Workshop: Elasticsearch & Kibana

Alessandro Fortunati, Nicola Pagni, Andrea Tosti

Seacom Srl

Presentazioni

Il programma

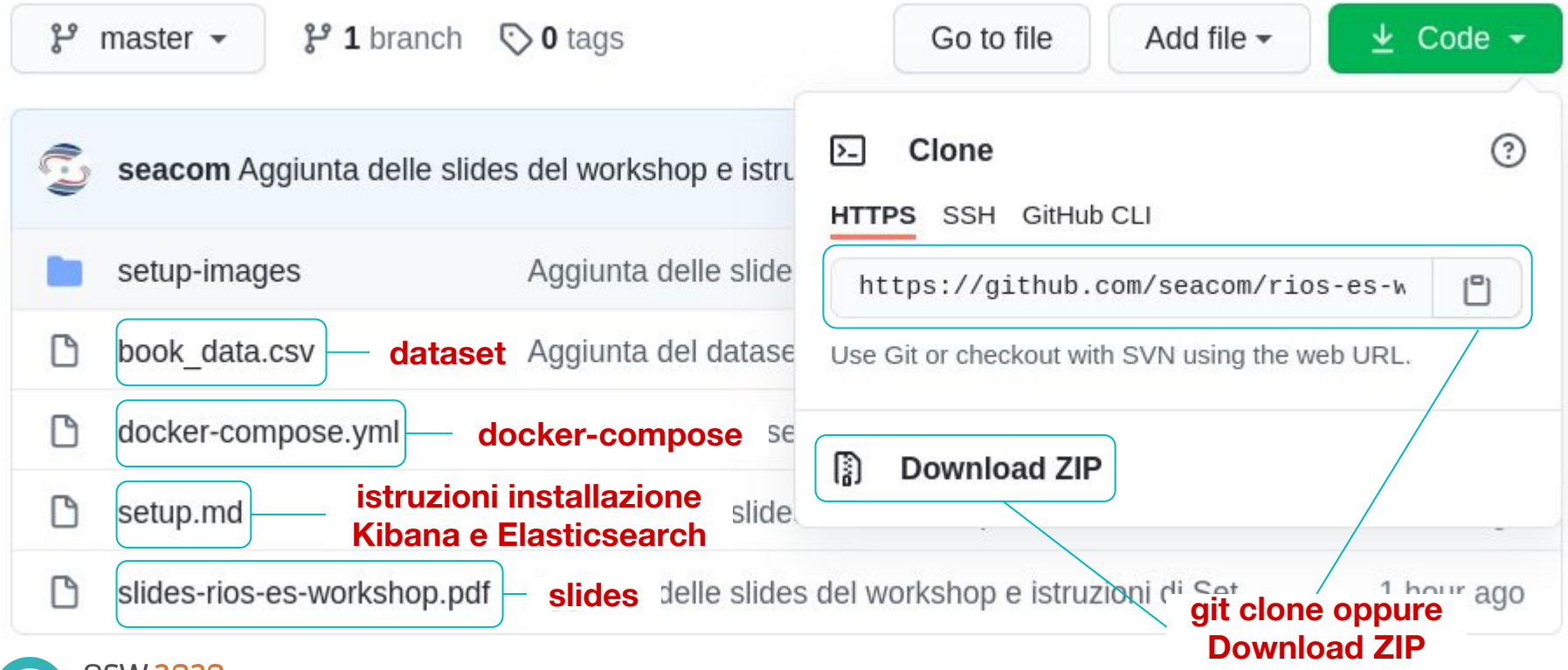
- Introduzione al Workshop
- Overview su Elasticsearch & Kibana
- LAB
  - Indicizzazione
  - Modellizzazione
  - Ricerca & Analisi
  - Kibana

(Alla fine di ogni laboratorio verrà mostrata la soluzione)

Prepariamo l’ambiente

Setup: repository

https://github.com/seacom/rios-es-workshop-2020



Setup: download di Elasticsearch e Kibana

Linux	https://artifacts.elastic.co/downloads/elasticsearch/elasticsearch-7.10.0-linux-x86_64.tar.gz https://artifacts.elastic.co/downloads/kibana/kibana-7.10.0-linux-x86_64.tar.gz
Mac	https://artifacts.elastic.co/downloads/elasticsearch/elasticsearch-7.10.0-darwin-x86_64.tar.gz https://artifacts.elastic.co/downloads/kibana/kibana-7.10.0-darwin-x86_64.tar.gz
Windows	https://artifacts.elastic.co/downloads/elasticsearch/elasticsearch-7.10.0-windows-x86_64.zip https://artifacts.elastic.co/downloads/kibana/kibana-7.10.0-windows-x86_64.zip
Docker	docker pull docker.elastic.co/elasticsearch/elasticsearch:7.10.0 docker pull docker.elastic.co/kibana/kibana:7.10.0

Setup: estrazione

- Per semplicità estrarremo gli archivi all'interno della cartella **rios-es-workshop-2020**
- Per velocizzare l'estrazione degli archivi su Windows è consigliato l'utilizzo di **7-Zip**

Linux	tar -xzvf elasticsearch-7.10.0-linux-x86_64.tar.gz tar -xzvf kibana-7.10.0-linux-x86_64.tar.gz
Mac	tar -xzvf elasticsearch-7.10.0-darwin-x86_64.tar.gz tar -xzvf kibana-7.10.0-darwin-x86_64.tar.gz
Windows	Estrarre il contenuto di: elasticsearch-7.10.0-windows-x86_64.zip kibana-7.10.0-windows-x86_64.zip
Docker	Nessuna operazione necessaria

Setup: avvio di Elasticsearch e Kibana

- Su Windows è consigliato l'utilizzo del prompt dei comandi eseguito come Amministratore
- Aprire quindi due terminali / prompt, uno per Elasticsearch, l'altro per Kibana
- Dirigersi con un browser su http://localhost:9200 per vedere se Elasticsearch è raggiungibile

(Step 1) Docker su Linux/Mac/Windows	sudo sysctl -w vm.max_map_count=262144 (ulteriori info)
(Step 2) Linux	cd elasticsearch-7.10.0/ && ./bin/elasticsearch cd kibana-7.10.0-linux-x86_64/ && ./bin/kibana
(Step 2) Mac	cd elasticsearch-7.10.0/ && ./bin/elasticsearch cd kibana-7.10.0-darwin-x86_64/ && ./bin/kibana
(Step 2) Windows	cd elasticsearch-7.10.0/ && .\bin\elasticsearch.bat cd kibana-7.10.0-windows-x86_64 && .\bin\kibana.bat
(Step 2) Docker	cd rios-es-workshop-2020 && docker-compose up -d

Setup: Primo avvio avvenuto con successo?

- Dirigersi con un browser su http://localhost:5601 per vedere se Kibana è raggiungibile
- Se non ottenete errori, congratulazioni!
- Se ottenete un messaggio con scritto "Kibana server is not ready yet", controllate che sul terminale / prompt dei comandi di Kibana compaiano i messaggi come il seguente (per consultare i log di Kibana con docker-compose: docker-compose logs kibana)

```
log   [11:56:30.978] [info][listening] Server running at http://localhost:5601
log   [11:56:34.526] [info][server][Kibana][http] http server running at http://localhost:5601
```

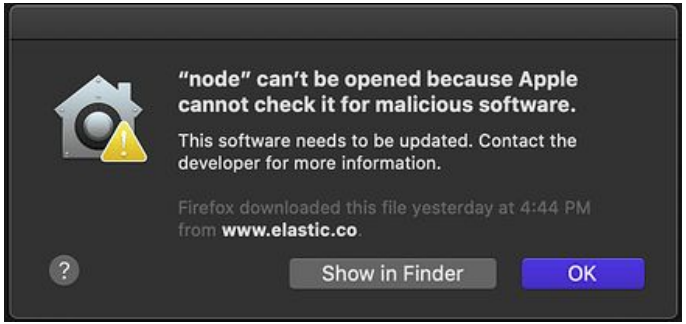
- Se sul terminale / prompt di Kibana ottenete un messaggio di questo tipo:

```
log   [11:48:41.120] [warning][savedobjects-service] Another Kibana instance appears to be migrat
ing the index. Waiting for that migration to complete. If no other Kibana instance is attempting mi
grations, you can get past this message by deleting index .kibana_task_manager_1 and restarting Kib
ana.
```

allora interrompete l'esecuzione di Elasticsearch e eliminate la cartella "data" presente nella cartella elasticsearch-7.10.0 (**Attenzione:** l'eliminazione della cartella "data" comporta l'eliminazione di tutti i dati presenti su Elasticsearch)

Setup: Primo avvio avvenuto con successo?

- Se avete problemi ad avviare Kibana su Mac OS dove *node* non può essere aperto per motivi di sicurezza



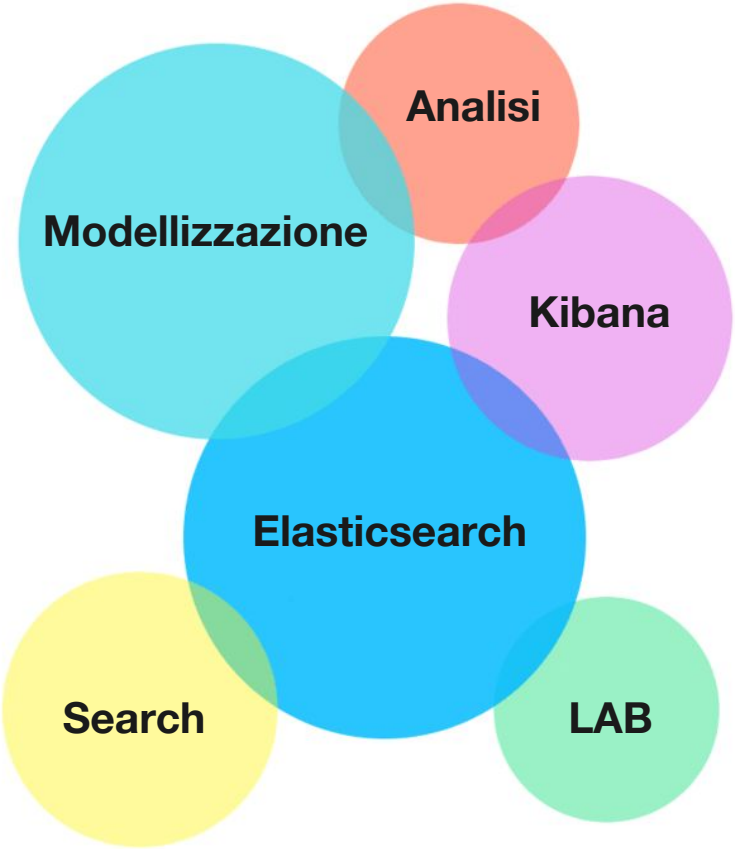
allora avete modi diversi di risolvere:

1- System Preferences > Security and Privacy > Developer Tools. Modificare per permettere al terminale di eseguire software che non soddisfa le preferenze di sicurezza del sistema.

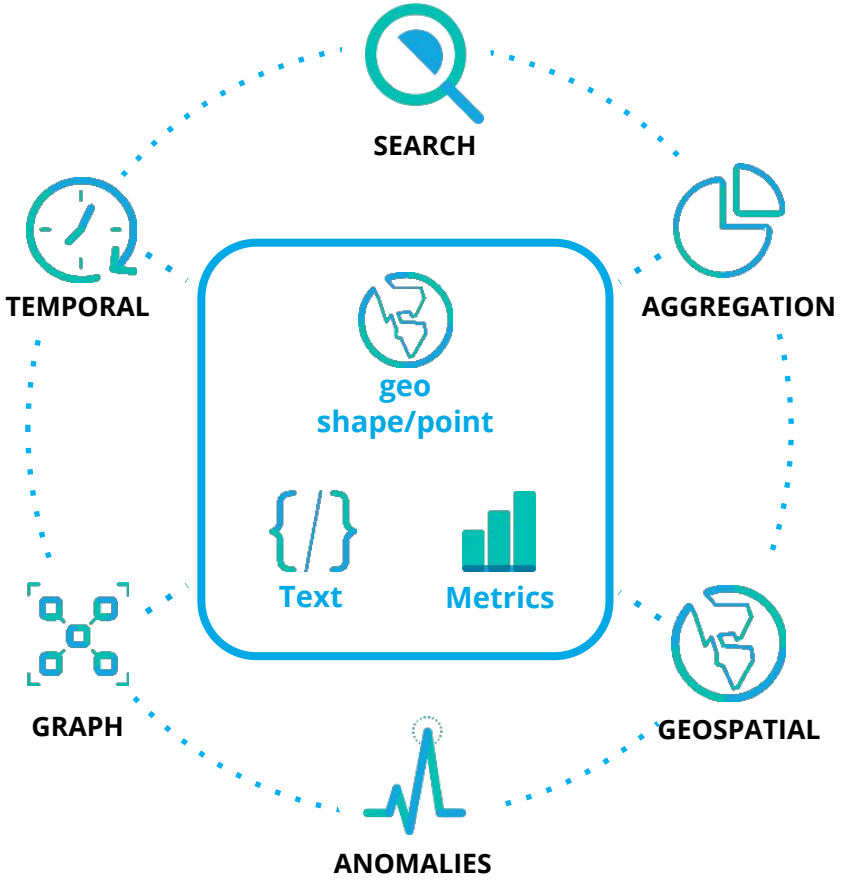
2- Fare click su "Show in Finder", eseguire *node*, dare i permessi di sicurezza a *node*, dopodiché provare a riavviare Kibana

Overview

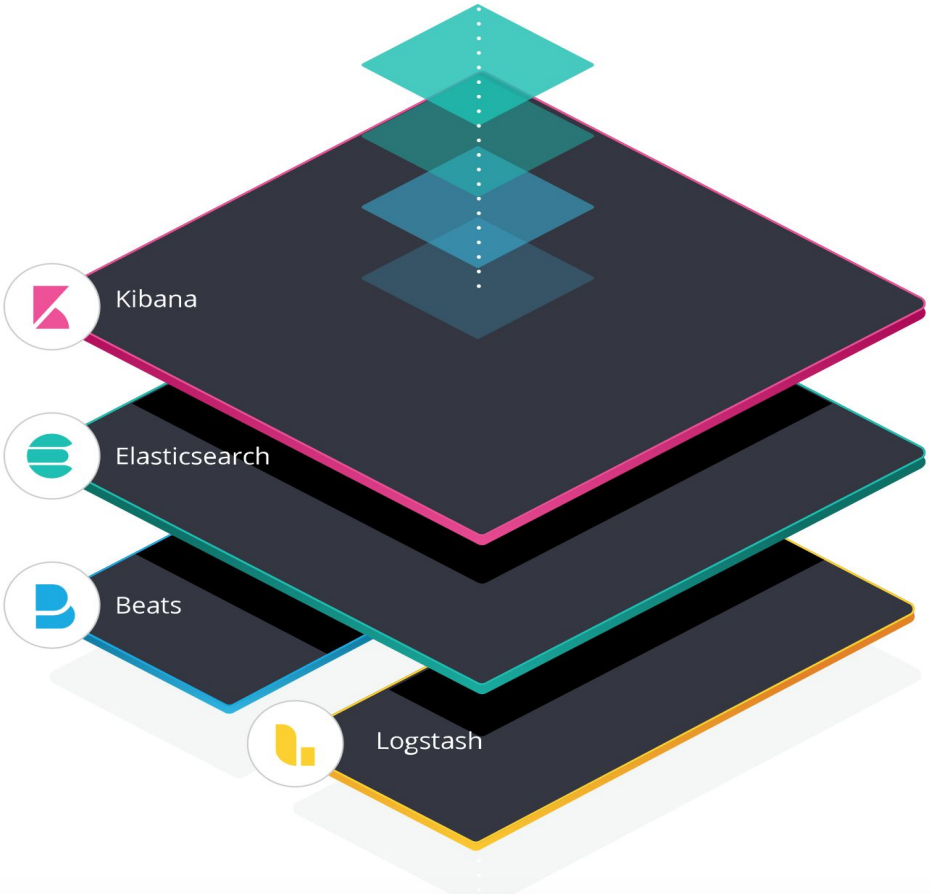
I temi del workshop



Che cos'è Elasticsearch?



Elastic Stack



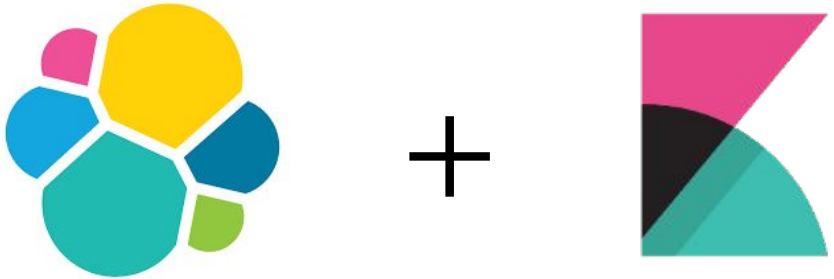
Elastic Stack subscriptions

The Elastic Stack — Elasticsearch, Kibana, Beats, and Logstash — powers a variety of use cases. And we have flexible plans to help you get the most out of your on-prem subscriptions.

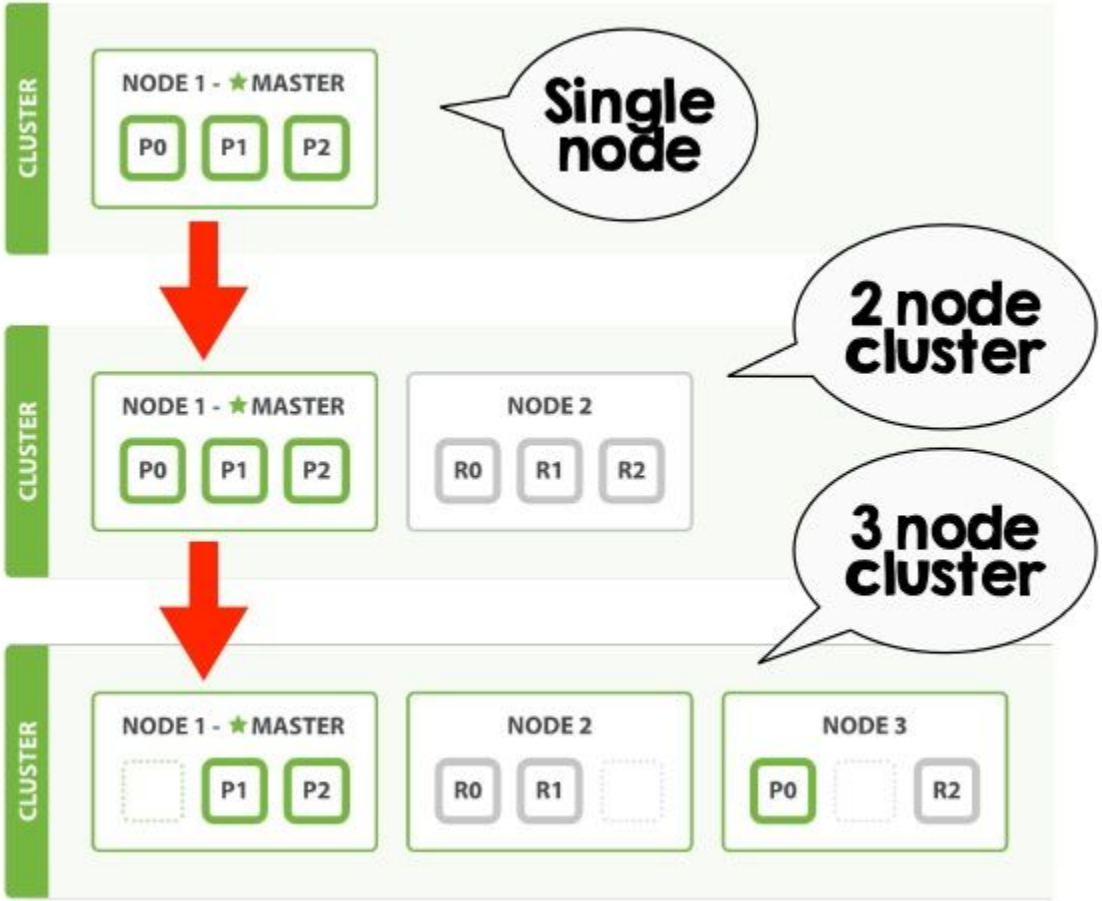
Our **resource-based pricing philosophy** is simple: You only pay for the data you use, at any scale, for every use case.

FREE				
Open Source	**Basic**	**Gold**	**Platinum**	**Enterprise**
Apache 2.0: Now and always.	The forever-free plan.	More features. Dedicated support.	Advanced functionality. Around the clock support.	Stack orchestration and endpoint protection by default.
Feature highlights include:	**Everything in Open Source plus:**	**Everything in Basic plus:**	**Everything in Gold plus:**	**Everything in Platinum plus:**
✓ Clustering & high availability ✓ Powerful search and analysis	✓ Core Elastic Stack security features ✓ Capabilities such as Elastic APM, Security, App Search, Workplace	✓ Reporting ✓ Kibana third-party alerting actions[3] ✓ Watcher	✓ Advanced Elastic Stack security features ✓ Machine learning	✓ Access to Elastic Endgame[2] ✓ Access to ECE & ECK orchestration features

Noi utilizzeremo...



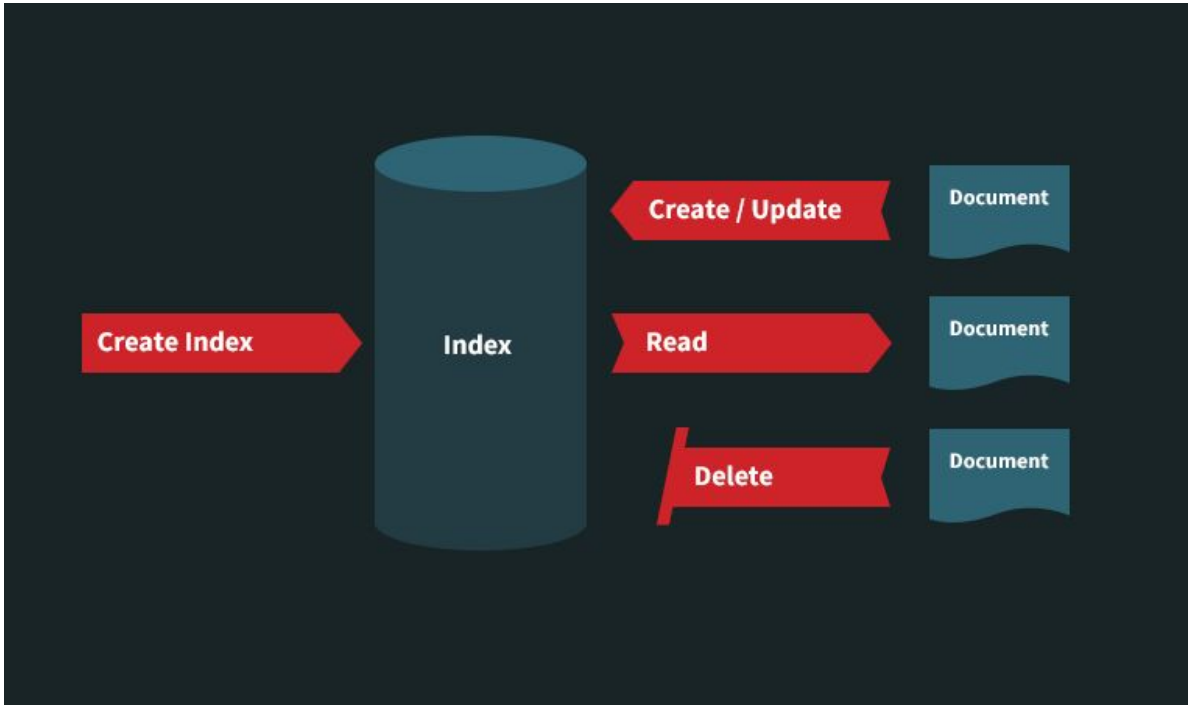
L’ architettura



RDBMS vs Elasticsearch

RDBMS	ES
	
Table	Type
Row	Document
Column	Field

CRUD


Create Index
Index
Create / Update
Read
Delete
Document
Document
Document

Mapping - caratteristiche

- Dynamic Mapping *(default).*
- Un mapping può essere esteso ma non ri-definito.
- Può contenere al più **1000** field di primo livello.
- Può contenere field la cui profondità (inner object) è al massimo di **20**.
- E' sempre consigliato definire prima il mapping e poi iniziare a caricare i dati.
- **Index Template**, **Alias**, **ILM** sono fondamentali per la gestione di un indice.

Come si crea un Mapping

```
PUT /<INDEX>/_mapping
{
  "properties": {
    "<FIELD>": {
      "type": "<TYPE>"
    }
  }
}
```

Vedi documentazione

Come si aggiorna un Mapping

```
PUT /<INDEX>
{
  "mappings": {
    "properties": {
      "<NEW_FIELD>": {
            "type": "text"
      }
    }
  }
}
```

Come si indicizza un nuovo documento

```
POST <INDEX>/_doc/
{
  "@timestamp": "2099-11-15T13:12:00",
  "message": "GET /search HTTP/1.1 200 1070000",
  "user": {
    "id": "kimchy"
  }
}
```

Vedi documentazione

Come si cancella un documento

```
DELETE /<INDEX>/_doc/<ID>
```

Vedi documentazione

Come si aggiorna un documento

```
POST <INDEX>/_update/<ID>
{
  "doc": {
    "name": "new_name"
  }
}
```

Vedi documentazione

Come si copiano i documenti da un indice A verso un indice B

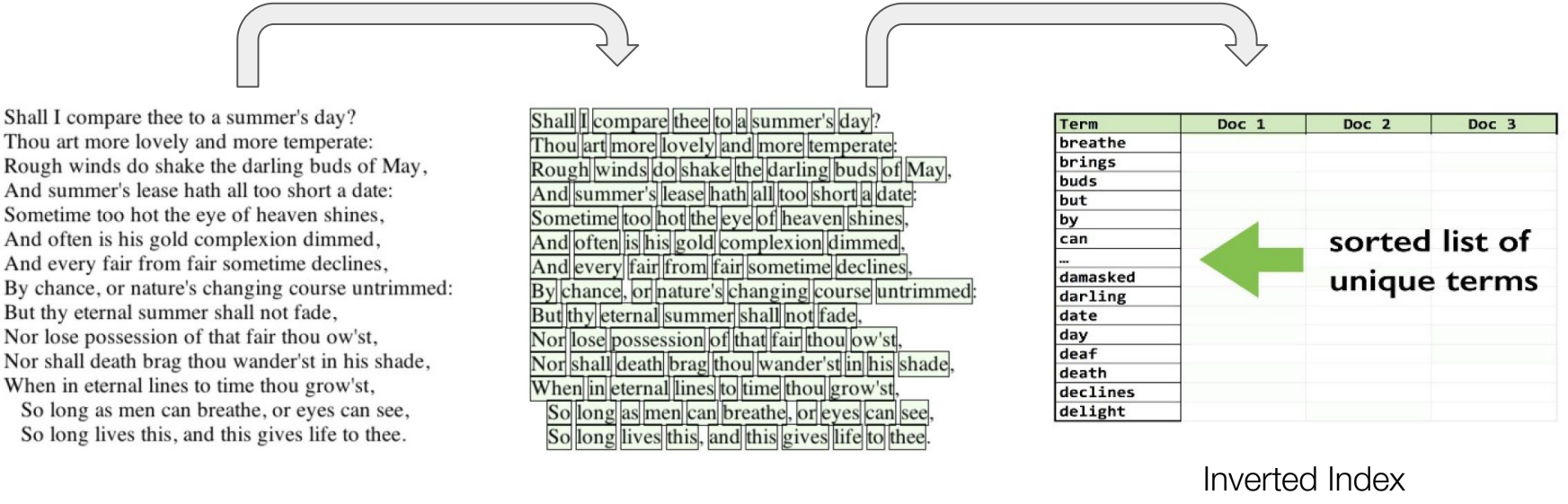
```
POST /_reindex
{
  "source": {
    "index": "A"
  },
  "dest": {
    "index": "B"
  }
}
```

Vedi documentazione

Indicizzazione e Storicizzazione ??

- Sono 2 processi diversi, entrambi vengono coinvolti durante l'inserimento del dato
- Hanno 2 obiettivi differenti:
  - l'indicizzazione è indispensabile per ricercare un dato
  - La storicizzazione è utile per vedere le informazioni restituite *(dato raw)*

Indicizzazione del testo

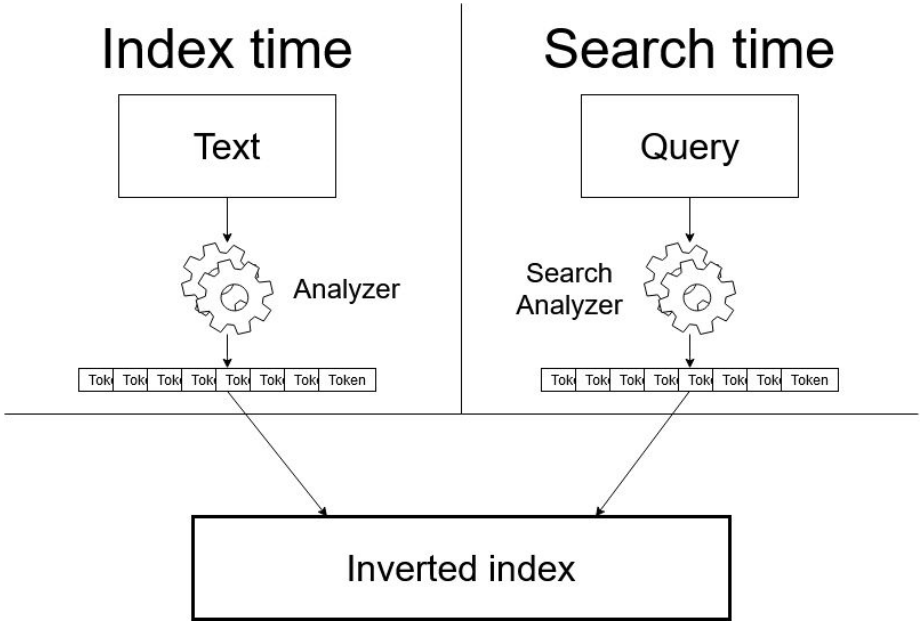


Inverted Index

Processo di analisi

Vedi documentazione

Applicazione del processo di analisi



Esempio di Ricerca

1 - L'utente scrive SuMMer nella barra di ricerca

2 - Supponiamo di avere nell'indice target un documento con il seguente testo:

Shall I compare thee to a summer's day?
Thou art more lovely and more temperate:
Rough winds do shake the darling buds of May,
And summer's lease hath all too short a date:
Sometime too hot the eye of heaven shines,
And often is his gold complexion dimmed,
And every fair from fair sometime declines,
By chance, or nature's changing course untrimmed:
But thy eternal summer shall not fade,
Nor lose possession of that fair thou ow'st,
Nor shall death brag thou wander'st in his shade,
When in eternal lines to time thou grow'st,
So long as men can breathe, or eyes can see,
So long lives this, and this gives life to thee.

3 - La parola summer's viene *tokenizzata* e successivamente *filtrata* con l'ausilio del token filter di tipo **Elision**.

4 - Viene fatto un confronto tra:
SuMMer *(summer)* === summer's *(summer)*

*Nelle ricerche viene dato spesso per scontato il poter cercare senza tenere in considerazione minuscole e maiuscole (filtro **lowercase**)*

QueryDSL

- Il framework per costruire ricerche.
- Sono delle configurazioni JSON.
- Utilizzate da tutte le librerie.
- Possiamo definire quanti criteri vogliamo *(ciascuno con una sua tipologia di query)* e combinarli tra loro in *AND,OR, NOT*.

```
GET <INDEX>/_search
{
  "query": {
    "bool": {
      "must": [
        { "match": { "title":   "Search"        }},
        { "match": { "content": "Elasticsearch" }}
      ],
      "filter": [
        { "term":  { "status": "published" }},
        { "range": { "publish_date": { "gte": "2015-01-01" }}}
      ]
    }
  }
}
```

Vedi documentazione

QueryDSL

- took
- _shards
- hits


```
{
  "took" : 23,
  "timed_out" : false,
  "_shards" : {
    "total" : 1,
    "successful" : 1,
    "skipped" : 0,
    "failed" : 0
  },
  "hits" : {
    "total" : {
      "value" : 10000,
      "relation" : "gte"
    },
    "max_score" : 1.0,
    "hits" : [
      {
        "_index" : "flights",
        "_type" : "_doc",
        "_id" : "IYFm03UBE1qXmdWSSkH9",
        "_score" : 1.0,
        "_source" : {
                ...
        }
      },
      {
        "_index" : "flights",
        "_type" : "_doc",
        "_id" : "IoFm03UBE1qXmdWSSkH9",
        "_score" : 1.0,
        "_source" : {
                ...
        }
      }
    ]
  }
}
```

Aggregations

- Il framework per fare analisi.
- Utili per costruire dashboard.
- Utili per costruire filtri di ricerca.

Esempio

Dato un dataset di libri (doc = libro).

Immaginare di dover cercare gli autori che hanno scritto più libri.

- *Implementarlo tramite query* ✗
- *Implementarlo tramite aggs* ✓

```
GET <INDEX>/_search
{
  "aggs": {
    "my-agg-name": {
      "terms": {
        "field": "<FIELD>"
      }
    }
  }
}
```

Vedi documentazione

Aggregations

Esempio

Dato un dataset di libri (doc = libro).

Immaginare di dover cercare gli autori che hanno scritto più libri.


```
"aggregations" : {
  "author-aggs" : {
    "doc_count_error_upper_bound" : 0,
    "sum_other_doc_count" : 0,
    "buckets" : [
      {
        "key" : "Dan Brown",
        "doc_count" : 24
      },
      {
        "key" : "J. K. Rowling",
        "doc_count" : 13
      },
      {
        "key" : "Oriana Fallaci",
        "doc_count" : 10
      },
      {
        "key" : "Alessandro Baricco",
        "doc_count" : 7
      }
    ]
  }
}
```

Ingest Pipeline

- Feature nativa di Elasticsearch.
- Usate per manipolare/arricchire il documento prima che questo sia indicizzato.
- Definite a livello di cluster come configurazioni JSON.
- Utilizzate durante indicizzazioni, aggiornamenti, re-indicizzazioni.
- Possiamo definire una pipeline di **processors** come **Set**, **Drop**, **Split**, **Script**, etc...

Vedi documentazione

Definire una Ingest Pipeline

```
PUT _ingest/pipeline/<NAME>
{
  "description" : "...",
  "processors" : [
    {
      "set" : {
        "field": <FIELD>,
        "value": <VALUE>
      }
    }
  ]
}
```

Usare una Ingest Pipeline

```
POST /_reindex
{
  "source": {
    "index": "A"
  },
  "dest": {
    "index": "B",
    "pipeline": <NAME>
  }
}
```

Temi trattati

- Overview architettura di base di Elasticsearch
- Come si crea/aggiorna/cancella un indice
- Come si (re)-indicizzano/cancellano/aggiornano i documenti
- Analisi del testo e come viene applicato
- Search & Aggregations
- Ingest Pipeline

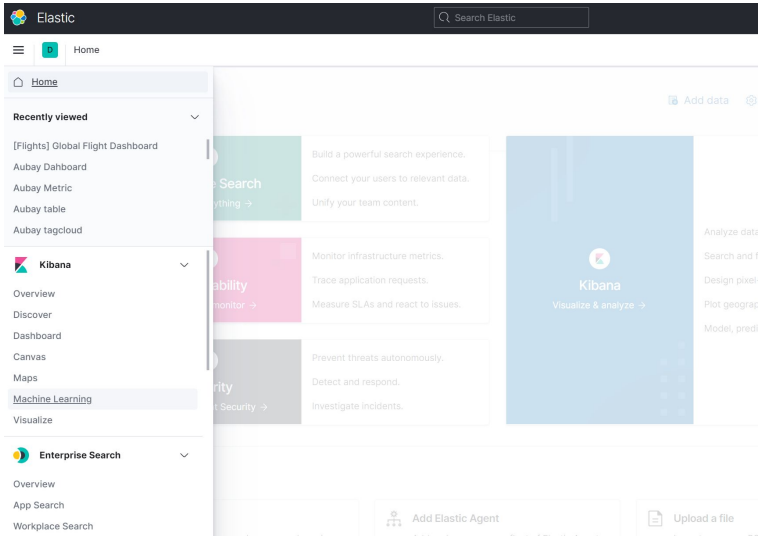
LAB !

LAB - Indicizzazione

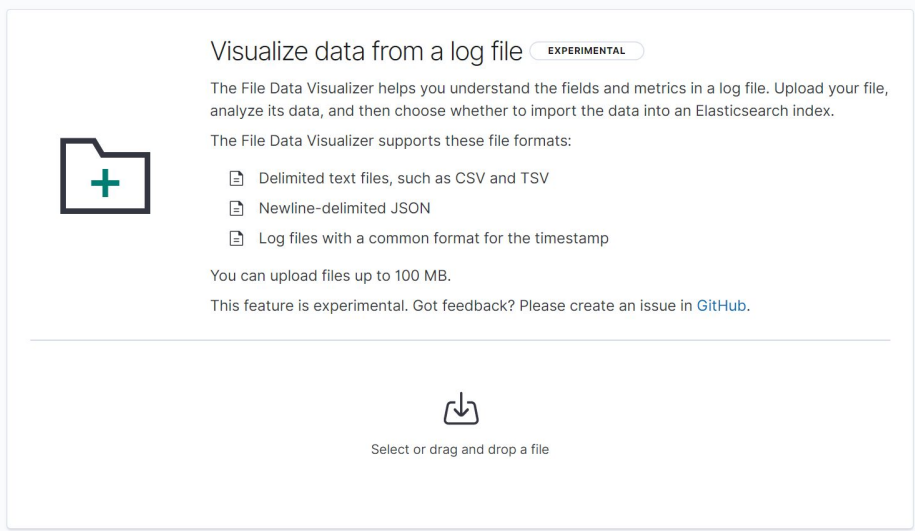
- Avviamo Elasticsearch (localhost:9200) e kibana (localhost:5601)
- Spostiamoci su Kibana
- Prendiamo il dataset e indicizziamone il contenuto su Elasticsearch

LAB - Indicizzazione

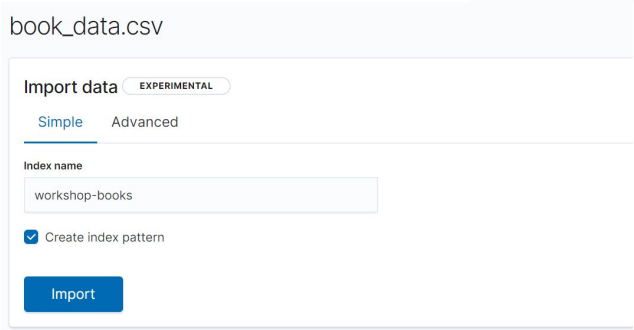
1



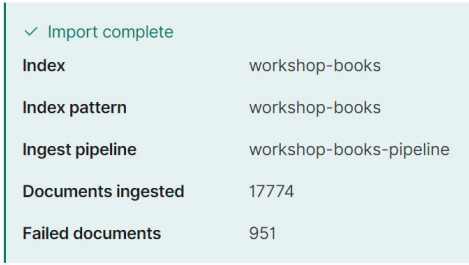
2



3



4



LAB - Indicizzazione

- Da Kibana, spostarsi nel modulo Discover per verificare quanto indicizzato
- Dal Discover, creare una vista e salvarla con i seguenti fields:
    - book_title , book_edition, book_authors, genres
- ...possiamo notare come i fields genres e book_authors non sono nella forma ottimale, ossia una lista di stringhe

LAB - Modellizzazione

- Da Kibana, spostarsi nel modulo Dev Tools
- Recuperare il mapping dell'indice books creato con **GET workshop-books/_mapping**

  *Nota: Per vedere tutti gli indici nel cluster eseguire invece **GET _cat/indices***

- Creare un indice con lo stesso mapping, ad esempio **workshop-books-2**

LAB - Modellizzazione

- Creare una Ingest Pipeline tale per cui i campi genres e book_authors vengono storicizzati e indicizzati nel modo corretto. *(nota: utilizzare lo **split** processor Vedi documentazione)*
- Simulare la pipeline Vedi documentazione
- Se la simulazione ha dato esito positivo, utilizzare la pipeline nell'operazione di re-indicizzazione dall'indice **workshop-books** verso l'indice **workshop-books-2**
- Eseguire una GET workshop-books-2/_search per verificare l'indicizzazione

LAB - Ricerca

- Dato l'indice **workshop-books-2**, immaginiamo di dover implementare delle ricerche a partire da ipotetiche parole che potrebbe digitare l'utente sulla barra di ricerca del nostro sito immaginario workshop-book-store.it

LAB - Ricerca

1. L'utente digita "**harry PoTTer azkaBAN**". Restituire SOLO i libri il cui titolo contiene tutte e 3 le parole. Vedi documentazione
2. Stessa ricerca "**hary PoTer zkaBAN**"; ma stavolta notiamo che l'utente ha commesso degli errori di digitazione. Restituire SOLO i libri che contengono tutte e 3 le parole nel titolo o nella sua descrizione, nonostante gli errori. Vedi documentazione
3. L'utente seleziona il filtro "**Agatha Christie**" come autore. Il nostro sito vuole dare maggiore visibilità ai generi "**Thriller**". *Suggerimento: utilizzare must+should*

LAB - Ricerca

1. L'utente digita "**Killer**". Implementare la ricerca in modo che siano coinvolti i soli fields book_title e book_desc *(basta che la parola sia presente in almeno uno dei due).*
2. Evolvere la query precedente restituendo, per ciascun documento, quali sono stati i campi matchati. *Suggerimento: utilizzare le named queries*

LAB - Analisi

1. Capire quali sono gli autori più presenti nel dataset. Funziona???

   Suggerimento: ...verificare il datatype di quel campo, forse è necessaria una reindex!

   Suggerimento(2): ...verificare anche il datatype del campo genres

2. Evolvere la richiesta precedente, effettuando l'analisi solo per i libri che hanno un rating di almeno **4.6**.

   Suggerimento: utilizzare la query range

LAB - Analisi con Kibana

- Creare una dashboard con le seguenti caratteristiche:
  - Deve essere presente un pie chart che mostra i 10 autori più presenti.
  - Deve essere presente un contatore con il numero di libri.
  - Deve essere presente un word cloud che mostra le 10 categorie più presenti.
  - Aggiungere anche la ricerca salvata nella discovery

OSW 2020

opensourceweek