

Elasticsearch: Accelerating the Django Admin

Presented by Kate Kligman

October 15, 2018 @ DjangoCon 2018



@KateKligman

www.katekligman.com

Our Platform



GROVE
collaborative

- Django 1.11
- Heroku
- AWS RDS Postgres
- 7036% 3-year growth

The Django Admin



WELCOME, [redacted]

[VIEW SITE](#) / [CHANGE PASSWORD](#) / [LOG OUT](#)



DASHBOARD

BOOKMARKS

APPLICATIONS ▾

ADMINISTRATION ▾



Home > [redacted] > Customers

Select customer to change

ADD CUSTOMER +



kligman

Search

2 results (Show all)

Action:



Go

0 of 2 selected



EMAIL

CARD ON FILE

ACTIVE

IS VIP

TRACKING LINK

FIRST NAME

LAST NAME



ka[redacted]om



www.grove.co

Kate

Kligman



k[redacted]e.co



None

Kate

Kligman

2 customers

FILTER

By Active Customer?

All

Yes

By staff status

All

Yes

No

By superuser status

All

Yes

No

Customer Data Model

- First Name
- Last Name
- Email
- Phone Number
- Customer's Street Address (1st line only)

Django Full-Text Search Query

- **LIKE** and **OR** with uppercase full-text search
- Boolean **AND** for terms
- Left table joins across relationships

```
WHERE ("grove_customer"."is_staff" = TRUE
      AND (UPPER("grove_customer"."email"::text) LIKE UPPER('%kligman%')
           OR UPPER("grove_customer"."last_name"::text) LIKE UPPER('%kligman%')
           OR UPPER("grove_customer"."first_name"::text) LIKE UPPER('%kligman%')
           OR UPPER("grove_customer"."phone"::text) LIKE UPPER('%kligman%')
           OR UPPER("grove_address"."addr1"::text) LIKE UPPER('%kligman%'))))
```

Elasticsearch!



elasticsearch

Elasticsearch Terminology

Elasticsearch

SQL Translation

Index

Database

Mapping

Table Schema

Field

Table Column

Document

Table Row



Elasticsearch mappings can be brittle!

Inverted Index

Django is awesome!

Inverted Index

django (1)

is (1)

awesome (1)



Document #12345

Customer Record

Inverted Index

kate (1)

kligman (1)

415-555-1212 (1)



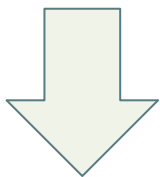
Document #12345



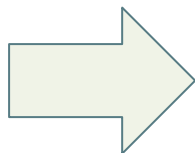
Django Model Primary Key (eg ID)

Django Admin <> Elasticsearch

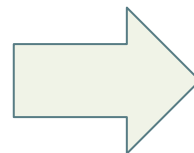
Q Search



elasticsearch



ID #1
ID #2
ID #3



Postgres

Rehydrate Search Results into Querysets

Elasticsearch returns a list of meta identifiers that contain ids

```
response = es_client.query(...)
```

```
ids = [r['meta']['id'] for r in response]
```


Preserve the search results ordering when retrieving objects

```
order = Case(*[When(id=i, then=pos) for pos, i in enumerate(ids)])
```

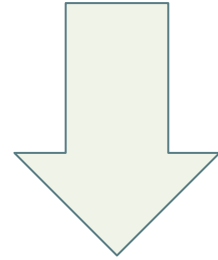
```
results = Customer.objects.filter(id__in=ids).order_by(order)
```

```
return results
```

Django Admin Search Parameters



A screenshot of the Django Admin search interface. It features a search bar with a magnifying glass icon on the left, the text 'kligman' entered, and a 'Search' button on the right. The search bar has a blue border and a light blue background.



<https://grove.co/admin/.../?q=kligman>

Elasticsearch Hook

Change the Django Admin Class Definition

```
class CustomerAdmin(admin.ModelAdmin):
```



```
    def get_search_results(self, request, queryset, search_term):  
        results = CustomerElasticSearch().search(search_term)  
        return results, False
```

Update Elasticsearch

- Signals: `post_save`, `post_create`, `post_delete`
- Model hooks: `def save()`
- Streaming services or scheduled jobs
- Celery tasks

Two Official Python Implementations

`elasticsearch-dsl-py`

`elasticsearch-py`

Customer Data Model

- First Name
- Last Name
- Email
- Phone Number
- Customer's Street Address (1st line only)

Python DSL Mapping

```
class ElasticSearchCustomer(DocType):
```

```
    email = Keyword()
```

```
    first_name = Text()
```

```
    last_name = Text()
```

```
    phone = Text()
```

```
    ship_address__addr1 = Text()
```

Python DSL: Search

```
s = Search(using=connection, index=index)
response = s.query(
    'multi_match',
    query='Kate Kligman',
    fields=['email', 'first_name', 'phone', ...],
    operator='and'
).execute()
```

Lower-level implementation

elasticsearch-py

elasticsearch-py: Query Fragment

```
{  
  'multi_match' : {  
    'query': 'Kate Kligman',  
    'fields': ['email', 'first_name', 'last_name', ...],  
    'operator': 'and'  
  }  
},
```

elasticsearch-py: Nested Query Fragment

```
'nested': {  
    'path': 'shipping_addresses',  
    'query': {  
        'match': {  
            'shipping_addresses.addr1': {  
                'query': search_terms,  
                'operator': 'and'  
            }  
        }  
    }  
}
```

Lesson #1: Own Your Search!

- Our query complexity increased
- DSL issues with nested documents
- Keeping parity with official docs

Lesson #2: Implicit Schemas

- Always create your mapping (schema) first
- Otherwise, Elasticsearch will do it for you
- It will be weird and probably slow
- Delete and try again

Lesson #3: Documentation Versioning

Elasticsearch Reference

+ Elasticsearch Reference: 6.4 (current) ⬆ ⬇ ⬇ ⬆

+ [Getting Started](#)

+ [Set up Elasticsearch](#)

+ [Upgrade Elasticsearch](#)

+ [API Conventions](#)

+ [Document APIs](#)

+ [Search APIs](#)

+ [Aggregations](#)

Lesson #4: Elasticsearch Testing

- Spot checks on queries can appear valid
- Run batched sets of data through to check ordering

Lesson #5: Explore Hosting Options



amazon
Elasticsearch
Service



elastic cloud



The Grove Customer Search Solution

- elasticsearch-py
- AWS Elasticsearch 6.0
- Django admin and an additional API endpoint use the same search
- < 1s searches across millions of records
- High precision search

Resources

- <https://github.com/elastic/elasticsearch-py>
- <https://github.com/elastic/elasticsearch-dsl-py>
- <https://www.elastic.co/guide/en/elasticsearch/reference/current/index.html>
- <https://github.com/dzharii/awesome-elasticsearch>

Questions?
