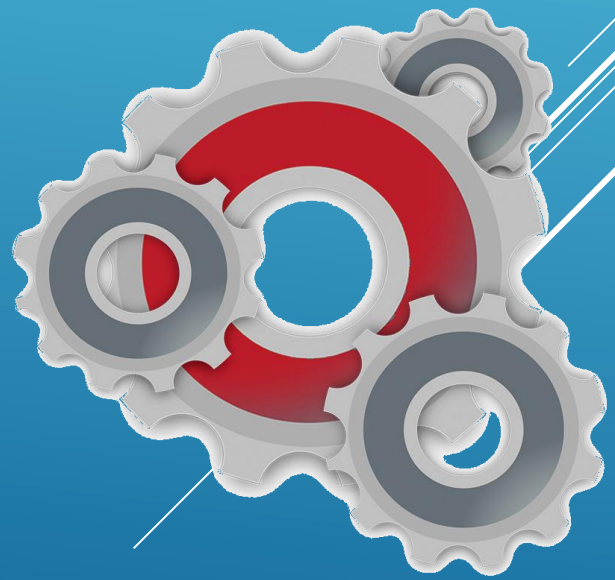
ELASTIC DELPHI 2

Optimizing Storage and Search

- 20 years programming with relational databases
  - MSSQL, Interbase/Firebird, MySQL, Oracle
- Specialise in manipulating large data sets
- 30 months working with Elasticsearch
  - Developed Elastic Explorer

NIGEL TAVENDALE

www.allthingssyslog.com



ALL THINGS SYSLOG

CODE SAMPLES AVAILABLE AT:

https://github.com/ntavendale/ElasticDelphi

ELASTIC EXPLORER AVAILABLE FROM:

http://www.elasticexplorer.com/

NIGEL TAVENDALE

nigel.tavendale@allthingssyslog.com

ALL THINGS SYSLOG

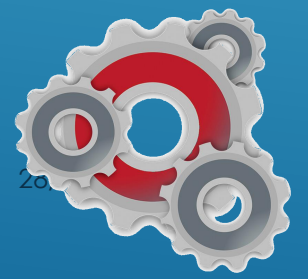
ADVANCED TOPICS

MAPPINGS

Define types to mange the amount of storage space required and influence searching.

ROUTES

Search Optimization



MAPPINGS

ES is *not* schema-less. If you don't have a predefined mapping or template it will derive one and add to it as you post data.

Once a decision is made about a field's datatype – you are stuck with it.

Can be applied to an index after it is created.

Can be defined in Templates that are used in index creation.

Define document fields and their types.

Single default mapping per index as of version 7.

Save space by defining some fields as keywords and others as text

INDEXES

```
{
    “_id”:”doc01”, “message”: “The car is blue”
}
{
     “_id”:”doc02”, “message”: “The car is large”
}
{
     “_id”:”doc03”, “message”: “The bicycle is blue”
}
```

doc03	bicycle	blue
doc01	car	blue
doc02	car	large

INVERTED INDEXES - TEXT

```
{
    “_id”:”doc01”, “message”: “The car is blue”
}
{
     “_id”:”doc02”, “message”: “The car is large”
}
{
     “_id”:”doc03”, “message”: “The bicycle is blue”
}
```

bicycle			doc03
blue	doc01		doc03
car	doc01	doc02	
large		doc02	

INVERTED INDEXES – KEYWORD

```
{
    “_id”:”doc01”, “message”: “The car is blue”
}
{
     “_id”:”doc02”, “message”: “The car is large”
}
{
     “_id”:”doc03”, “message”: “The bicycle is blue”
}
```

Keywords only searchable by their exact value. Every field value in each document gets it’s own index entry unless it is *exactly* the same as another.

bicycle blue	doc03
car blue	doc01
car large	doc02

INDEXING SAMPLE DATA

```
{
  "type":"BSD",
  "facility":"UserLevel",
  "severity":"Debug",
  "timeStamp":"2019-06-28T07:00:00.000Z",
  "host":"192.168.8.158",
  "process":"SysLogSimSvc",
  "processId":2559,
  "logMessage":"161.200.1.6: a warning has been generated for application http"
}
```

Index logMessage field using text, keyword & text + keyword (default).

KEYWORD vs. TEXT

With no mapping defined default is Keyword + Text. Using text saves disk space – especially if there is a lot of repetition in data.

Results for test data set of 5000 messages:

Keyword + Text	1353 KB
Keyword Only	1157 KB
Text Only	737 KB

KEYWORDS

Use for fields you would typically filter on – statuses, place names (cities, countries), groups. Small number of words, lots of repetition.

TEXT

Use for fields where you index everything. Message bodies, product descriptions, user reviews.

TEMPLATES

Define mappings for an index prior to the index creation.

Define fields and their data types.

Define indexes mappings created on using wildcards.

log-* : log-windows-event

log-syslog

TEMPLATES

Dynamic property - determines what happens if document doesn't match exactly

true	New fields are added to the mapping and indexed
false	New fields not added to the mapping or indexed. Document is still indexed
strict	Document rejected. Exception thrown internally. 200 still sent back

IMHO – don't use strict, unless there is no other option.

ROUTES

Customize the distribution of documents among shards.

Route names arbitrary.

```
{"index":{"_index":"routed-msg", "routing":"Critical"}}
{"type":"BSD","facility":"MailSystem","severity":"Critical","timeStamp":"1899-12-
28T07:00:00.000Z","host":"192.168.8.2","process":"SysLogSimSvc","processId":
2559,"logMessage":"Reconnaissance activity detected 111.148.118.9:40083 ->
161.200.1.9:443 TCP"}
```

Speeds up searches if shards distributed across multiple nodes in a cluster.

```
http://192.168.85.122:9200/routed-msg/_search?q=*:*&size=10&routing=Critical
```

WATCH OUT!

Document IDs only unique per route!

If you use routing you can get duplicates. Do not rely on uniqueness!