Make Your Data

FABulous

Philipp Krenn

@xeraa

elastic

elastic

Developer Advocate

Meetup

ViennaDB
Papers We Love Vienna

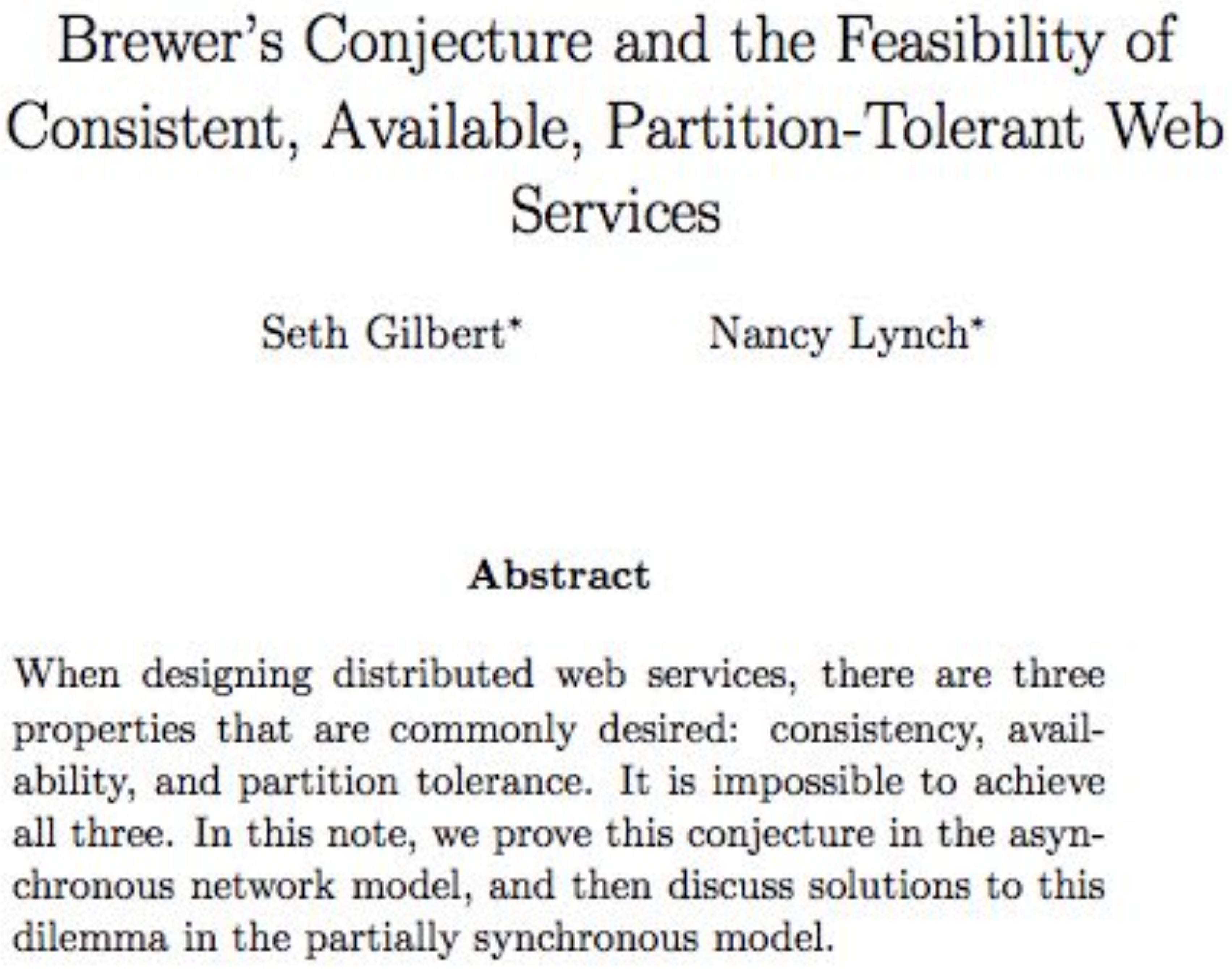
What is the perfect datastore solution?

It depends...

Pick your tradeoffs

MAGIC

CAP Theorem



Brewer's Conjecture and the Feasibility of Consistent, Available, Partition-Tolerant Web Services

Seth Gilbert[*]   Nancy Lynch[*]

Abstract

When designing distributed web services, there are three properties that are commonly desired: consistency, availability, and partition tolerance. It is impossible to achieve all three. In this note, we prove this conjecture in the asynchronous network model, and then discuss solutions to this dilemma in the partially synchronous model.

Consistent

"[...] a **total order** on all operations such that each operation looks as if it were completed at a single instant."

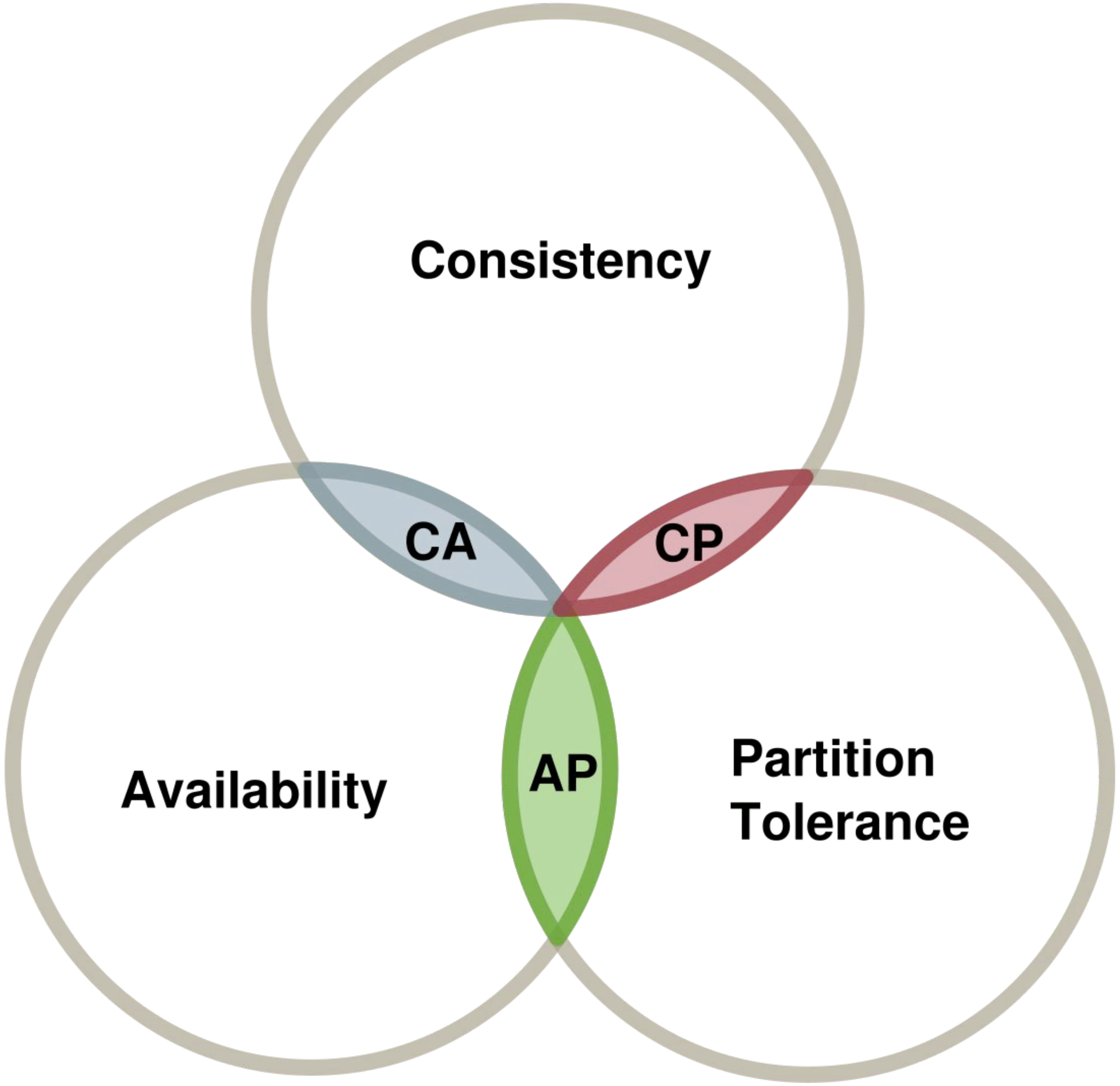
Available

"[...] every request received by a non-failing node in the system must result in a response."

Partition Tolerant

"[...] the network will be allowed to lose arbitrarily many messages sent from one node to another."

elastic



https://berb.github.io/diploma-thesis/original/061_challenge.html

Robinson Crusoe

> /dev/null breaks CAP: effect of write are always consistent, it's always available, and all replicas are consistent even during partitions.
>
> — https://twitter.com/ashic/status/591511683987701760

FAB Theory

elastic

Fast

Near real-time instead of batch processing

Accurate

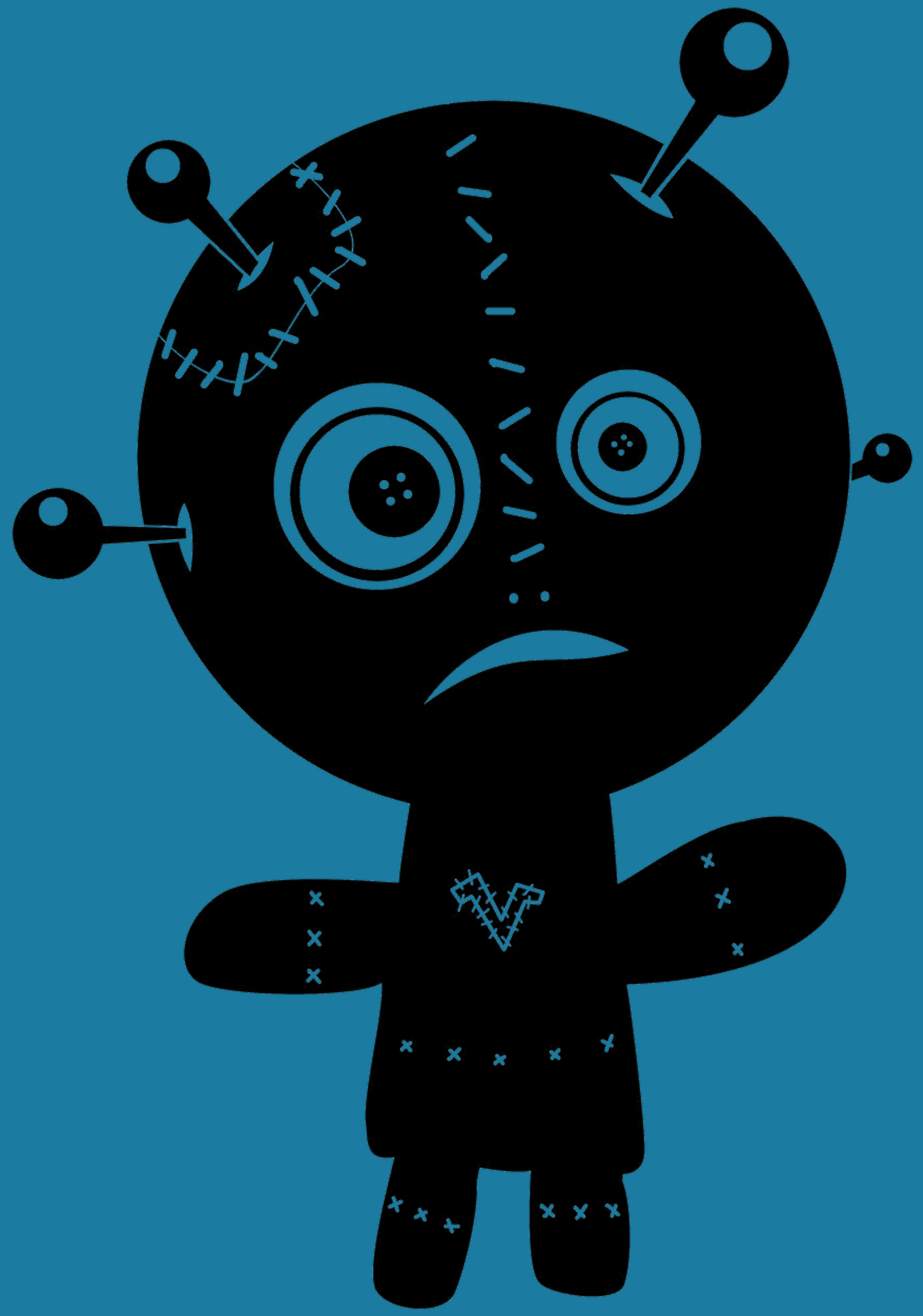
Exact instead of approximate results

Big

Parallel computing tools are needed to handle the data

The 42 V's of Big Data and Data Science

https://www.elderresearch.com/company/blog/42-v-of-big-data



elastic

SAY BIG DATA ONE MORE TIME

Fast ✅

Big ✅

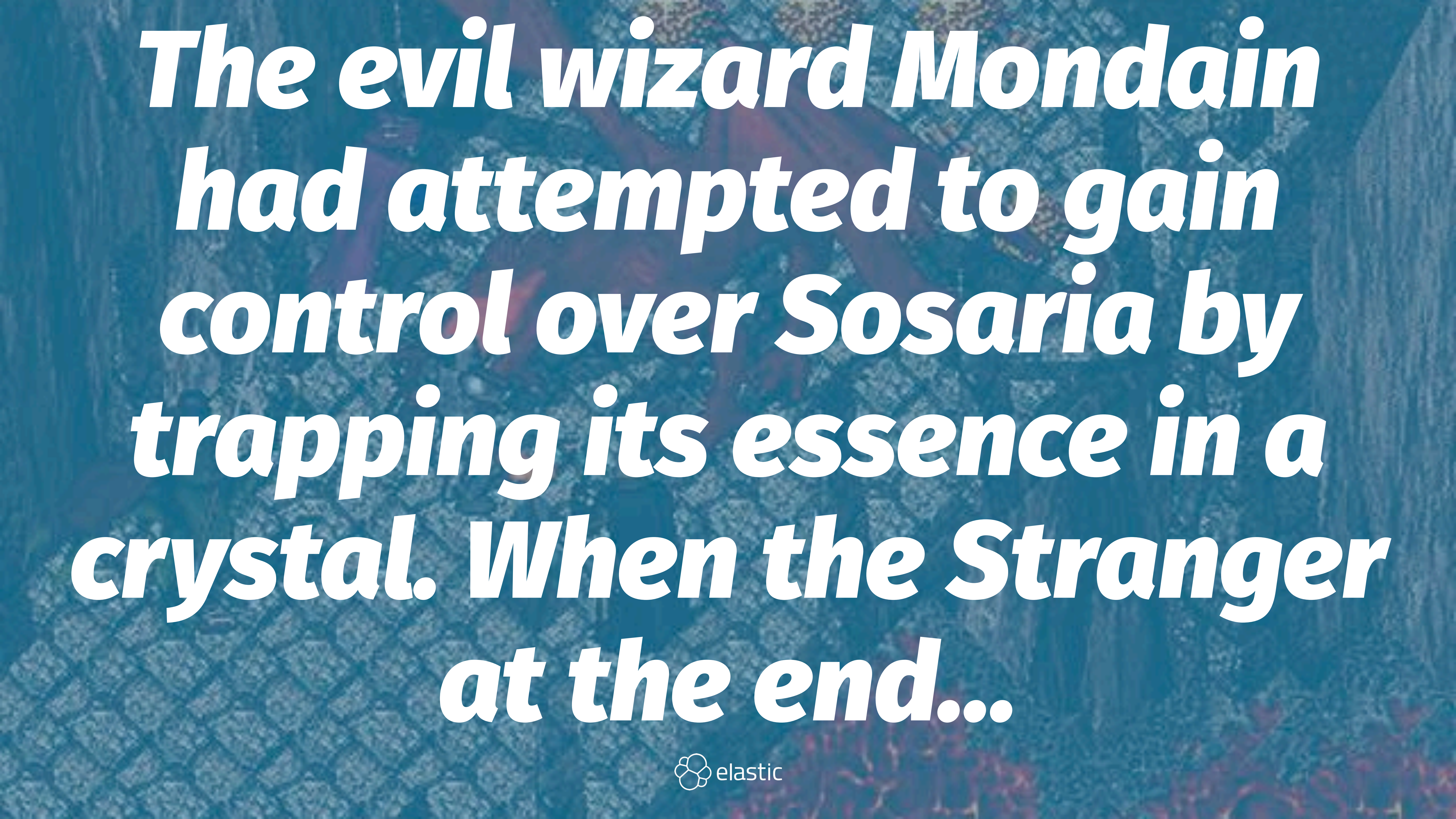
Accurate ❔

elasticsearch

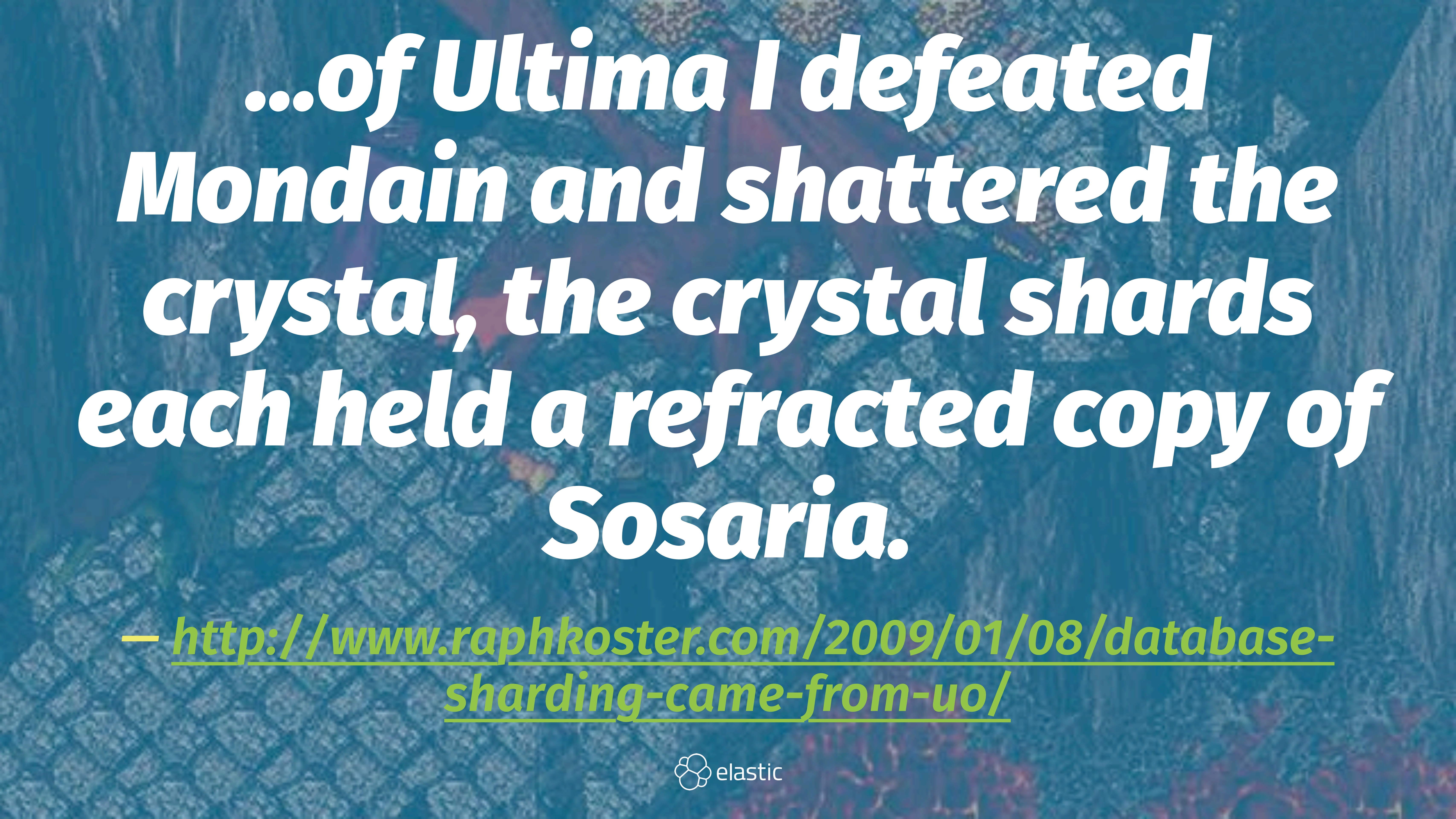
elastic

Shard

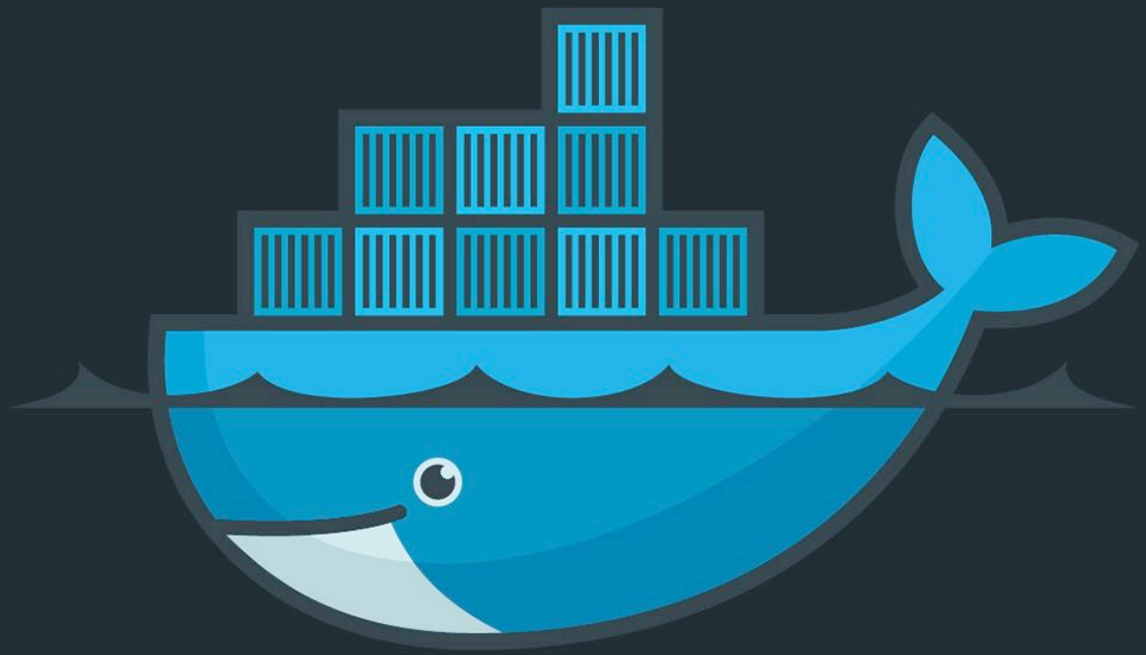
Unit of scale



The evil wizard Mondain had attempted to gain control over Sosaria by trapping its essence in a crystal. When the Stranger at the end…



> ***…of Ultima I defeated Mondain and shattered the crystal, the crystal shards each held a refracted copy of Sosaria.***
>
> — *http://www.raphkoster.com/2009/01/08/database-sharding-came-from-uo/*



docker

HD

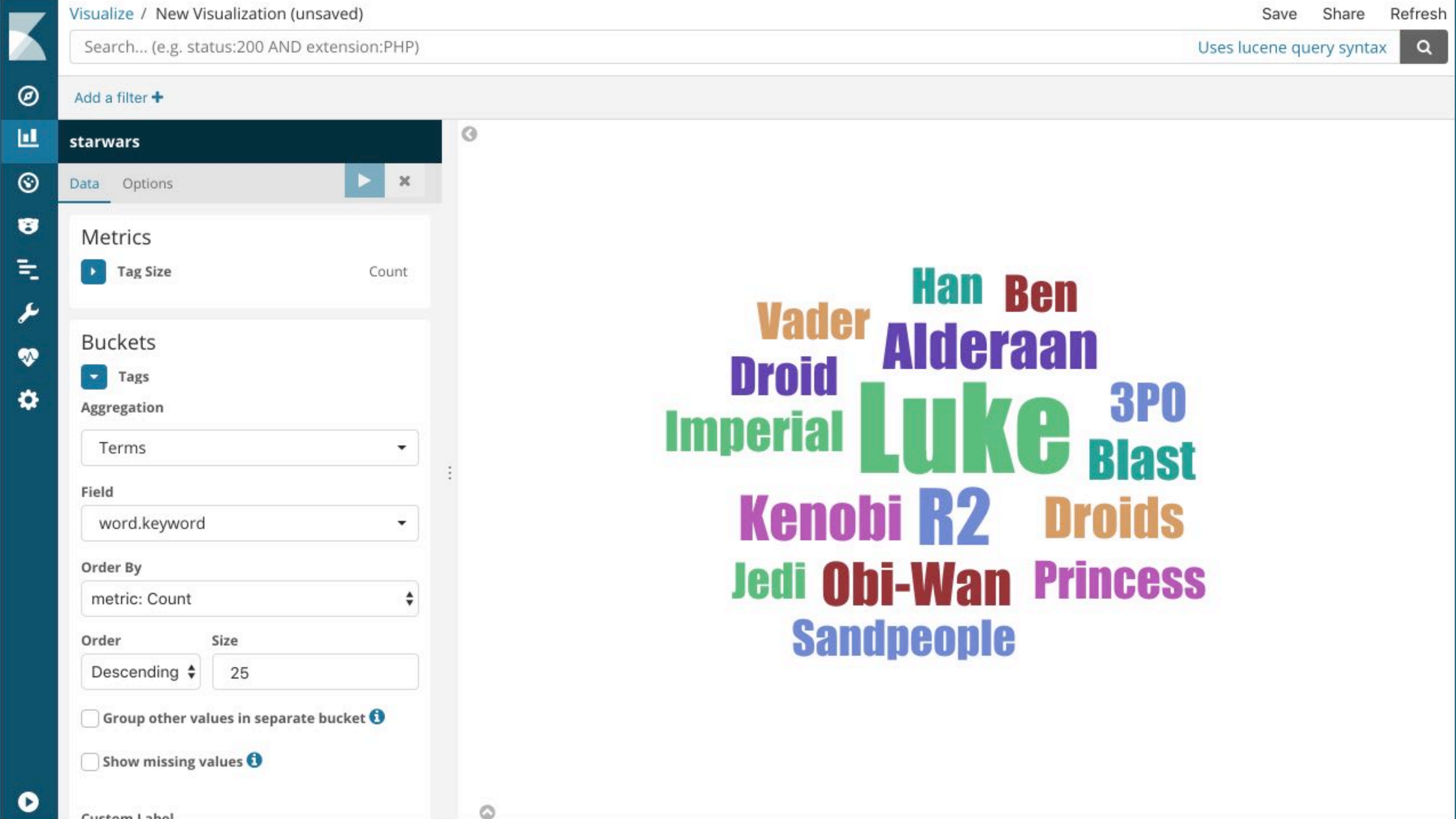
```yaml
---
version: '2'
services:
  kibana:
    image: docker.elastic.co/kibana/kibana:6.2.4
    links:
      - elasticsearch
    ports:
      - 5601:5601

  elasticsearch:
    image: docker.elastic.co/elasticsearch/elasticsearch:6.2.4
    volumes:
      - esdata1:/usr/share/elasticsearch/data
    ports:
      - 9200:9200

volumes:
  esdata1:
    driver: local
```

Terms Aggregation

Word Count	Word Count
Luke **64**	Droid **13**
R2 **31**	3PO **13**
Alderaan **20**	Princess **12**
Kenobi **19**	Ben **11**
Obi-Wan **18**	Vader **11**
Droids **16**	Han **10**
Blast **15**	Jedi **10**
Imperial **15**	Sandpeople **10**

```
PUT starwars
{
  "settings": {
    "number_of_shards": 5,
    "number_of_replicas": 0
  }
}
```

```
{ "index" : { "_index" : "starwars", "_type" : "_doc", "routing": "0" } }
{ "word" : "Luke" }
{ "index" : { "_index" : "starwars", "_type" : "_doc", "routing": "1" } }
{ "word" : "Luke" }
{ "index" : { "_index" : "starwars", "_type" : "_doc", "routing": "2" } }
{ "word" : "Luke" }
{ "index" : { "_index" : "starwars", "_type" : "_doc", "routing": "3" } }
{ "word" : "Luke" }
...
```



Metrics

Tag Size: Count

Buckets

Tags

Aggregation: Terms

Field: word.keyword

Order By: metric: Count

Order: Descending

Size: 25

☐ Group other values in separate bucket

☐ Show missing values

Custom Label

```
GET starwars/_search
{
  "query": {
    "match": {
      "word": "Luke"
    }
  }
}
```

```
{
  "took": 6,
  "timed_out": false,
  "_shards": {
    "total": 5,
    "successful": 5,
    "skipped": 0,
    "failed": 0
  },
  "hits": {
    "total": 64,
    "max_score": 3.2049506,
    "hits": [
      {
        "_index": "starwars",
        "_type": "_doc",
        "_id": "0vVdy2IBkmPuaFRg659y",
        "_score": 3.2049506,
        "_routing": "1",
        "_source": {
          "word": "Luke"
        }
      },
      ...
```

```
GET starwars/_search
{
  "aggs": {
    "most_common": {
      "terms": {
        "field": "word.keyword",
        "size": 1
      }
    }
  },
  "size": 0
}
```

```
{
  "took": 13,
  "timed_out": false,
  "_shards": {
    "total": 5,
    "successful": 5,
    "skipped": 0,
    "failed": 0
  },
  "hits": {
    "total": 288,
    "max_score": 0,
    "hits": []
  },
  "aggregations": {
    "most_common": {
      "doc_count_error_upper_bound": 10,
      "sum_other_doc_count": 232,
      "buckets": [
        {
          "key": "Luke",
          "doc_count": 56
        }
      ]
    }
  }
}
```

```
{ "index" : { "_index" : "starwars", "_type" : "_doc", "routing": "0" } }
{ "word" : "Luke" }
{ "index" : { "_index" : "starwars", "_type" : "_doc", "routing": "1" } }
{ "word" : "Luke" }
...
{ "index" : { "_index" : "starwars", "_type" : "_doc", "routing": "9" } }
{ "word" : "Luke" }
{ "index" : { "_index" : "starwars", "_type" : "_doc", "routing": "0" } }
{ "word" : "Luke" }
{ "index" : { "_index" : "starwars", "_type" : "_doc", "routing": "0" } }
{ "word" : "Luke" }
...
```

Routing

```
shard# = hash(_routing) % #primary_shards
```

```
GET _cat/shards?index=starwars&v

index    shard prirep state   docs store ip         node
starwars 3     p      STARTED   58 6.4kb 172.19.0.2 Q88C3vO
starwars 4     p      STARTED   26 5.2kb 172.19.0.2 Q88C3vO
starwars 2     p      STARTED   71 6.9kb 172.19.0.2 Q88C3vO
starwars 1     p      STARTED   63 6.6kb 172.19.0.2 Q88C3vO
starwars 0     p      STARTED   70 6.7kb 172.19.0.2 Q88C3vO
```

(Sub) Results Per Shard

```
shard_size = (size * 1.5 + 10)
```

How Many?

Results per shard

Results for aggregation

```
"doc_count_error_upper_bound": 10

"sum_other_doc_count": 232
```

```
GET starwars/_search
{
  "aggs": {
    "most_common": {
      "terms": {
        "field": "word.keyword",
        "size": 1,
        "show_term_doc_count_error": true
      }
    }
  },
  "size": 0
}
```

```
"aggregations": {
  "most_common": {
    "doc_count_error_upper_bound": 10,
    "sum_other_doc_count": 232,
    "buckets": [
      {
        "key": "Luke",
        "doc_count": 56,
        "doc_count_error_upper_bound": 9
      }
    ]
  }
}
```

```
GET starwars/_search
{
  "aggs": {
    "most_common": {
      "terms": {
        "field": "word.keyword",
        "size": 1,
        "shard_size": 20,
        "show_term_doc_count_error": true
      }
    }
  },
  "size": 0
}
```

```
"aggregations": {
  "most_common": {
    "doc_count_error_upper_bound": 0,
    "sum_other_doc_count": 224,
    "buckets": [
      {
        "key": "Luke",
        "doc_count": 64,
        "doc_count_error_upper_bound": 0
      }
    ]
  }
}
```

Inverse Document Frequency

elastic

```
GET starwars/_search
{
  "query": {
    "match": {
      "word": "Luke"
    }
  }
}
```

```
...
{
  "_index": "starwars",
  "_type": "_doc",
  "_id": "0vVdy2IBkmPuaFRg659y",
  "_score": 3.2049506,
  "_routing": "1",
  "_source": {
    "word": "Luke"
  }
},
{
  "_index": "starwars",
  "_type": "_doc",
  "_id": "2PVdy2IBkmPuaFRg659y",
  "_score": 3.2049506,
  "_routing": "7",
  "_source": {
    "word": "Luke"
  }
},
{
  "_index": "starwars",
  "_type": "_doc",
  "_id": "0_Vdy2IBkmPuaFRg659y",
  "_score": 3.1994843,
  "_routing": "2",
  "_source": {
    "word": "Luke"
  }
},
...
```

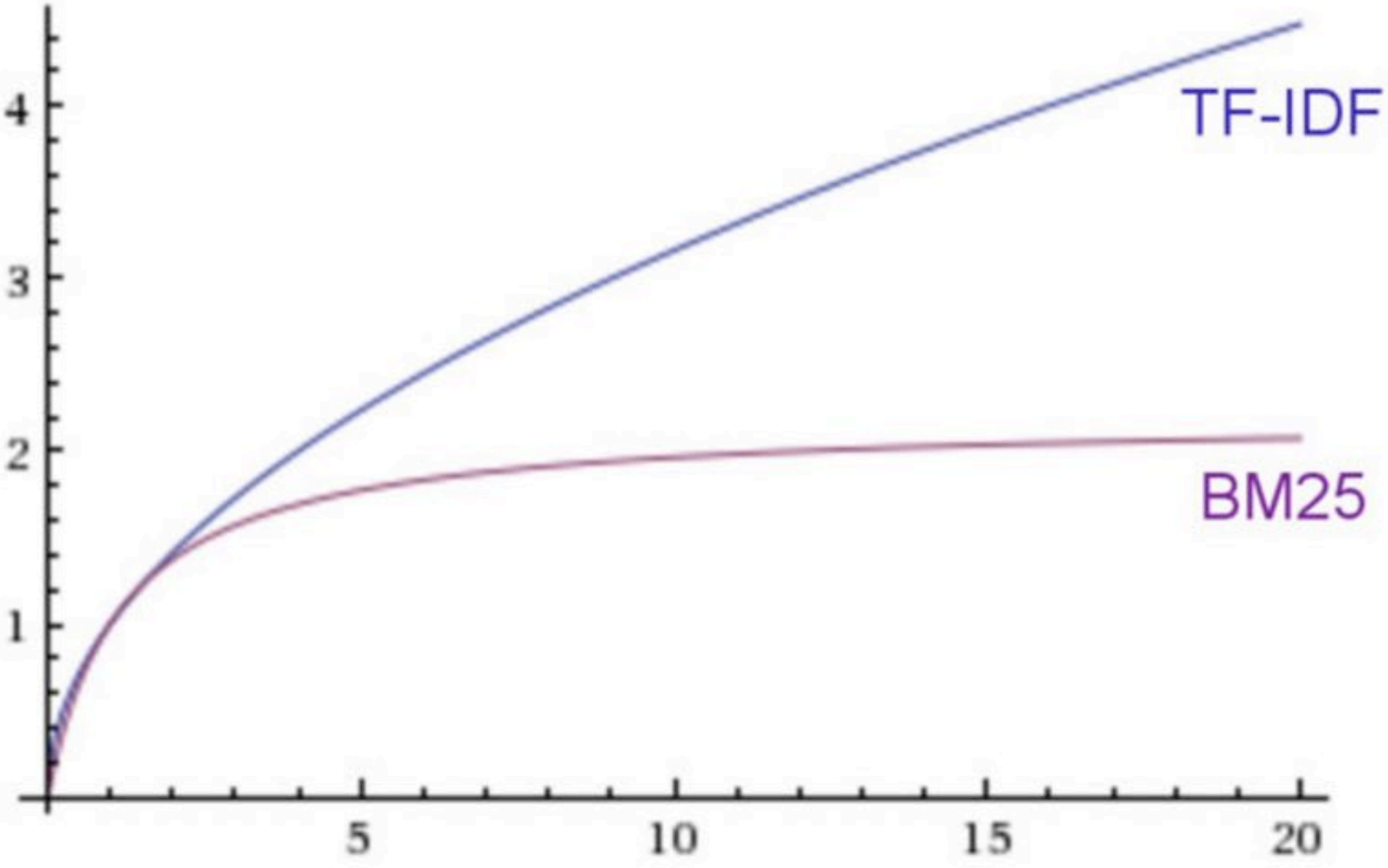
Term Frequency / Inverse Document Frequency (TF/IDF)

BM25

Default in Elasticsearch 5.0

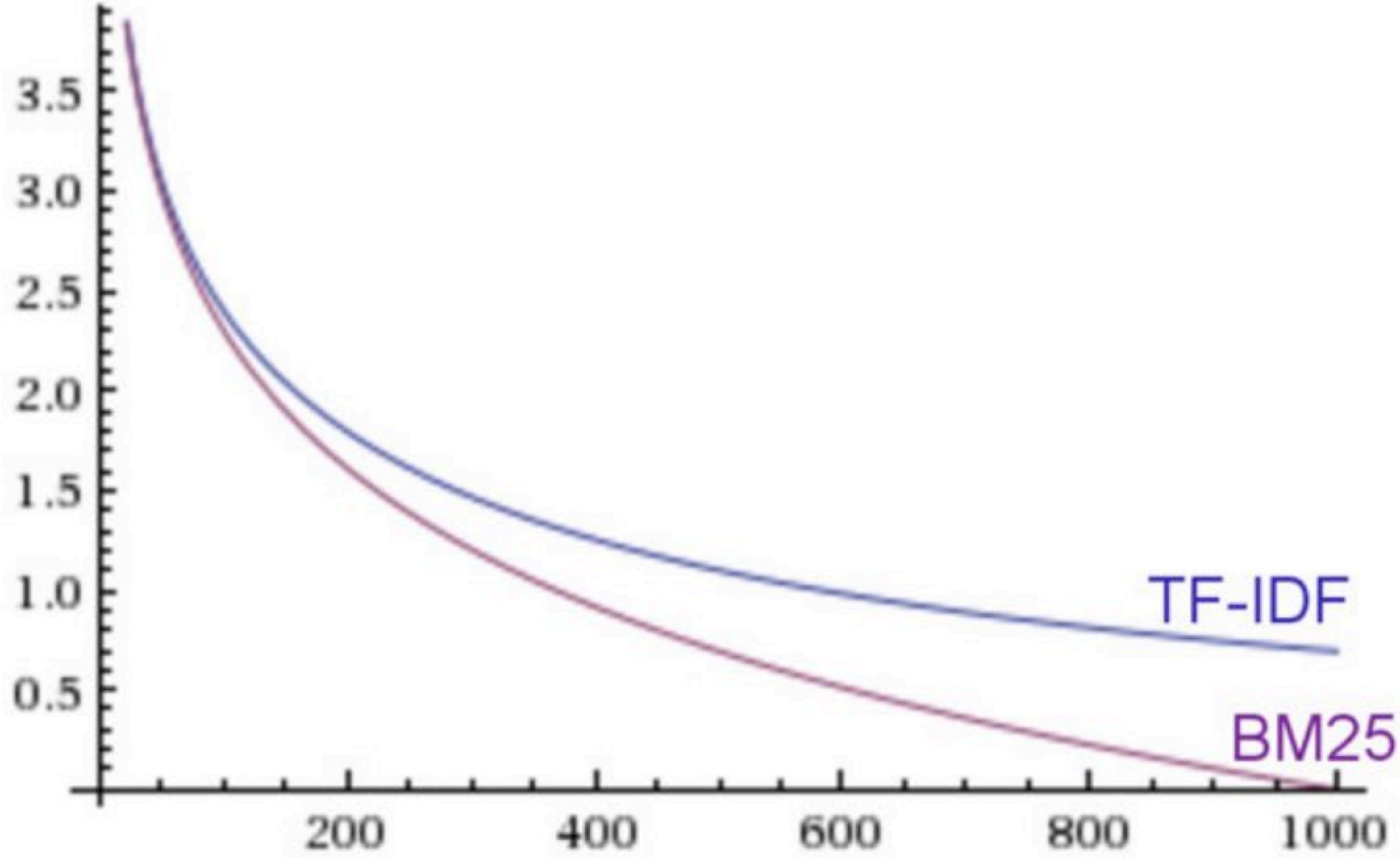
Term Frequency

$$tf(t\ in\ d) = \sqrt{frequency}$$


4
3
2
1
5
10
15
20
TF-IDF
BM25

Inverse Document Frequency

$$idf(t) = 1 + log(\frac{numDocs}{docFreq + 1})$$


3.5
3.0
2.5
2.0
1.5
1.0
0.5
200
400
600
800
1000
TF-IDF
BM25

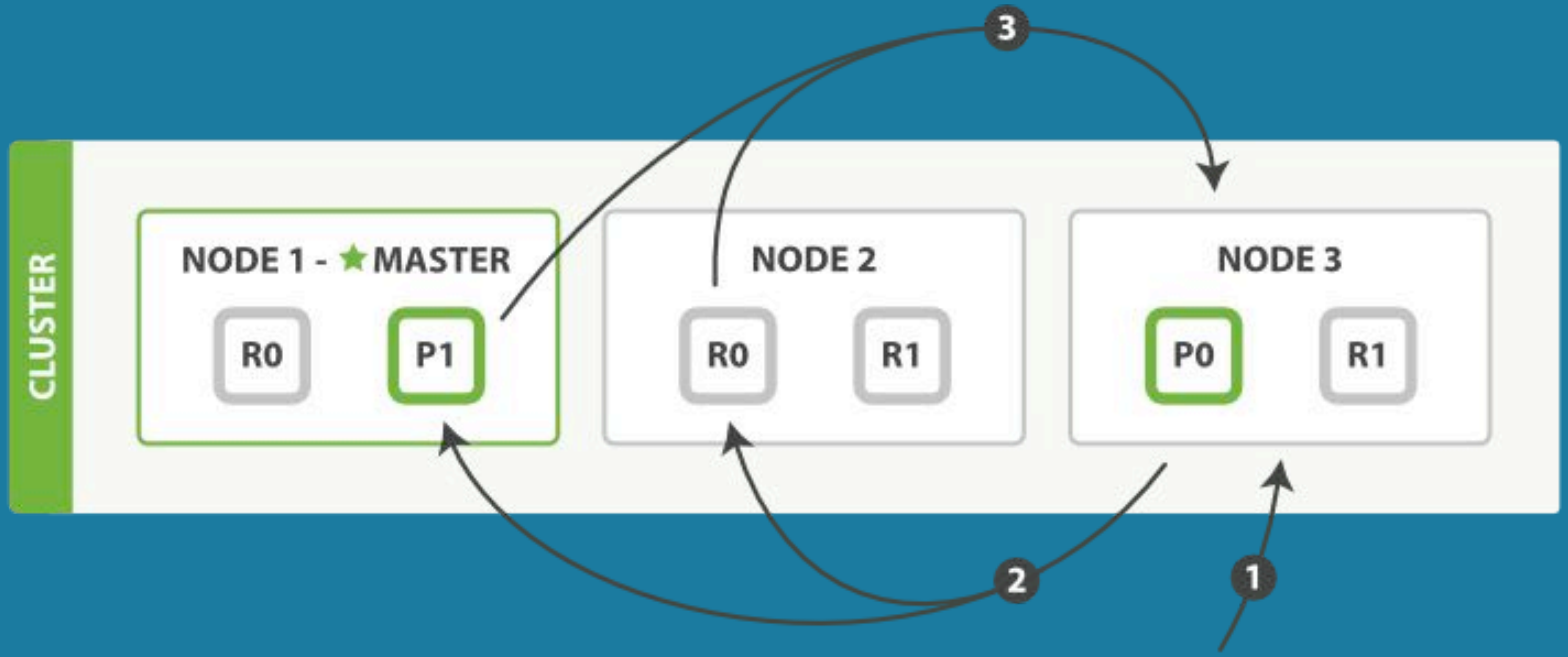
Field-Length Norm

$$norm(d) = \frac{1}{\sqrt{numTerms}}$$

Query Then Fetch

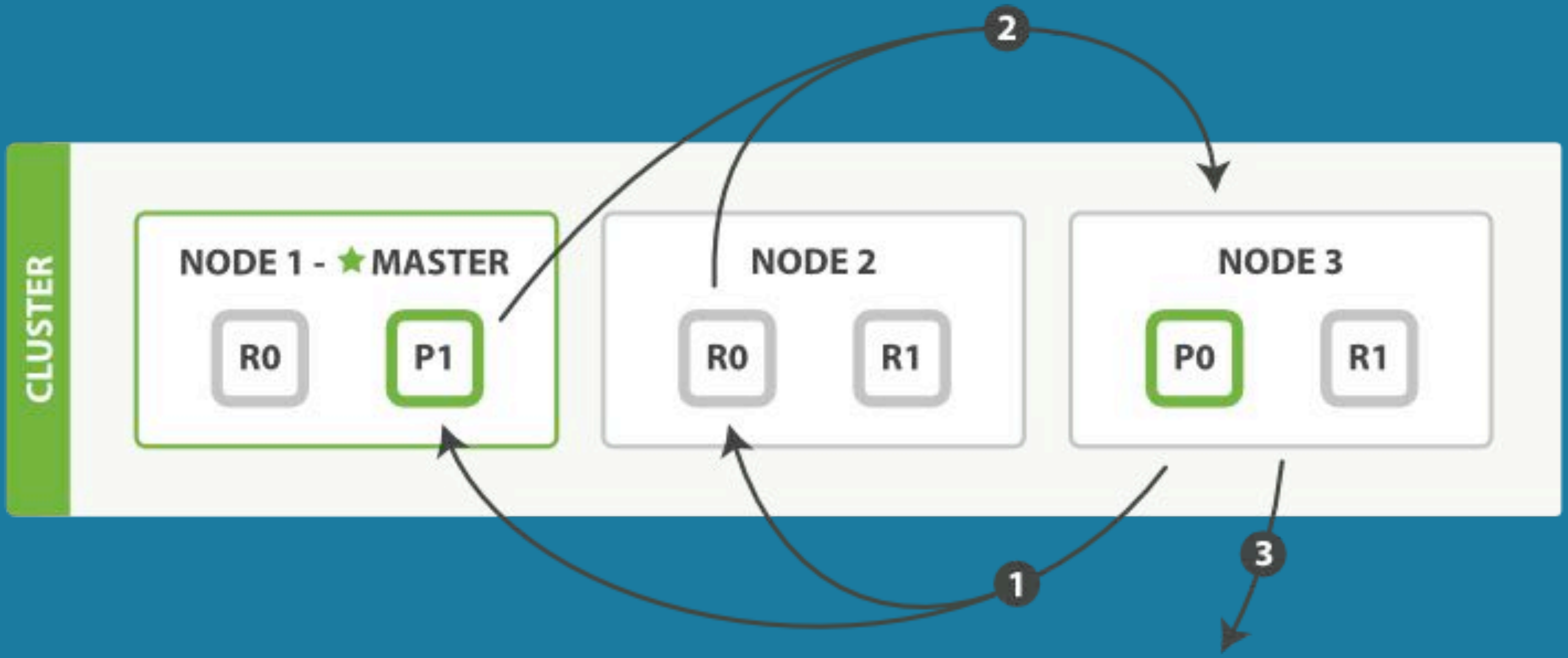
elastic

Query



elastic

Fetch



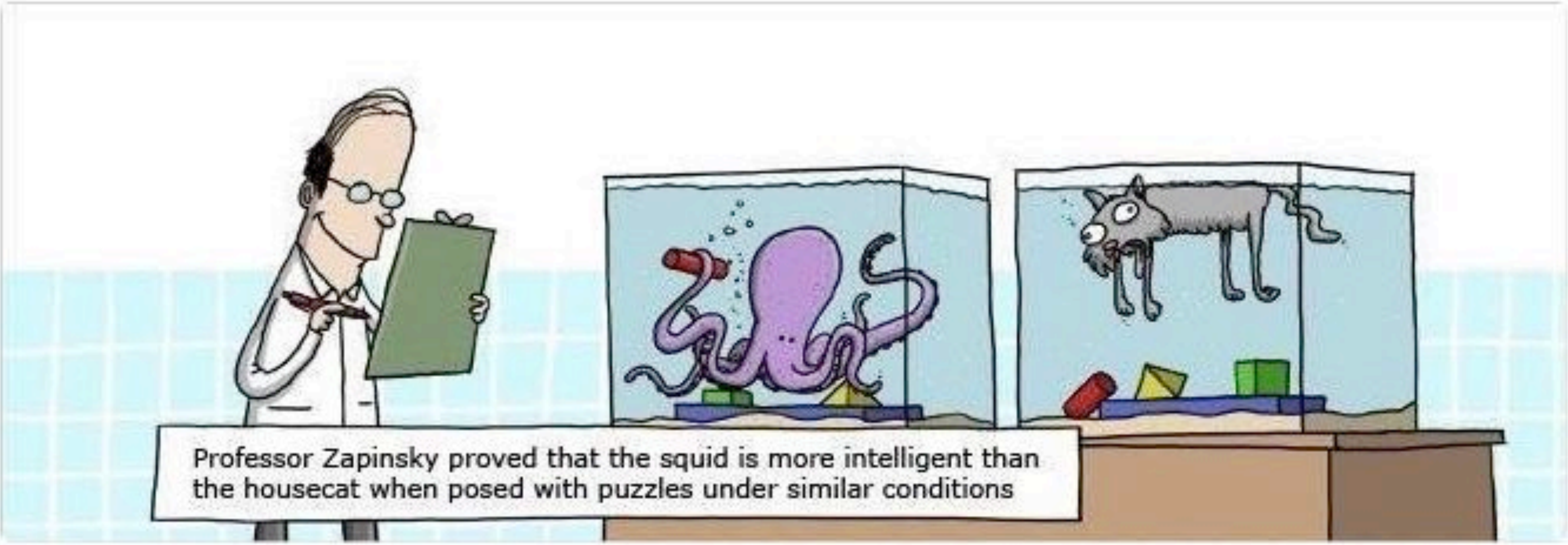
elastic

DFS Query Then Fetch

Distributed Frequency Search

```
GET starwars/_search?search_type=dfs_query_then_fetch
{
  "query": {
    "match": {
      "word": "Luke"
    }
  }
}
```

```
{
  "_index": "starwars",
  "_type": "_doc",
  "_id": "0fVdy2IBkmPuaFRg659y",
  "_score": 1.5367417,
  "_routing": "0",
  "_source": {
    "word": "Luke"
  }
},
{
  "_index": "starwars",
  "_type": "_doc",
  "_id": "2_Vdy2IBkmPuaFRg659y",
  "_score": 1.5367417,
  "_routing": "0",
  "_source": {
    "word": "Luke"
  }
},
{
  "_index": "starwars",
  "_type": "_doc",
  "_id": "3PVdy2IBkmPuaFRg659y",
  "_score": 1.5367417,
  "_routing": "0",
  "_source": {
    "word": "Luke"
  }
},
...
```


Professor Zapinsky proved that the squid is more intelligent than the housecat when posed with puzzles under similar conditions

Often Non or Minor Issue

Lots of documents

Even distribution

Don’t use `dfs_query_then_fetch` in production. It really isn’t required.

— https://www.elastic.co/guide/en/elasticsearch/guide/current/relevance-is-broken.html

Single Shard

Default in 7.0

Simon Says

Use a single shard until it blows up

```
PUT starwars/_settings
{
  "settings": {
    "index.blocks.write": true
  }
}
```

```
POST starwars/_shrink/starwars_single
{
  "settings": {
    "number_of_shards": 1,
    "number_of_replicas": 0
  }
}
```

```
GET starwars_single/_search
{
  "query": {
    "match": {
      "word": "Luke"
    }
  },
  "_source": false
}
```

```
{
  "_index": "starwars_single",
  "_type": "_doc",
  "_id": "0fVdy2IBkmPuaFRg659y",
  "_score": 1.5367417,
  "_routing": "0"
},
{
  "_index": "starwars_single",
  "_type": "_doc",
  "_id": "2_Vdy2IBkmPuaFRg659y",
  "_score": 1.5367417,
  "_routing": "0"
},
{
  "_index": "starwars_single",
  "_type": "_doc",
  "_id": "3PVdy2IBkmPuaFRg659y",
  "_score": 1.5367417,
  "_routing": "0"
},
GET starwars_single/_search
{
  "aggs": {
    "most_common": {
      "terms": {
        "field": "word.keyword",
        "size": 1
      }
    }
  },
  "size": 0
}
```

```
{
  "took": 1,
  "timed_out": false,
  "_shards": {
    "total": 1,
    "successful": 1,
    "skipped": 0,
    "failed": 0
  },
  "hits": {
    "total": 288,
    "max_score": 0,
    "hits": []
  },
  "aggregations": {
    "most_common": {
      "doc_count_error_upper_bound": 0,
      "sum_other_doc_count": 224,
      "buckets": [
        {
          "key": "Luke",
          "doc_count": 64
        }
      ]
    }
  }
}
```

Conclusion

Tradeoffs…

Consistent Available

Partition Tolerant

Fast Accurate Big

Questions?

Philipp Krenn @xeraa

PS: Stickers

elastic