

Our Search has an Elastic Heart

Terri Chu

About Me



- Florida
- Georgia Tech
- Quality Assurance - Performance Testing
- Iron Yard
- Back End Developer

Why Elastic Search?



What We'll Cover Today

- Elastic Search Lingo
- Setup Locally for use in Rails
- Creating Index and Documents
- Initial Data Load
- SQL —> Elastic Search
- Recap + Tips

Lingo

Setup

Create

**Data
Load**

**SQL
to ES**

Recap

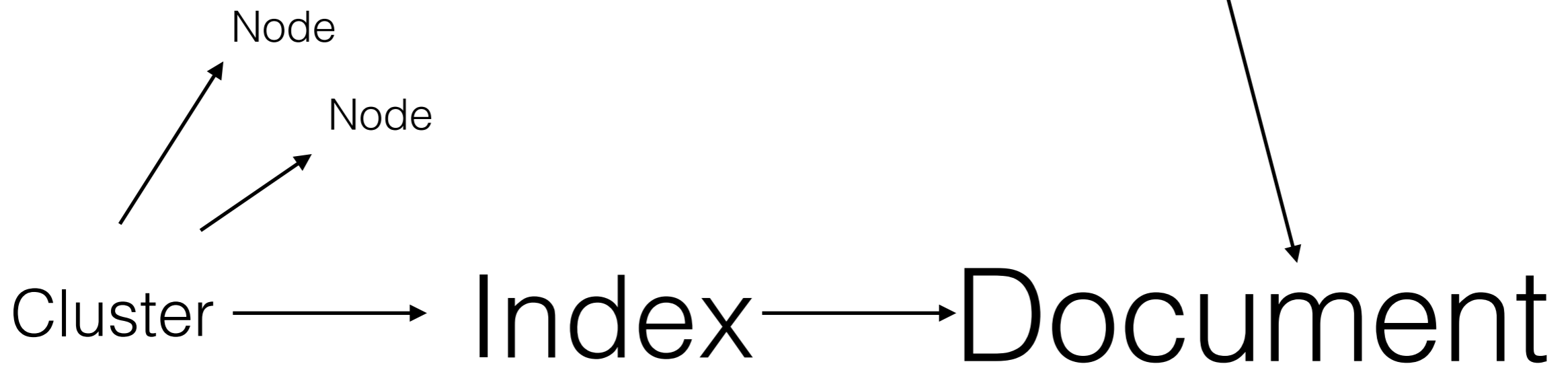
Lingo





elastic

This is the good stuff!



Setup

Lingo

Setup

Create

**Data
Load**

**SQL
to ES**

Recap

Setup Local

- Requirements? java \geq 1.8, homebrew
- Recommended Version: Elastic Search 5.3 or 5.4

Install

```
→ ~ brew install elasticsearch
```

Run Elastic Search

```
→ ~ elasticsearch
[2017-06-15T20:16:55,726][INFO ][o.e.n.Node               ] [] initializing ...
[2017-06-15T20:16:55,915][INFO ][o.e.e.NodeEnvironment   ] [ijXaa3D] using [1] data paths, mounts [[/ (C/
[2017-06-15T20:16:55,916][INFO ][o.e.e.NodeEnvironment   ] [ijXaa3D] heap size [1.9gb], compressed ordin
[2017-06-15T20:16:55,918][INFO ][o.e.n.Node               ] node name [ijXaa3D] derived from node ID [ijX
[2017-06-15T20:16:55,919][INFO ][o.e.n.Node               ] version[5.4.0], pid[23000], build[780f8c4/201
```

Lingo

Setup

Create

Data
Load

SQL
to ES

Recap

Setup in Rails

- Where? Gemfile

```
# pagination gems must be added before the Elasticsearch gems in your Gemfile
gem 'elasticsearch-model'
gem 'elasticsearch-rails'
# required for AWS authentication
gem 'faraday_middleware-aws-signers-v4'
```

Lingo

Setup

Create

Data
Load

SQL
to ES

Recap

Rails configuration

- Where? config/initializers/elastic_search.rb

```
config = {  
  host: ENV['elastic_search_url'], # http://localhost:9200  
  transport_options: {  
    request: { timeout: 15 },  
    headers: { content_type: 'application/json' }  
  },  
  reload_on_failure: true, # reload connections on failure  
  reload_connections: true # retrieve and use the information from the Nodes Info API on every 10,000th request  
}  
  
config[:log] = true if Rails.env.development? && !(defined?(Rake) && Rake.application.name)
```

Lingo

Setup

Create

Data
Load

SQL
to ES

Recap

AWS Considerations

Lingo

Setup

Create

**Data
Load**

**SQL
to ES**

Recap

Full Config for AWS

```
require 'faraday_middleware/aws_signers_v4'

config = {
  host: ENV['elastic_search_url'], # http://localhost:9200
  transport_options: {
    request: { timeout: 15 },
    headers: { content_type: 'application/json' }
  },
  reload_on_failure: true, # reload connections on failure
  reload_connections: true # retrieve and use the information from the Nodes Info API on every 10,000th request
}

config[:log] = true if Rails.env.development? && !(defined?(Rake) && Rake.application.name)

Elasticsearch::Model.client = Elasticsearch::Client.new(config) do |f|
  if Rails.env.production?
    f.request :aws_signers_v4,
              credentials: Aws::Credentials.new(ENV['aws_access_key_id'], ENV['aws_secret_access_key']),
              service_name: 'es',
              region: 'us-east-1'
  end
end
```

Lingo

Setup

Create

Data
Load

SQL
to ES

Recap

Create Index & Documents

Lingo

Setup

Create

**Data
Load**

**SQL
to ES**

Recap

Model to Index

- Where? Top of the model file

```
class Person < ActiveRecord::Base
  include Elasticsearch::Model
  include Elasticsearch::Callbacks
```

- *Custom Callbacks*
- *Querying in Rails*

```
Person.search(query).records
```

Lingo

Setup

Create

Data
Load

SQL
to ES

Recap

Index & Document Name

- Where? Public area of model
 - Index Name - Use Environment Name
 - Document Type - Use Model Name

```
class Person < ActiveRecord::Base
  include Elasticsearch::Model
  include Elasticsearch::Callbacks

  index_name "#{Rails.env}_hirewire_index" # Ex. production_hirewire_index
  document_type 'person'
end
```

Lingo

Setup

Create

Data
Load

SQL
to ES

Recap

Index Settings & Mappings

- Where? Public area of model
- What? Fields to index (part of model or calculated)

```
index_name "#{Rails.env}_hirewire_index" # Ex. production_hirewire_index
document_type 'person'

settings index: { number_of_shards: 1, number_of_replicas: 1 } do
  mappings dynamic: 'true' do
    indexes :discoverable, type: 'boolean'
    indexes :distance_willing_to_travel, type: 'integer'
    indexes :location, type: 'geo_point'
    indexes :interested_positions, type: 'text'
    indexes :user, type: 'nested' do
      indexes :last_sign_in_at, type: 'date', format: 'strict_date_time'
    end
  end
end
end
```

Lingo

Setup

Create

Data
Load

SQL
to ES

Recap

Document Data

- Where? Any place in the public model
- What? JSON object containing all data

```
def as_indexed_json(options = {})
  json = as_json( only: [:id, :distance_willing_to_travel, :discoverable],
                 include: {
                   user: { only: [:last_sign_in_at] }
                 })
  location = { lat: latitude, lng: longitude }
  interested_positions = interested_in_positions.pluck('positions.name').join(', ')
  additional_attributes = { location: location , interested_positions: interested_positions}
  json.merge(additional_attributes)
end
end
```

Lingo

Setup

Create

Data
Load

SQL
to ES

Recap

as_indexed_json output

```
{
  id: 1,
  discoverable: true,
  distance_willing_to_travel: 10,
  interested_positions: "Server, Bartender, Programmer"
  location: {
    lat: 34.3423423,
    lng: -84.3432432
  }
  user: {
    last_sign_in_at: "2017-06-20T13:30:30Z"
  }
}
```

Lingo

Setup

Create

Data
Load

SQL
to ES

Recap

Initial Data Load

Lingo

Setup

Create

Data
Load

SQL
to ES

Recap

Rake Task

```
client = Elasticsearch::Client.new(host: ENV['elastic_search_url']) do |f|
  if Rails.env.production?
    f.request :aws_signers_v4,
              credentials: Aws::Credentials.new(ENV['aws_access_key_id'], ENV['aws_secret_access_key']),
              service_name: 'es',
              region: 'us-east-1'
  end
end

index_name = "#{Person.index_name}_#{SecureRandom.hex}"

client.indices.create index: index_name,
                     body: { settings: Person.settings.to_hash,
                           mappings: Person.mappings.to_hash }

client.indices.put_alias(index: index_name, name: Person.index_name)
persons = Person.all
persons.import
```

** May take a long time with large data sets*

Lingo

Setup

Create

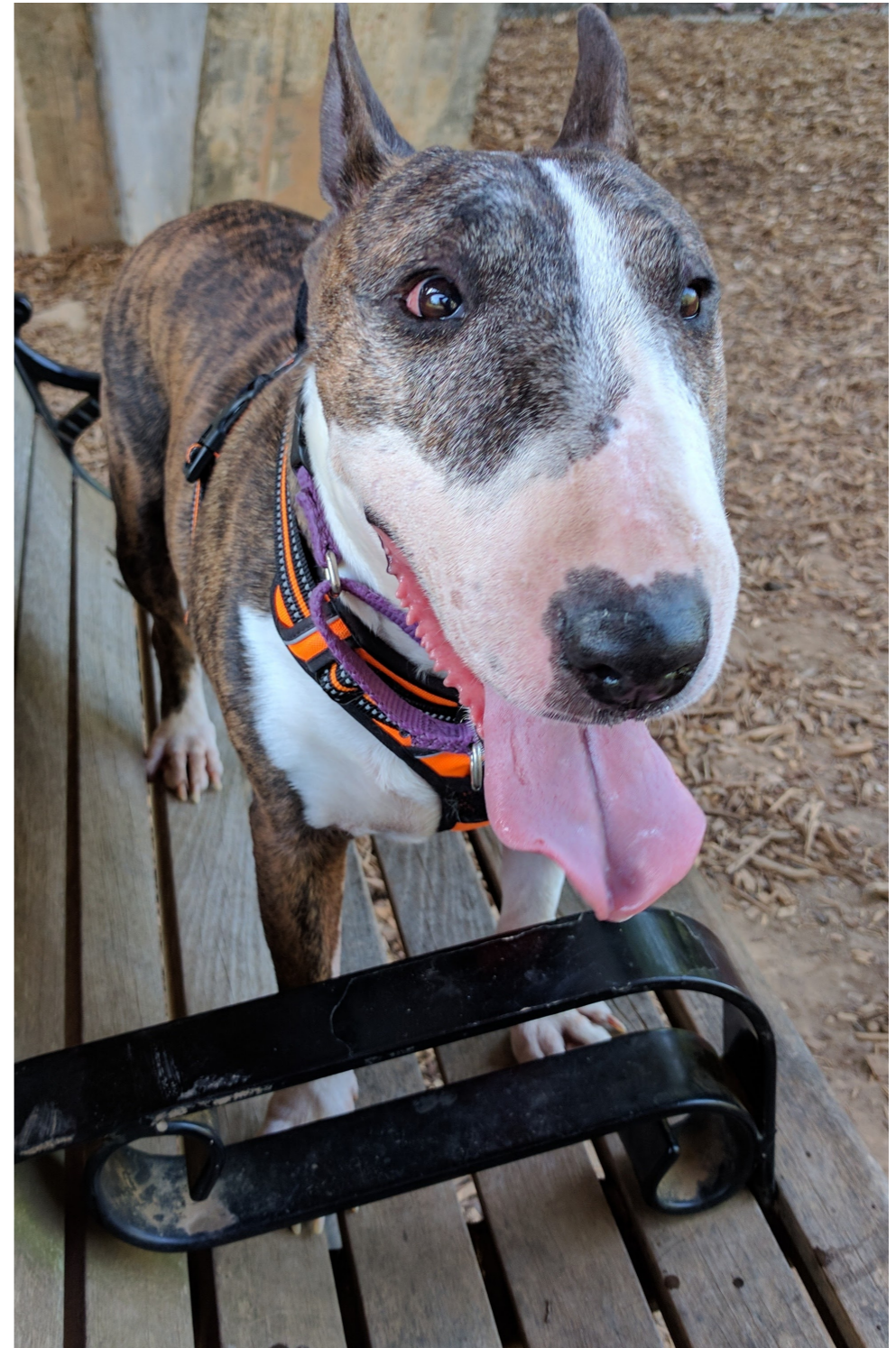
Data
Load

SQL
to ES

Recap

Onto the Fun!

Replacing SQL Queries
with Elastic Search



Lingo

Setup

Create

Data
Load

SQL
to ES

Recap

Term Search

SQL

```
SELECT * FROM persons JOIN interested_positions ON interested_positions.person_id = persons.id  
WHERE interested_positions.name LIKE '%Server%';
```

Elastic Search

```
def positions_interested_term_search(term)  
  {  
    "query": {  
      "term": { "positions_interested_in": term }  
    }  
  }  
end  
query = positions_interested_term_search('Server')  
Person.search(query).records
```

Lingo

Setup

Create

Data
Load

SQL
to ES

Recap

Range Search

SQL

```
# SELECT * FROM persons WHERE distance_willing_to_travel <= 25;
```

Elastic Search

```
def distance_willing_to_travel_range_search(distance)
  {
    "query": {
      "range": {
        "distance_willing_to_travel": {
          "lte": distance
        }
      }
    }
  }
end
query = distance_willing_to_travel_range_search(25)
Person.search(query).records
```

Lingo

Setup

Create

Data
Load

SQL
to ES

Recap

GeoDistance Search

SQL

```
Hirewire=# SELECT * FROM persons WHERE ASIN( SQRT(SIN(RADIANS(<LAT> - persons.latitude) / 2) ^ 2 + SIN(RADIANS(<LNG> - persons.longitude) / 2) ^ 2 * COS(RADIANS(<LNG>) * COS(RADIANS(<LAT>)))) * 7926.3352 < 25;
```

Elastic Search

```
# NOTE: applied as a filter to a query
def geo_distance_filter(lat, lng, distance)
{
  "geo_distance": {
    "distance": "#{distance}mi", # can be km
    "distance_type": 'sloppy_arc',
    "location": {
      "lat": lat,
      "lon": lng
    }
  }
}
end
query = {
  "bool": {
    "must": { "match_all": {} },
    "filter": geo_distance_search(84.34233, -39.23234, 25)
  }
}
Person.search(query).records
```

Lingo

Setup

Create

Data
Load

SQL
to ES

Recap

Nested Search

SQL

```
Hirewire=# SELECT * FROM persons JOINS users ON users.id = persons.user_id WHERE DATE(users.last_sign_in_at) >= DATE(NOW() - interval '21 days');
```

Elastic Search

```
def user_signed_in_range_search(days)
  {
    "query": {
      "nested": {
        "path": 'users',
        "query": {
          "range": {
            "user.last_sign_in_at": {
              "gte": "now-#{days}d"
            }
          }
        }
      }
    }
  }
end
query = user_signed_in_range_search(21)
Person.search(query).records
```

Lingo

Setup

Create

Data
Load

SQL
to ES

Recap

Converting SQL to ES

Bool Query as the base

```
query = {  
  "query": {  
    "bool": {  
      "must": [], # AND  
      "must_not": [], # NOT  
      "should": [], # OR  
      "minimum_should_match": 1, # applies to should  
      "filter": [] # geo-distance  
    }  
  }  
}
```

Lingo

Setup

Create

Data
Load

SQL
to ES

Recap

Tying it all Together

```
query = {
  "query": {
    "bool": {
      "must": [], # AND
      "must_not": [], # NOT
      "should": [], # OR
      "minimum_should_match": 1, # applies to should
      "filter": [] # geo-distance
    }
  }
}

query['query']['bool']['must'] << user_signed_in_range_search(21)
query['query']['bool']['must'] << distance_willing_to_travel_range_search(25)
query['query']['bool']['should'] << positions_interested_term_search('Developer')
query['query']['bool']['should'] << positions_interested_term_search('Server')
query['query']['bool']['filter'] << geo_distance_search(84.34233, -39.23234, 25)

Person.search(query).records
```

Lingo

Setup

Create

Data
Load

SQL
to ES

Recap

Recap + Tips

- Start with one Model
- Only index what you need
- Pick a unique Index name and use an alias
- Break your queries up



Refresh Data With No Down Time

- Why?
- How?
 - Rake task
 - Elastic Search Aliases ← *used in our Import*
 - Redis
- Article reference
<https://berislavbabic.com/refresh-your-elasticsearch-index-with-zero-downtime/>

elasticsearch-head

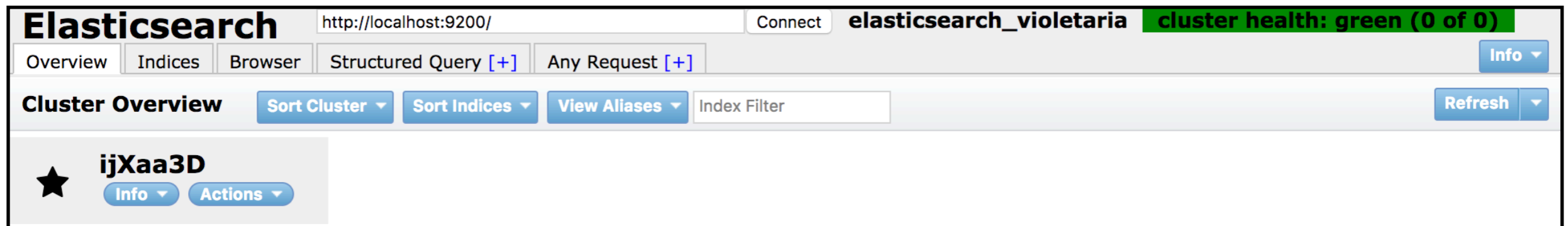
- Requirements? Docker for Mac
- install elasticsearch-head
<https://github.com/mobz/elasticsearch-head>

```
> docker run -p 9100:9100 mobz/elasticsearch-head:5
```

Configure ES

- Where? /usr/local/etc/elasticsearch/
elasticsearch.yml

```
http.cors.enabled: true
http.cors.allow-origin: /https?:\/\/localhost(:[0-9]+)?/
```



The screenshot shows the Elasticsearch Kibana interface. At the top, the title is "Elasticsearch" and the URL is "http://localhost:9200/". There is a "Connect" button and the cluster name "elasticsearch_violetaria" with a green status bar indicating "cluster health: green (0 of 0)". Below the title bar, there are navigation tabs: "Overview", "Indices", "Browser", "Structured Query [+]", and "Any Request [+]", along with an "Info" dropdown. The main content area is titled "Cluster Overview" and includes buttons for "Sort Cluster", "Sort Indices", "View Aliases", and an "Index Filter" input field. A "Refresh" button is also present. In the bottom left corner, there is a star icon and the name "ijXaa3D" with "Info" and "Actions" dropdowns.

* *elasticsearch-head does not really work well with AWS*

Thank you!

Terri Chu

@PerfTestGoddess

www.getlostthere.com

