INTRODUZIONE A

Solr

Antonio Scaiella
17 Gennaio 2023
Corso di Information Retrieval A.A. 2022/2023

AGENDA

- Apache SolR
  - Introduzione ad Apache SolR
    - Cos’è Apache SolR?
    - Caratteristiche
    - Architettura
    - Core e Collection
  - Installazione
    - Download
    - GUI
  - Creazione core
  - Indexing and searching

Antonio Scaiella
Gennaio 2023

INTRODUZIONE

Apache Solr è una piattaforma di ricerca open source costruita sulla libreria Java open source **Lucene**

Apache Lucene funziona sulla base di un **indice inverso** per archiviare documenti (dati) e fornisce funzionalità di ricerca e indicizzazione tramite Java API.

> inverte una chiave incentrata sulla pagina (pagina->parole) in una chiave incentrata sulla parola chiave (parola->pagine) per un recupero più rapido dei risultati di ricerca.

Le comunicazioni in Solr vengono eseguite tramite client REST
wget, curl, POSTMAN di Chrome o client in Java,Python ecc

Offre i dati in svariati formati: JSON,XML,CSV ..

Antonio Scaiella
Gennaio 2023

INTRODUZIONE

Features of Solr

- Advanced Full-Text Search.
- Optimized for high traffic. (Affidabilità)
- Near Real-Time Indexing.
- Easy sharding & replication (Scalabilità & Affidabilità)
- Zookeper (Load Balancing)
- Atomic updates with optimized concurrency.(versioning)
- Supports pagination, sorting, facets.
- Supports spell checking, text highlighting and auto-suggestion.

Antonio Scaiella
Gennaio 2023

INTRODUZIONE

- Tutto in Solr è memorizzato come un **documento** il quale è un insieme di **fields**.
- La raccolta di documenti insieme forma un **indice**.
- Il campo(field) non è altro che una rappresentazione **chiave-valore** dei dati.
- La definizione di un campo, il nome, tipo di campo, analizzatori, tali informazioni sono archiviate nel file schema.xml e tutte le configurazioni di Solr sono archiviate in solrconfig.xml

Antonio Scaiella
Gennaio 2023

INTRODUZIONE

Definizione di un Field in Solr

- **name**: Il nome del field
- **type**: tipo del field e.g. **text**,**string** long,int,date
- **indexed**: il field deve far parte dell'indice?
- **stored**: ll valore originale deve essere memorizzato?
- **multiValued**: è possibile assegnare più valori al campo?

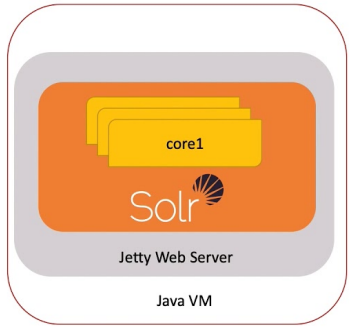
Antonio Scaiella
Gennaio 2023

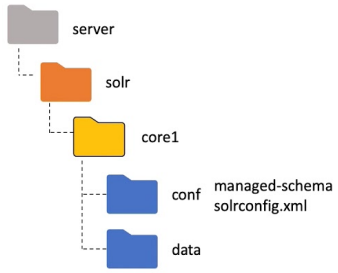
INTRODUZIONE

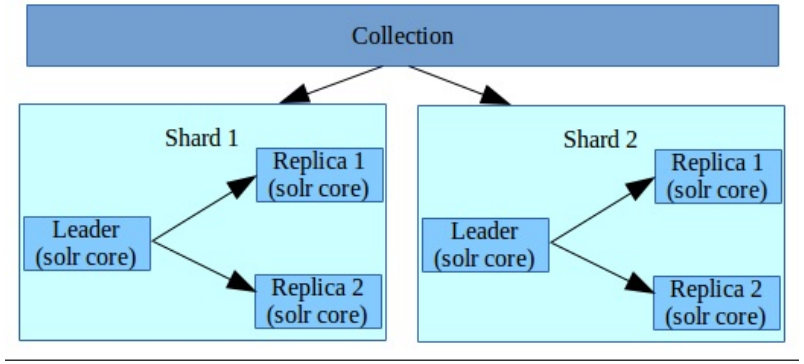
Core & collection

La **Collection** è un indice logico distribuito su più server. Il **Core** è quella parte del server che esegue una **collection**. Nella ricerca non distribuita, il server singolo che esegue Solr può avere più Collection e ciascuna di queste è anche un **Core**. Quindi la **Collection** e il **Core** sono gli stessi se la ricerca non è distribuita.

Solr Core

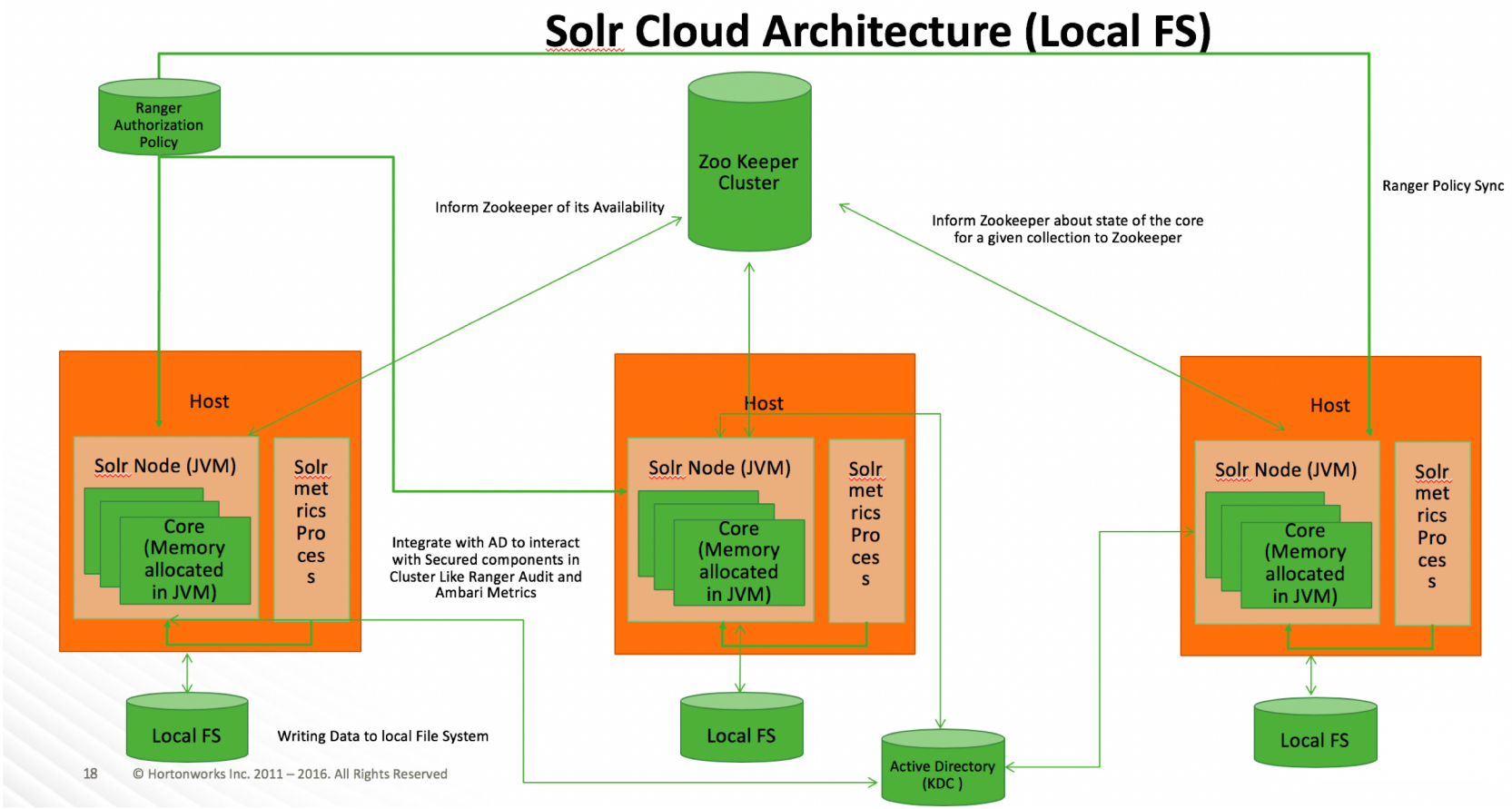






Antonio Scaiella
Gennaio 2023

INTRODUZIONE



Antonio Scaiella
Gennaio 2023

INSTALLAZIONE

Download

- Windows
  https://solr.apache.org/downloads.html

  Scarichiamo
  Binary releases: solr-8.11.2.zip
- Linux
  wget https://dlcdn.apache.org/lucene/solr/8.11.2/solr-8.11.2.tgz
  tar xvzf solr-8.11.2.tgz

- java -version
- Almeno la "1.8"

Antonio Scaiella
Gennaio 2023

AVVIO

Start

- .\bin\solr start -p "porta"
- .\bin\solr start –p 8983

Stop

- .\bin\solr stop -p 8983

GUI

- http://localhost:8983/solr/#/

Antonio Scaiella
Gennaio 2023

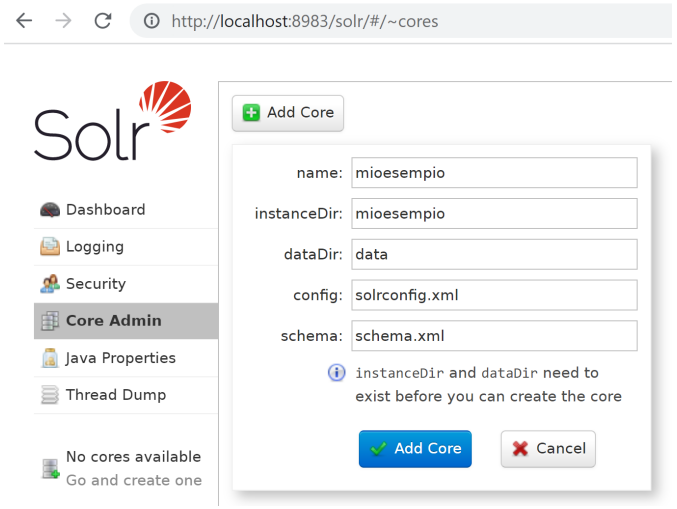
Creazione Core

- Command line:
  - Creazione
    Da solr-8.11.1 digitiamo: .\bin\solr create -c "nome core/collection"
    Da solr-8.11.1 digitiamo: .\bin\solr create -c mioesempio
  - Cancellazione
    .\bin\solr delete -c mioesempio

Antonio Scaiella
Gennaio 2023

Creazione Core

- Andiamo in solr-8.11.1\server\solr
- Creiamo un cartella con il nome del core
- Creiamo le sottocartelle:
  - «conf»
  - «data»
- Copiamo il contenuto di «solr-8.11.1\server\solr\configsets_default\conf» nella nostra cartella «conf»
- NOTA: se abbiamo già modificato la configurazione ad hoc copiamo la nostra "conf"
- Dalla GUI: inseriamo i vari campi e premiamo "Add Core"



Import Dataset di esempio

- Spostarsi in solr-8.11.1\example\exampledocs
    - java -jar -Dc=mioesempio post.jar *.xml

Antonio Scaiella
Gennaio 2023

The Standard Query Parser (lucene)

- q

Definisce la query utilizzando la sintassi standard. Questo parametro è obbligatorio.

- q.op

Specifica l'operatore predefinito per le query. I valori possibili sono "AND" o "OR".

- df

Specifica un campo predefinito dove ricercare

- fq

Specifica un filtro a priori sulle query

- fl

Specifica la lista dei field da ritornare

- sort

Specifica un ordinamento diverso da quello per score di default

The Standard Query Parser (lucene)

- Per la ricerca si può usare un singolo termine come : "test" o "hello"
- Più termini usando le virgolette: "hello world“
- Le Wildcard:
  - ? Esempio: ca?o estenderà a-> caso caro capo
  - * Esempio: cas* estenderà a-> casa casi cassetto ecc

The Standard Query Parser (lucene) Esempi

- q

Test

- df

Name

- fq

cat:"software”

- fl

id,name,manu,cat,score

The Standard Query Parser (lucene)

- Fuzzy Searches
  - casa~1 -> Casa caso casi cara
  - Tutti i termini a distanza 1 (il valore della distanza è a scelta)
- Proximity
  - "jakarta apache"~10
  - Stiamo chiedendo tutti i documenti che contengono i due termini a distanza al più 10 termini tra di essi

The Standard Query Parser (lucene)

- Range
  - popularity:[52 TO 10000]
  - title:{Aida TO Carmen}
- Con le parentesi quadrate denotiamo l'inclusione del primo e/o ultimo termine
- Con le parentesi graffe l'esclusione

The Standard Query Parser (lucene)

- Boosting a Term with "^"
  - jakarta^4 apache
  - "jakarta apache"^4 "Apache Lucene"

- Querying Specific Fields
  - title:"The Right Way" AND text:go
  - title:"Do it right" AND go ->se text è un field default

The Standard Query Parser (lucene)

- Boolean Operators
  - OR
    - "jakarta apache" OR Jakarta
    - "jakarta apache" || Jakarta
  - AND
    - "jakarta apache" AND "Apache Lucene"
    - "jakarta apache" && "Apache Lucene“
  - NOT
    - "jakarta apache" NOT "Apache Lucene"
    - "jakarta apache" ! "Apache Lucene"

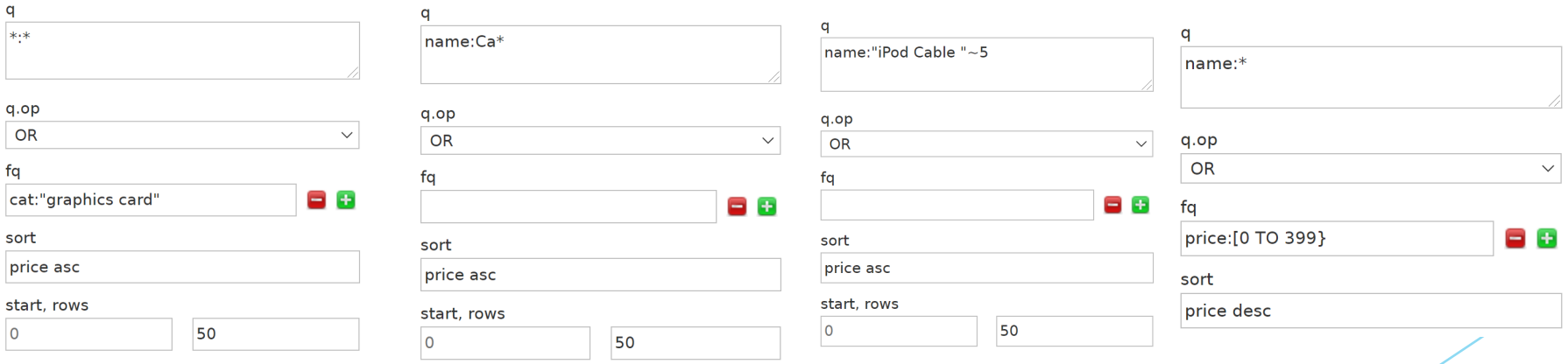
The Standard Query Parser (lucene)

- Grouping
  - (jakarta OR apache) AND website

The Standard Query Parser (lucene)

Esempi:

- Boosting
  - q: name:"Test"^55 manu:"Apache"^8



HTTP query

- http://localhost:8983/solr/mioesempio/select?indent=true&q.op=OR&q=name%3A%22Apple%22

The DisMax Query Parser

- qf

Specifica i fields dove ricercare con eventuale boost

- fq

Specifica un filtro a priori sulle query

- mm (Minimum Should Match)

Specifica il minimo numero di clausole che devono essere rispettate

Possono essere espresse come intero ed anche come percentuale

- bq

Il parametro bq specifica una clausola aggiuntiva, facoltativa, che verrà aggiunta alla query principale per influenzare il punteggio (esempio boost a doc recenti)

```
bq=date:[NOW/DAY-1YEAR TO NOW/DAY]
```

- Pf Phrase Fields

"aumentare" il punteggio dei documenti nei casi in cui tutti i termini nel parametro q appaiono molto vicini

Aggiungere Documenti con Post

Solr ha un comando "post" nella sua directory bin/. Usando questo comando, puoi indicizzare vari formati di file come JSON, XML, CSV in Apache Solr.

./post -c "nome_core" dati.csv

./post -h

Examples:

** JSON file:./post -c wizbang events.json*

** XML files: ./post -c records article*.xml*

** CSV file: ./post -c signals LATEST-signals.csv*

** Directory of files: ./post -c myfiles ~/Documents*

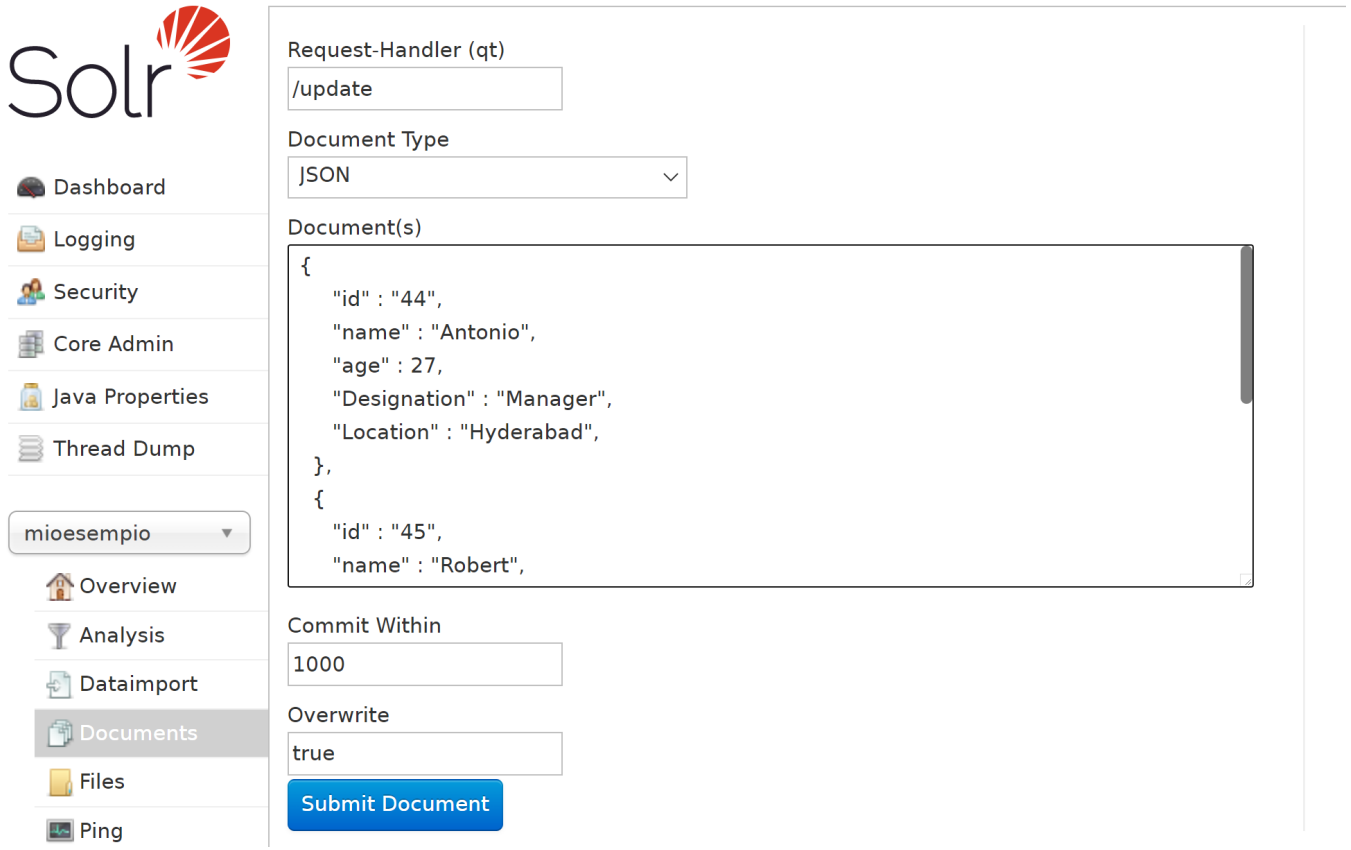
** Web crawl: ./post -c gettingstarted http://lucene.apache.org/Solr -recursive 1 -delay 1*

** Standard input (stdin): echo '{commit: {}}' | ./post -c my_collection -*

type application/json -out yes -d

** Data as string: ./post -c signals -type text/csv -out yes -d $'id,value\n1,0.47'*

Aggiungere Documenti con la Solr Web Interface



Aggiungere Documenti con Java Client API

SolrJ è un'API che semplifica la comunicazione tra le applicazioni Java e Solr. SolrJ nasconde molti dei dettagli della connessione a Solr e consente alla tua applicazione di interagire con Solr con semplici metodi di alto livello.

```
public HttpSolrClient getConnecton() {
    String urlString = "http://localhost:8983/solr/mioesempio";
    HttpSolrClient solr = new HttpSolrClient.Builder(urlString).build();
    solr.setParser(new XMLResponseParser());
    return solr;

}

public void index() throws SolrServerException, IOException {
    HttpSolrClient solrClient = getConnecton();

    SolrInputDocument document = new SolrInputDocument();
    document.addField("id", "123456");
    document.addField("name", "Kenmore Dishwasher");
    document.addField("price", "599.99");
    solrClient.add(document);
    solrClient.commit();
}
```