    { hourOfRide:  
      { $hour: "$pickup_time" } },  
    count: { $sum: 1 }  
  } } ] )
```

\$hour

Starting in Times Square

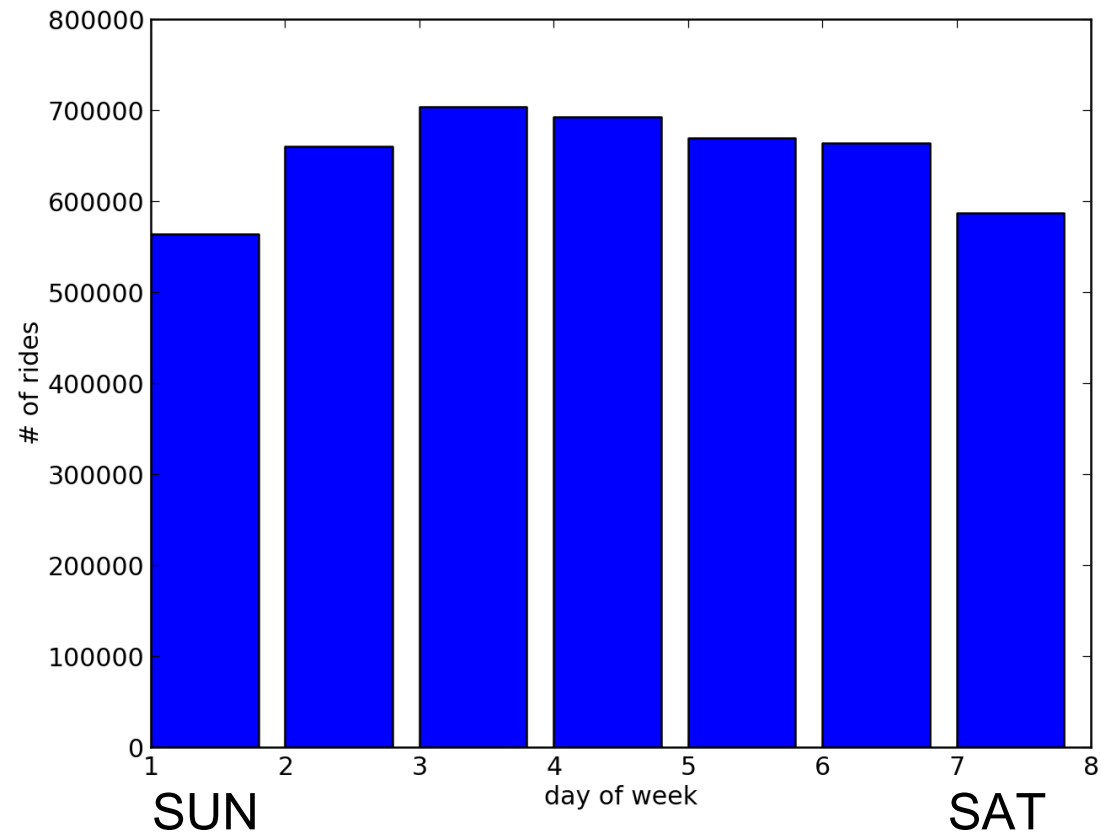


Ending in Times Square

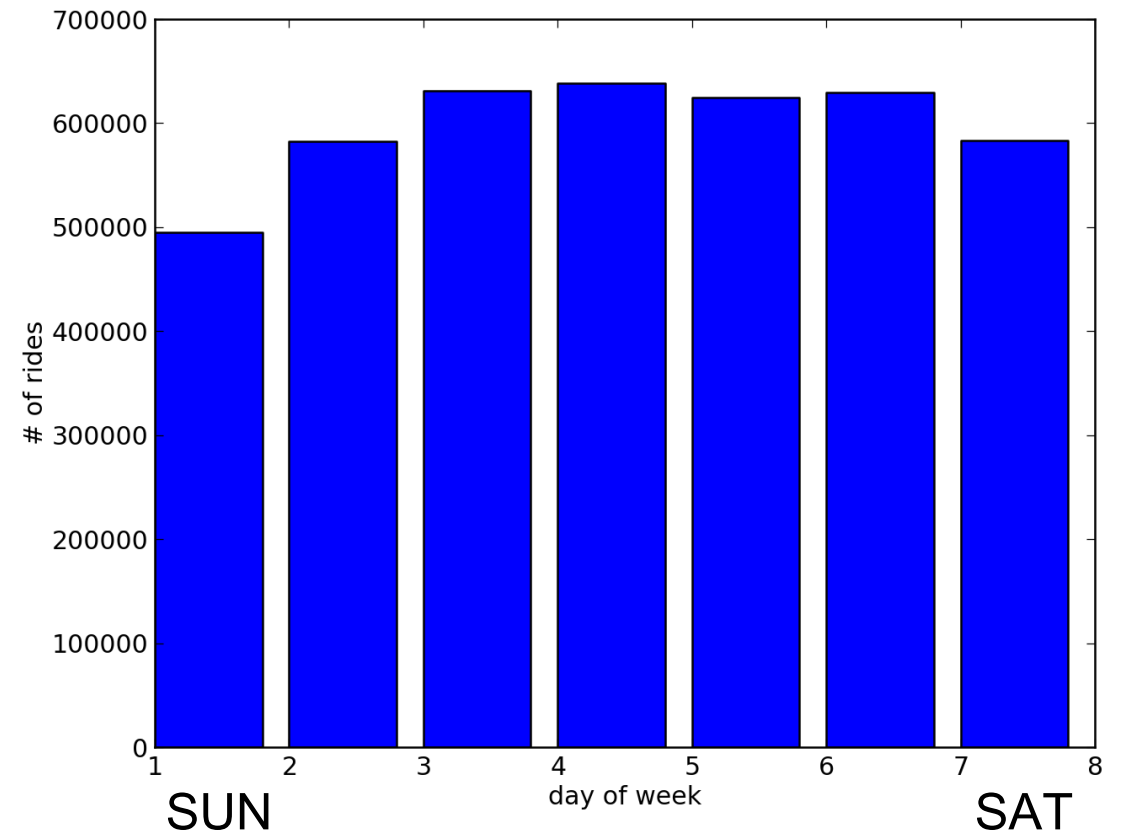


\$dayOfWeek

Starting in Times Square

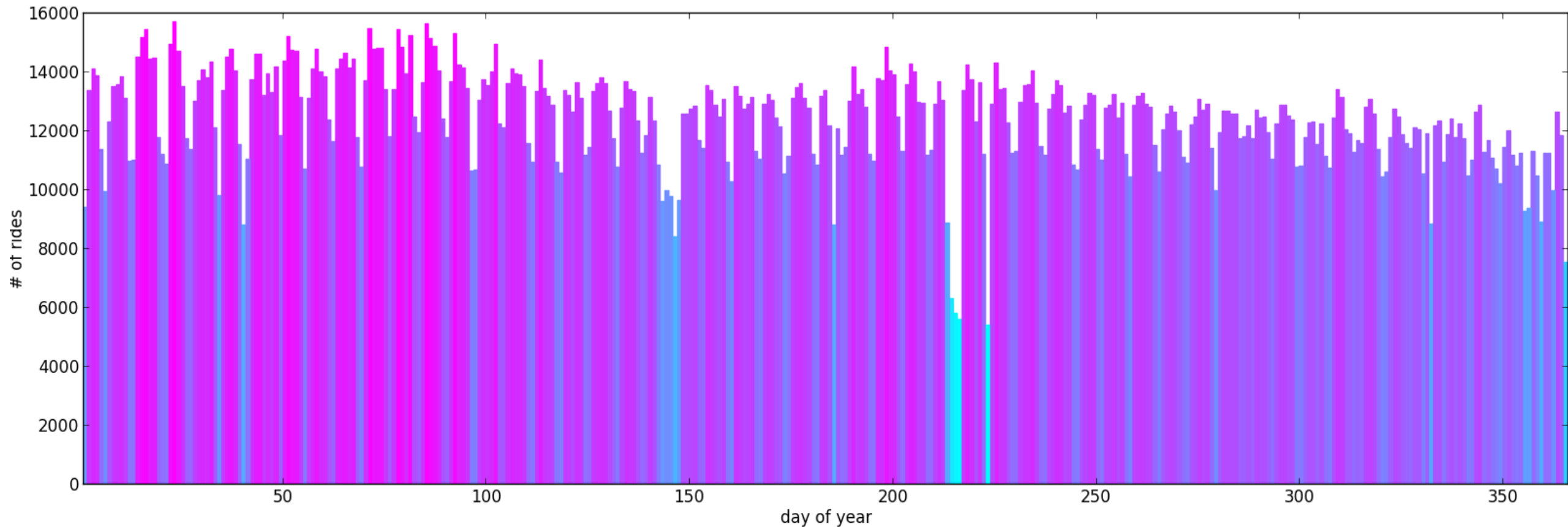


Ending in Times Square



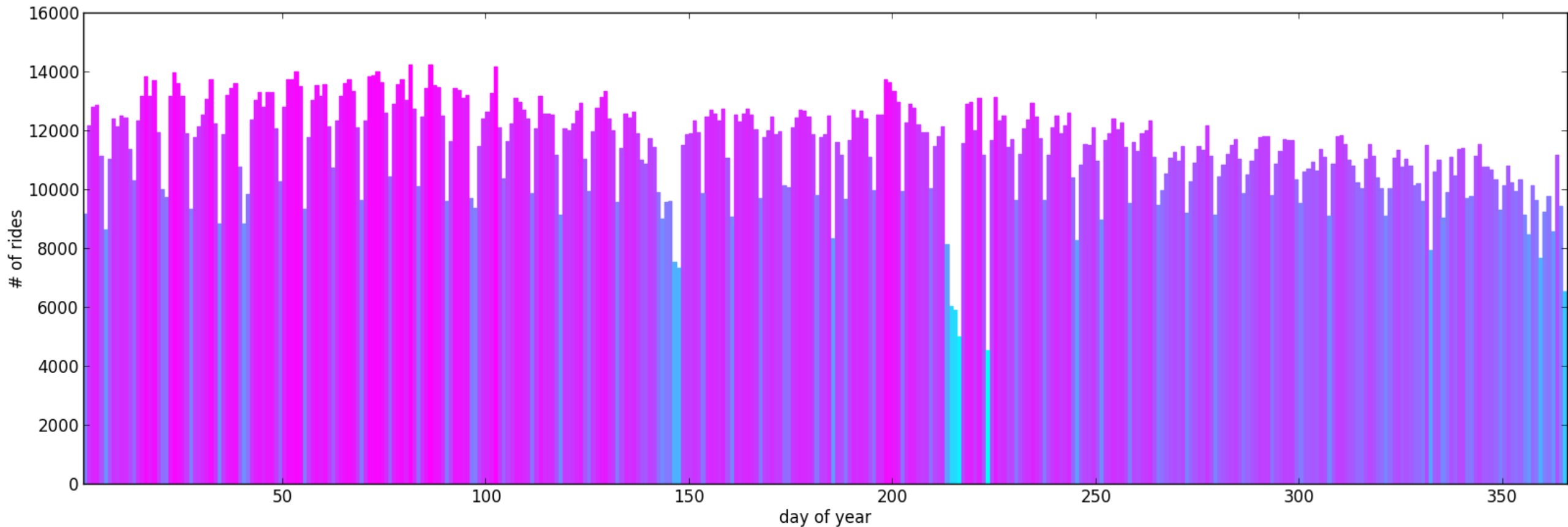
\$dayOfYear

Starting in Times Square



\$dayOfYear

Ending in Times Square



Future of Monary

Very exciting time for Monary!

- Matt + Kyle: inserts, aggregation
- Me: SSL, error-handling, multidimensional array extraction, DB commands, documentation, packaging
- TBD: Optimized search, logging, upserts, new C bindings, type-safe mode, and more!

Future Monary Integration

- Currently, a lot of projects that use MongoDB connect with Pymongo
 - Blaze!
- There are a lot of projects that could use a connection with MongoDB
 - Pandas!
- Endless other use cases

HUGE Thanks

- My mentors at MongoDB
 - A. Jesse Jiryu Davis
 - Jason Carey
- Summer Interns
 - Matt Cotter
 - Kyle Suarez
- David Beach

A person dressed as Batman stands in the center of a busy city street at night, likely Times Square in New York City. The person is wearing a black Batman suit with a cowl and a cape that is spread wide to the sides. The background is filled with the bright, colorful lights and billboards of the city, including signs for "KING", "Hankook", "Foot Locker", and "Goodbye Ch". The street is wet, reflecting the lights, and there are other people walking in the background.

*Thank you
OpenDataSciCon!*

Contact

anna.herlihy@mongodb.com

Source: bitbucket.org/djcbeach/monary

PyPI: pypi.python.org/pypi/monary

Documentation: monary.readthedocs.org

This Talk: github.com/aherlihy/monary-talk