


# Integrating Elasticsearch with Analytics Workflows

ODSC West  
November 1, 2019

STEPHANIE KIRMER  
 @data\_stephanie  
[github.com/skirmer/elastic\\_analytics](https://github.com/skirmer/elastic_analytics)

## AGENDA

**Introducing Elasticsearch**

**Libraries for R and Python**

**Querying and Filtering**

**Summarizing and Sorting**

**Further Reading**

# Introducing Elasticsearch

- Part of a family of data storage options called **NoSQL**
  - Not the same as tabular or SQL style data storage
  - Allows ingestion of masses of unstructured data quickly/flexibly
- Optimized for fast and **powerful searching**
- **Scales to "big data"**– but usable for small projects
- **Open source tool**

\*Sometimes abbreviated "ES"

# A Sidebar About “Search”

Searching	Not Searching (Querying, for example)
Accommodates uncertainty, ambiguity	Requires precise, specific, clear requests
Tries to help you figure out what you need	Extremely literal
Example: Googling “data storage ideas”	Example: Typing <code>https://www.elastic.co/</code> in browser

Elasticsearch is designed to be great for searching

- Built on top of a technology called Lucene from Apache – Java only

- Allows easy API access to Lucene without Java

Elasticsearch (thanks to Lucene) can handle:

- Typos

- Wildcards

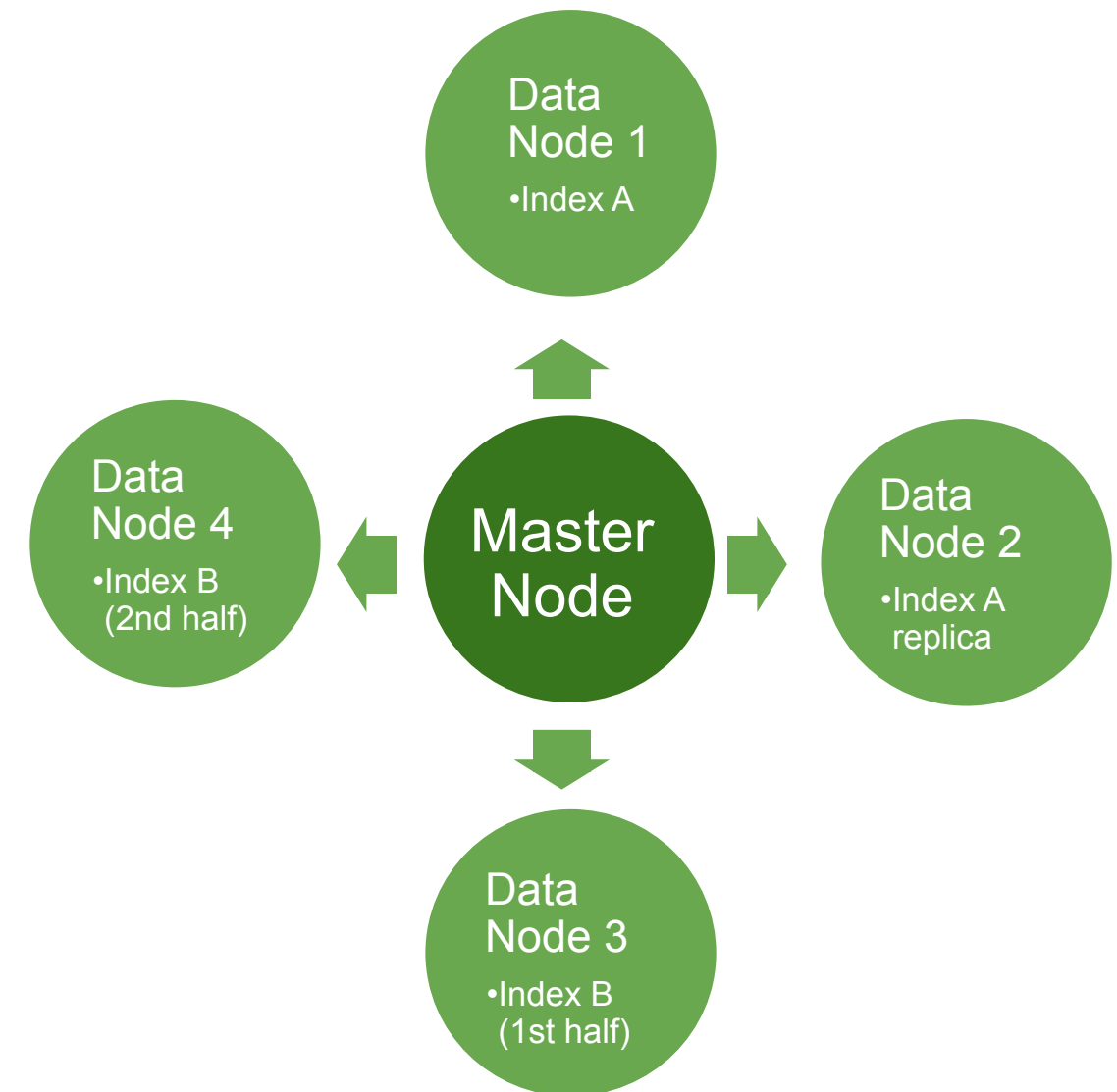
- Fuzzy matching

- Updating and searching simultaneously

# Visualizing Elasticsearch Storage

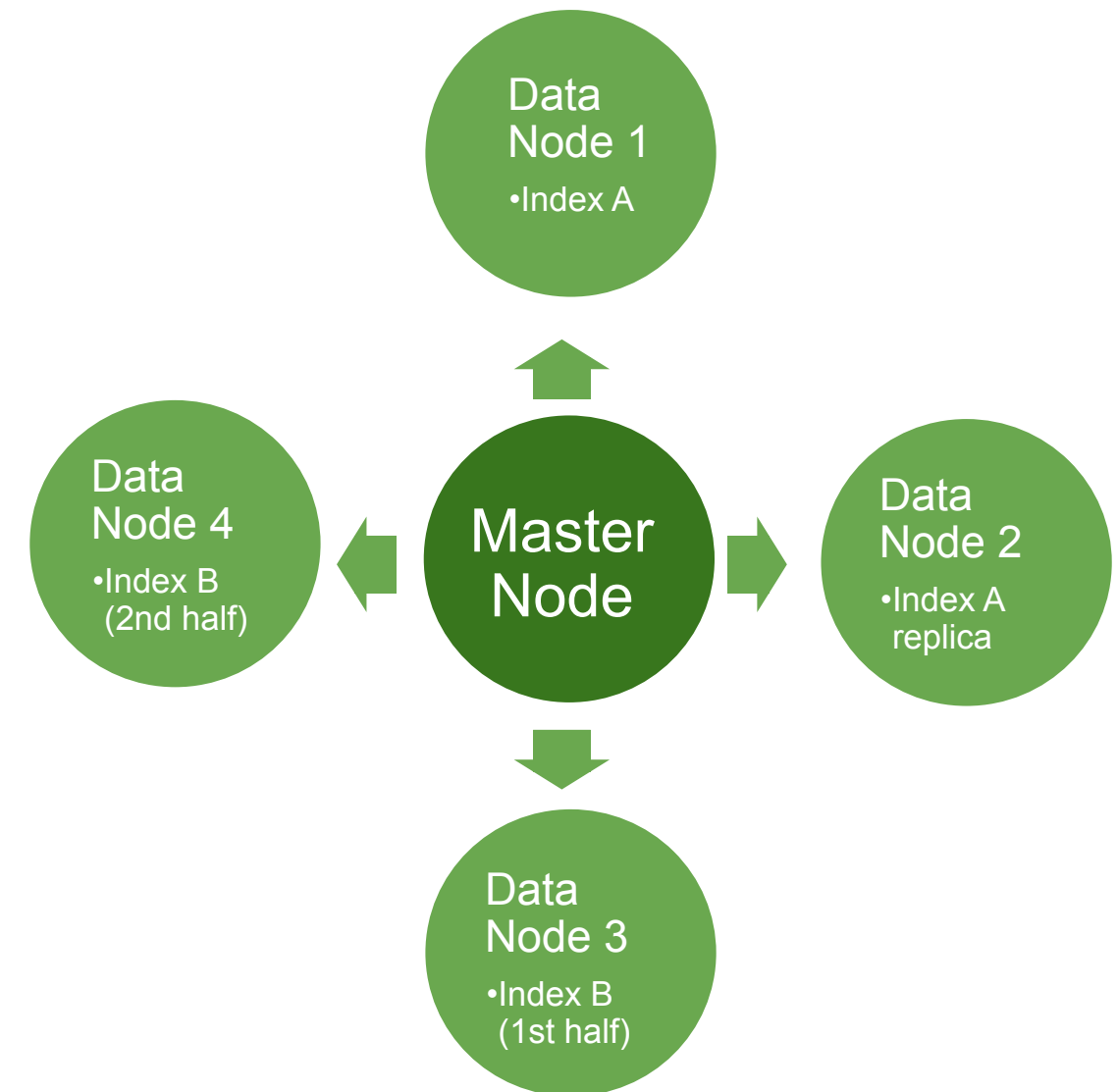
## Data Architecture

- Data is divided into **indices**
- **Indices:**
  - Are user-defined groupings of data with some commonality
  - can live on one node, or
  - can be "sharded" and broken across nodes
  - can be duplicated on different nodes



# Visualizing Elasticsearch Storage

- **Cluster** = group of nodes
- **Master Node** = central manager
  - Manages indices
  - Tracks and organizes nodes
  - Decides which shards to allocate to nodes
- **Data Node** = where data is kept/handled
  - Search, aggregation functions



# Tabular Data vs Document-Based Data

- Nested data
- metadata on outer layer (red braces)
  - content on inner layer (blue braces)

	institution_id	institution_name	deglevl_code	deglevel	degcip_4dig	ciptitle
1	3599	UNIVERSITY OF TEXAS - RIO GRANDE VALLEY	3	Baccalaureate	301	NATURAL RESOURCES CONSERVATION AND
2	3599	UNIVERSITY OF TEXAS - RIO GRANDE VALLEY	3	Baccalaureate	301	NATURAL RESOURCES CONSERVATION AND
3	3599	UNIVERSITY OF TEXAS - RIO GRANDE VALLEY	3	Baccalaureate	501	AREA STUDIES
4	3599	UNIVERSITY OF TEXAS - RIO GRANDE VALLEY	3	Baccalaureate	501	AREA STUDIES
5	3599	UNIVERSITY OF TEXAS - RIO GRANDE VALLEY	3	Baccalaureate	501	AREA STUDIES
6	3599	UNIVERSITY OF TEXAS - RIO GRANDE VALLEY	3	Baccalaureate	501	AREA STUDIES
7	3599	UNIVERSITY OF TEXAS - RIO GRANDE VALLEY	3	Baccalaureate	501	AREA STUDIES
8	3599	UNIVERSITY OF TEXAS - RIO GRANDE VALLEY	3	Baccalaureate	501	AREA STUDIES

```
{ "_index": "utexas",  
  "_type": "data",  
  "_id": "AWbU6WJiWX1fgzrfh4p1",  
  "_score": 1.0,  
  "_source":  
    { "institution_id": 3599,  
      "institution_name": "UNIVERSITY OF TEXAS - RIO GRANDE VALLEY",  
      "deglevl_code": 3,  
      "deglevel": "Baccalaureate",  
      "degcip_4dig": 901,  
      "ciptitle": "COMMUNICATION AND MEDIA STUDIES",  
      "grad_cohort": 2007,  
        "grad_cohort_label": "2007-2009",  
      "year_postgrad": 1,  
      "p25_earnings": 26518.57,  
      "p50_earnings": 42166.31,  
      "p75_earnings": 50439,  
      "system": "utsys",  
      "cellcount": 70 } }
```

Metadat

a

Content

Data courtesy of Annie Millerbernd of the San Antonio Express-News. You can learn more about it and see the original dataset here: <https://data.world/amillerbernd/ut-system-post-grad-earnings>



# Library Characteristics

Library	Returns	Query Language	Supports Authentication	R	Python
uptasticsearch	Tabular	Required	✗	●	●
elastic	JSON	Supported, not required	●	●	✗
elasticsearch-py	JSON	Supported, not required	●	✗	●

For python: `pip install [library name]` (use python3)

For R: `install.packages("[library_name]")`

uptasticsearch python: <https://github.com/uptake/uptasticsearch.git>

uptasticsearch welcomes public PRs!

Code and Data: <https://bit.ly/2LEEz8F>

## KEY CONSIDERATIONS

### Secure Authentication

Do you need to securely log in?

### Output Format

Do you mind handling JSON output?

### Query Construction

Is writing query language a barrier?

# Why Use Elasticsearch?



## Safe

- Copying your data easily and conveniently (via replicas) = if a node fails, your data is safe



## Fast

- ES can search in parallel on multiple nodes and replicas, and find your data faster



## Scalable

- Once you establish your ES database, you can add nodes and allow your database to grow



## Flexible

- Robust search helps you by discerning typos, ranking results, parsing text, and more



## Open Source

- Free to use at small scale, substantial documentation, community support

Among other reasons!

# Query Language Crash Course

---

Elastic Query DSL (domain specific language): a JSON-style syntax built to interact with ES databases.

## Why use query language?

Consistency across interfaces and media

Versatility and power in search, filtering, and aggregating – ES was built to work with this.

## Downsides?

It's sometimes hard to work with – idiosyncratic rules of syntax.

Changes happen – new version releases mean new rules

# Follow Along!

When you see a slide with this gray header, try running the code yourself!



## System Requirements:

Docker installed and running

Repository ready: `git clone https://github.com/skirmer/elastic_analytics.git`

## Setup Steps (see the README for commands to copy/paste)

Get into the top level of the cloned repo

At Terminal:

1. `./supporting_materials/setup_texas.sh 5.5`
2. `curl -X POST 'http://localhost:9200/utexas/_bulk' -H 'Content-Type: application/json' --data-binary @supporting_materials/ut_data.json`

### Handy docker tips

`docker ps` to check your containers  
`docker kill [name]` to hard shutdown your containers

Start up R/RStudio or your favorite Python IDE, and run further commands from there.

# Libraries for R and Python

Choosing the right tool for your needs

## Uptasticsearch (R)

---

```
test_up <- uptasticsearch::es_search(  
  es_host = "http://localhost:9200"  
  , es_index = "utexas"  
  , query_body = query_string  
  , size = 10)
```

```
query_string <- '{"query": {"match_all":{}}}'
```

## Elastic (R)

---

```
conn = elastic::connect(es_host =  
  "http://localhost:9200")
```

```
test_e <- elastic::Search(index = "utexas"  
  , body = query_string  
  , size = 10  
  , raw = TRUE  
  , conn = conn)
```

```
test_e2 <-  
jsonlite::fromJSON(test_e)$hits$hits
```

## Non-Query Language Option:

---

```
test_e <- elastic::Search(  
  index = "utexas"  
  , q = "grad_cohort:*"  
  , size = 10  
  , conn = conn  
  , raw = TRUE)
```

## Uptasticsearch (Py)

```
import json
import uptasticsearch
```

```
uptasticsearch.es_search(
    es_host="http://localhost:9200",
    query_body=query_string,
    es_index="utexas"
)
```

**Note:** uptasticsearch is not on pypi so you need to get it from github.

```
query_dict = {"query": {"match_all": {}}}}
query_string = '{"query": {"match_all": {}}}]'
```

## Elasticsearch-py (Py)

```
from elasticsearch import Elasticsearch
```

```
es = Elasticsearch(['http://localhost:9200'])
res = es.search(
    index="utexas",
    body= query_dict
)

res['hits']['hits']
```

Tip! quickly format using  
`from pandas.io.json`  
`import json_normalize`

## Non-Query Language Option (Elasticsearch\_dsl)

```
from elasticsearch_dsl import Search
```

```
res2 = Search(using = es).query("match", _index = 'utexas').execute()
res2.to_dict()['hits']['hits']
```

# Querying and Filtering



# Identifying Available Fields

**R:**

```
uptasticsearch::get_fields(es_host = "http://localhost:9200",  
es_indices = "utexas")
```

**At Command Line:**

```
curl http://localhost:9200/utexas/_mapping > fields.json
```

# Constructing a Basic Query

Return all records :

```
{  
"query": { "match_all": { } }  
}
```

```
query_dict = {"query": {"match_all": {}}}  
query_string = '{"query": {"match_all": {}}}'
```

# Constructing a Basic Query

Return all records :

```
{  
"query" : { "match_all": { } }  
}
```

# Constructing a Basic Query

Match one field :

```
{  
"query": { "match": { "ciptitle.raw": "COMPUTER SCIENCE" } }  
}
```

Tip: green text means  
something new has been  
added to the query

# Result Sample

```
{'query': {'match': {'ciptitle.raw': 'COMPUTER SCIENCE'}}}
```

	_id	_index	_score	_source.cellcount	_source.ciptitle	_source.degcip_4dig	...	_source.p25_earnings	_source.p50_earnings	_source.p75_earnings
0	AWq8Wt1a9WDJ8JPhaXL7	utexas	6.612041	-1	COMPUTER SCIENCE	1107	...	NaN	NaN	NaN
1	AWq8Wt0n9WDJ8JPhaXCx	utexas	5.731027	47	COMPUTER SCIENCE	1107	...	43797.75	56302.13	65343.96
2	AWq8Wt0n9WDJ8JPhaXCy	utexas	5.731027	53	COMPUTER SCIENCE	1107	...	47552.69	61813.50	79846.65
3	AWq8Wt1a9WDJ8JPhaXL-	utexas	5.731027	-1	COMPUTER SCIENCE	1107	...	NaN	NaN	NaN
4	AWq8Wt0n9WDJ8JPhaXCr	utexas	5.578471	-1	COMPUTER SCIENCE	1107	...	NaN	NaN	NaN
5	AWq8Wt1a9WDJ8JPhaXL9	utexas	5.578471	-1	COMPUTER SCIENCE	1107	...	NaN	NaN	NaN
6	AWq8Wt1a9WDJ8JPhaXL_	utexas	5.578471	-1	COMPUTER SCIENCE	1107	...	NaN	NaN	NaN
7	AWq8Wt1a9WDJ8JPhaXMC	utexas	5.578471	44	COMPUTER SCIENCE	1107	...	43425.67	53358.00	63429.86
8	AWq8Wt0n9WDJ8JPhaXCs	utexas	5.192957	-1	COMPUTER SCIENCE	1107	...	NaN	NaN	NaN
9	AWq8Wt0n9WDJ8JPhaXCt	utexas	5.192957	-1	COMPUTER SCIENCE	1107	...	NaN	NaN	NaN

Things to notice:

- Only Computer Science is shown
- NaNs are present in earnings, cellcount sometimes is -1  
(this means redacted due to small group size)

# Constructing a Basic Query

Match one field AND Greater Than one field :

```
{  
  "query":  
    { "bool" : {  
      "must" : [ { "match": { "ciptitle.raw": "COMPUTER SCIENCE" } }  
        , { "range" : { "cellcount" : { "gt" : 0 } } } ]  
    } }  
}
```

# Result Sample

```
{'query': {'bool': {'must': [{'match': {'ciptitle.raw': 'COMPUTER SCIENCE'}}, {'range': {'cellcount': {'gt': 0}}}]}}}
```

	_id	_index	_score	_source.cellcount	_source.ciptitle	_source.degcip_4dig	...	_source.p25_earnings	_source.p50_earnings
0	AWq8Wt0n9WDJ8JPhaXCx	utexas	6.731027	47	COMPUTER SCIENCE	1107	...	43797.75	56302.13
1	AWq8Wt0n9WDJ8JPhaXCy	utexas	6.731027	53	COMPUTER SCIENCE	1107	...	47552.69	61813.50
2	AWq8Wt1a9WDJ8JPhaXMC	utexas	6.578471	44	COMPUTER SCIENCE	1107	...	43425.67	53358.00
3	AWq8Wt0n9WDJ8JPhaXCu	utexas	6.192957	53	COMPUTER SCIENCE	1107	...	33636.58	41193.17
4	AWq8Wt0n9WDJ8JPhaXCw	utexas	6.192957	40	COMPUTER SCIENCE	1107	...	46477.67	57318.00
5	AWq8Wt1a9WDJ8JPhaXMB	utexas	6.192957	47	COMPUTER SCIENCE	1107	...	44541.92	57588.90
6	AWq8Wt0n9WDJ8JPhaXCv	utexas	6.174845	61	COMPUTER SCIENCE	1107	...	44244.25	58401.60
7	AWq8Wt0n9WDJ8JPhaXCz	utexas	6.174845	55	COMPUTER SCIENCE	1107	...	28598.25	44169.83
8	AWq8Wt1a9WDJ8JPhaXMA	utexas	6.174845	36	COMPUTER SCIENCE	1107	...	73903.50	85285.43

Things to notice:

- Only Computer Science is shown
- Cellcount values are all above 0 - no more NaNs or -1

# Constructing a Basic Query

Match two fields AND Greater Than one field :

```
{  
  "query":  
    { "bool" : {  
      "must" : [ { "match": { "ciptitle.raw": "COMPUTER SCIENCE" } }  
        , { "match": { "institution_id": "3599" } }  
        , { "range" : { "cellcount" : { "gt" : 0 } } } ]  
    } }  
}
```



# Result Sample

	_id	_index	_score	_source.cellcount	_source.ciptitle	_source.degcip_4dig	...	_source.p25_earnings
0	AWq8Wt0n9WDJ8JPhaXCx	utexas	7.731027	47	COMPUTER SCIENCE	1107	...	43797.75
1	AWq8Wt0n9WDJ8JPhaXCy	utexas	7.731027	53	COMPUTER SCIENCE	1107	...	47552.69
2	AWq8Wt0n9WDJ8JPhaXCu	utexas	7.192957	53	COMPUTER SCIENCE	1107	...	33636.58
3	AWq8Wt0n9WDJ8JPhaXCw	utexas	7.192957	40	COMPUTER SCIENCE	1107	...	46477.67
4	AWq8Wt0n9WDJ8JPhaXCv	utexas	7.174845	61	COMPUTER SCIENCE	1107	...	44244.25
5	AWq8Wt0n9WDJ8JPhaXCz	utexas	7.174845	55	COMPUTER SCIENCE	1107	...	28598.25

Things to notice:

- Only Computer Science is shown
- Cellcount values are all above 0
- Only UT-Rio Grande Valley is shown

_source.p50_earnings	_source.p75_earnings	_source.system	_source.year_postgrad	_type
56302.13	65343.96	utsys	1	data
61813.50	79846.65	utsys	5	data
41193.17	51356.40	utsys	1	data
57318.00	89080.00	utsys	10	data
58401.60	72430.97	utsys	5	data
44169.83	60239.68	utsys	1	data

# Some Other Querying Options

## **match\_phrase**

Match a set of words all together.

## **exists**

Supply a field, returns documents that have at least one non-null value in the original field.

## **wildcard**

Pass a string with a wildcard anywhere – but be careful, it can be a slow search!

## **filter**

Just like "must" except without scoring – we'll talk about this in a moment.

## **must\_not**

Instead of "must" – use to omit records with a word or phrase.

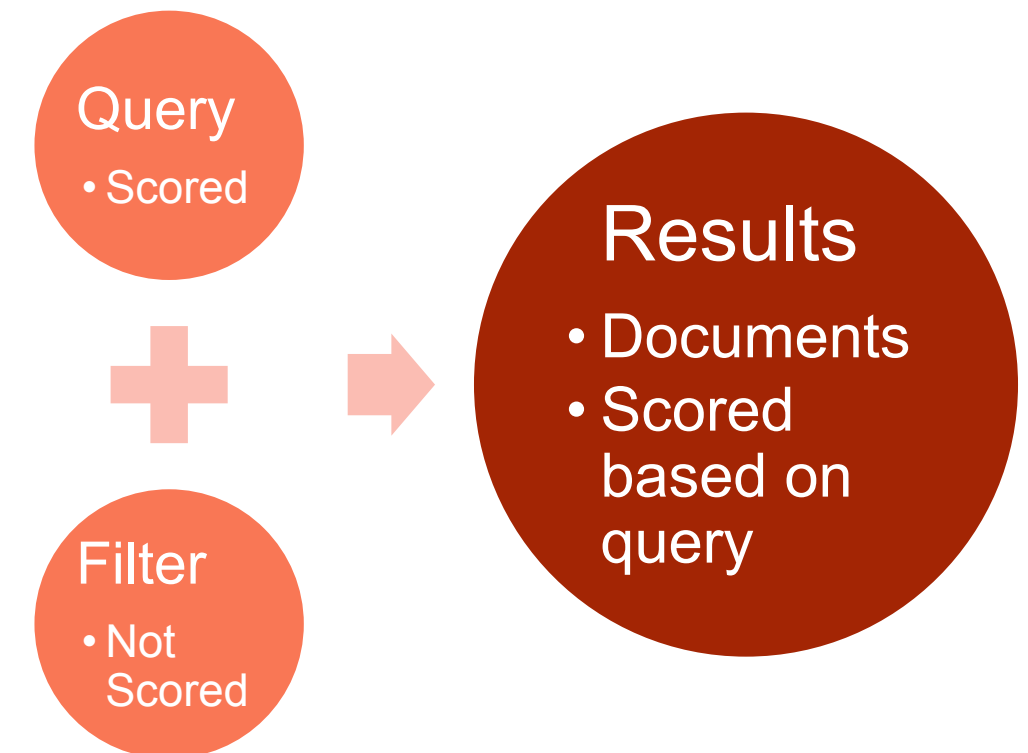
This is just a small sample- ES query language offers many very powerful search options!

# Query vs Filter: Scoring Results

ES queries can provide a **numeric score** indicating how well the document meets the criteria given.

When you use "**query**" at the beginning of the query, you get a score returned alongside your results.

When you use "**filter**", Elasticsearch does not score the results on the given criteria.

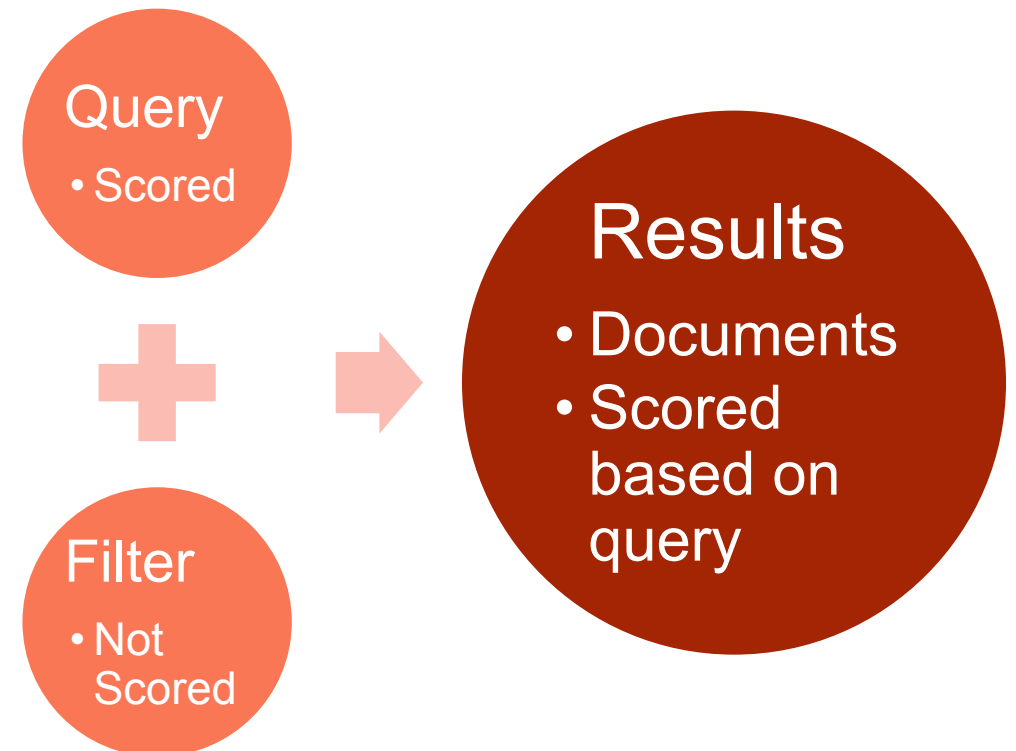


# Query vs Filter: Example

Do we want scores returned for this search? Yes.

```
{ "query":  
  { "bool": {  
    "must": [  
      { "match": { "ciptitle.raw": "AREA STUDIES" } }  
      , { "match": { "deglevel": "Baccalaureate" } }  
    ]  
    , "filter": [  
      { "match": { "institution_id": "3599" } }  
    ]  
  }  
}
```

Do we want the scores to include this criterion? NO.





# Query vs Filter: Example

Same query, first with two criteria scored (1 in filter) and second with only 1 criterion scored.

	_id	_index	_score	_source.cellcount	_source.ciptitle	_source.degcip_4dig	...
0	Awq8Wt0n9WDJ8JPhaXCJ	utexas	5.474233	-1	AREA STUDIES	501	...
1	Awq8Wt0n9WDJ8JPhaXCB	utexas	5.168861	-1	AREA STUDIES	501	...
2	Awq8Wt0n9WDJ8JPhaXCI	utexas	5.168861	-1	AREA STUDIES	501	...
3	Awq8Wt0n9WDJ8JPhaXCE	utexas	5.154995	-1	AREA STUDIES	501	...
4	Awq8Wt0n9WDJ8JPhaXCG	utexas	5.154995	-1	AREA STUDIES	501	...
5	Awq8Wt0n9WDJ8JPhaXCC	utexas	4.792192	-1	AREA STUDIES	501	...
6	Awq8Wt0n9WDJ8JPhaXCH	utexas	4.792192	-1	AREA STUDIES	501	...
7	Awq8Wt0n9WDJ8JPhaXCD	utexas	4.733091	-1	AREA STUDIES	501	...
8	Awq8Wt0n9WDJ8JPhaXCF	utexas	4.733091	-1	AREA STUDIES	501	...
[9 rows x 18 columns]							
	_id	_index	_score	_source.cellcount	_source.ciptitle	_source.degcip_4dig	...
0	Awq8Wt0n9WDJ8JPhaXCJ	utexas	5.145704	-1	AREA STUDIES	501	...
1	Awq8Wt0n9WDJ8JPhaXCB	utexas	4.843724	-1	AREA STUDIES	501	...
2	Awq8Wt0n9WDJ8JPhaXCI	utexas	4.843724	-1	AREA STUDIES	501	...
3	Awq8Wt0n9WDJ8JPhaXCE	utexas	4.813467	-1	AREA STUDIES	501	...
4	Awq8Wt0n9WDJ8JPhaXCG	utexas	4.813467	-1	AREA STUDIES	501	...
5	Awq8Wt0n9WDJ8JPhaXCC	utexas	4.479859	-1	AREA STUDIES	501	...
6	Awq8Wt0n9WDJ8JPhaXCH	utexas	4.479859	-1	AREA STUDIES	501	...
7	Awq8Wt0n9WDJ8JPhaXCD	utexas	4.443958	-1	AREA STUDIES	501	...
8	Awq8Wt0n9WDJ8JPhaXCF	utexas	4.443958	-1	AREA STUDIES	501	...

```
"match": {  
  "deglevel":  
    "Baccalaureate"  
}
```

In query (scored)

```
"match": {  
  "deglevel":  
    "Baccalaureate"  
}
```

In filter (not scored)

# Summarizing and Sorting

Get fancier with your searching!

# Sorting in Query

Query, then sort the output

```
{ "query":  
  { "bool" : {  
    "must" : [ { "range": { "p50_earnings": { "gte" : 75000 } } }  
      , { "match": { "institution_id": "3658" } }  
      , { "range": { "cellcount" : { "gt" : 0 } } } ]  
    }  
  },  
  "sort": {  
    "ciptitle.raw": "asc"  
  }  
}
```

# Result Sample

```
{'query': {'bool': {'must': [{'range': {'p50_earnings': {'gte': 75000}}}, {'match': {'institution_id': '3658'}}, {'range': {'cellcount': {'gt': 0}}}]}, 'sort': {'ciptitle.r
_index _source.cellcount _source.ciptitle _source.deglevel ... _source.institution_name _source.p25_earnings _source.p50_earnings
0 utexas 450 ACCOUNTING AND RELATED SERVICES Baccalaureate ... UNIVERSITY OF TEXAS-AUSTIN 75365.58 95295.42
1 utexas 401 ACCOUNTING AND RELATED SERVICES Baccalaureate ... UNIVERSITY OF TEXAS-AUSTIN 93491.84 129329.43
2 utexas 525 ACCOUNTING AND RELATED SERVICES Baccalaureate ... UNIVERSITY OF TEXAS-AUSTIN 68325.54 86671.19
3 utexas 536 ACCOUNTING AND RELATED SERVICES Baccalaureate ... UNIVERSITY OF TEXAS-AUSTIN 70508.76 89453.86
4 utexas 317 ACCOUNTING AND RELATED SERVICES Baccalaureate ... UNIVERSITY OF TEXAS-AUSTIN 83546.25 124343.03
5 utexas 131 AEROSPACE, AERONAUTICAL AND ASTRONAUTICAL ENGI... Baccalaureate ... UNIVERSITY OF TEXAS-AUSTIN 83546.25 105716.76
6 utexas 175 AEROSPACE, AERONAUTICAL AND ASTRONAUTICAL ENGI... Baccalaureate ... UNIVERSITY OF TEXAS-AUSTIN 63792.66 77877.64
7 utexas 193 AEROSPACE, AERONAUTICAL AND ASTRONAUTICAL ENGI... Baccalaureate ... UNIVERSITY OF TEXAS-AUSTIN 66558.09 81258.97
8 utexas 77 AEROSPACE, AERONAUTICAL AND ASTRONAUTICAL ENGI... Baccalaureate ... UNIVERSITY OF TEXAS-AUSTIN 81609.44 102520.63
9 utexas 75 AEROSPACE, AERONAUTICAL AND ASTRONAUTICAL ENGI... Baccalaureate ... UNIVERSITY OF TEXAS-AUSTIN 67923.63 84210.30
```

Things to notice:

- ciptitle field is sorted alphabetically
- Only UT-Austin is shown
- p50\_earnings are all above \$75,000 as requested



# Summarizing in Query

Summarize one field:

```
{
  "aggs": {
    "common_majors": {
      "terms": {
        "field": "ciptitle.raw"
      }
    }
  }
}
```

Create a new field called `common_majors`, which returns the number of records matching each value of `ciptitle.raw`.

Python Tip! Extract the correct piece of your query result:

```
res['aggregations']['common_majors']['buckets']
```

# Result Sample

Produces:

	doc_count	key
0	153	HISTORY
1	149	HEALTH PROFESSIONS AND RELATED PROGRAMS
2	130	BIOLOGICAL AND BIOMEDICAL SCIENCES
3	93	CLINICAL/MEDICAL LABORATORY SCIENCE/RESEARCH A...
4	90	MATHEMATICS
5	80	LIBERAL ARTS AND SCIENCES, GENERAL STUDIES AND...
6	78	PHYSICAL SCIENCES
7	77	CHEMISTRY
8	75	HEALTH AND PHYSICAL EDUCATION/FITNESS
9	75	POLITICAL SCIENCE AND GOVERNMENT

At Terminal: `./supporting_materials/cleanup_local.sh`

This shuts down the docker container, destroying our demo database – but you can create it again just by going back to the beginning.

# Further Reading

# Explore More about Elasticsearch!



## ES Query Language

- <http://elasticsearch-cheatsheet.jolicode.com/>
- [https://elasticsearch-dsl.readthedocs.io/en/latest/search\\_dsl.html](https://elasticsearch-dsl.readthedocs.io/en/latest/search_dsl.html)
- [https://www.elastic.co/guide/en/elasticsearch/reference/current/\\_introducing\\_the\\_query\\_language.html](https://www.elastic.co/guide/en/elasticsearch/reference/current/_introducing_the_query_language.html)
- <https://www.elastic.co/guide/en/elasticsearch/reference/6.4/query-dsl-bool-query.html>
- <https://www.elastic.co/guide/en/elasticsearch/reference/6.4/query-filter-context.html>
- <https://logz.io/blog/elasticsearch-queries/>

github.com/skirmer/elastic\_analytics  
www.stephaniekirmer.com  
@data\_stephanie

## Library Docs

- <https://elasticsearch-py.readthedocs.io/en/master/index.html>
- <https://github.com/ropensci/elastic>
- <https://github.com/UptakeOpenSource/uptasticsearch> – Make contributions, the packages are always improving!

## Data Credit:

The data being used in this tutorial is from data.world, and comes out of the hard work done by Annie Millerbernd of the San Antonio Express-News. You can learn more about it and see the original dataset here: <https://data.world/amillerbernd/ut-system-post-grad-earnings>

# Thank You!

---