

INTRODUZIONE A

Solr



Antonio Scaiella
7 Gennaio 2025

Corso di Information Retrieval A.A. 2024/2025

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 2025

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 2025

INTRODUZIONE

Features of Solr

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

Antonio Scaiella
Gennaio 2025

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

INTRODUZIONE

Definizione di un Field in Solr

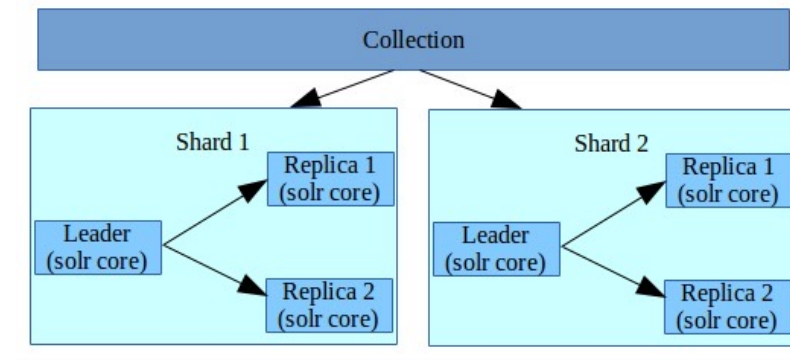
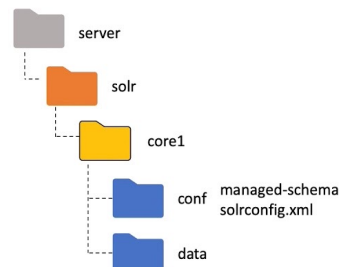
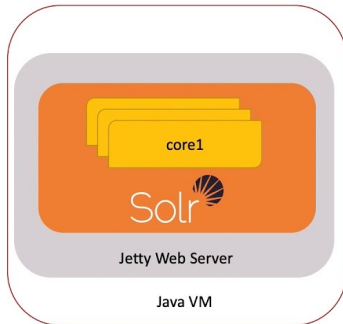
- **name:** Il nome del field
- **type:** tipo del field e.g. **text**,**string** long,int,date
- **indexed:** il field deve far parte dell'indice?
- **stored:** Il valore originale deve essere memorizzato?
- **multiValued:** è possibile assegnare più valori al campo?

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.

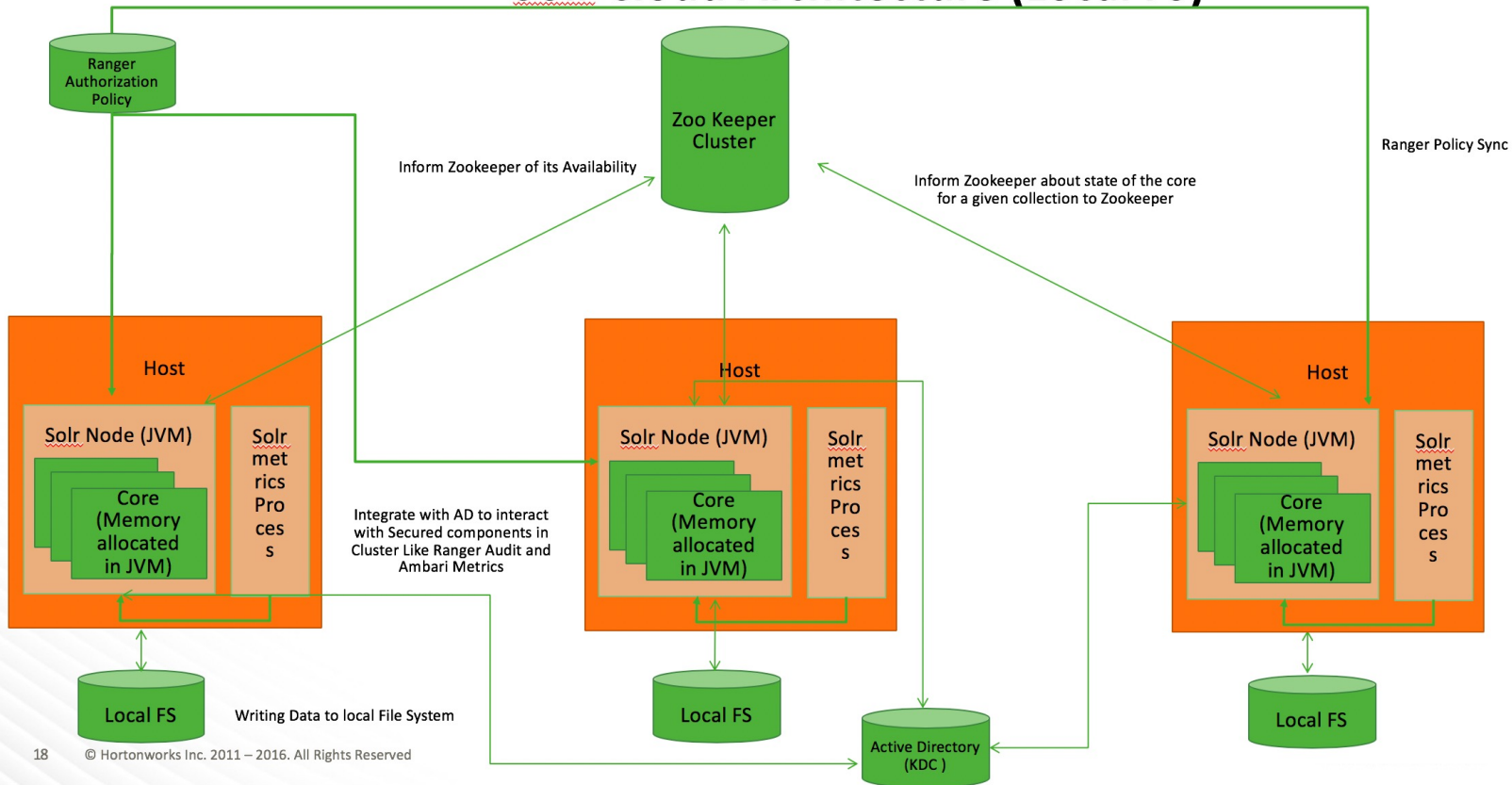
Solr Core



Antonio Scaiella
Gennaio 2025

INTRODUZIONE

Solr Cloud Architecture (Local FS)



INTRODUZIONE

- **Scalabilità:** Solr Cloud consente di aggiungere facilmente nuovi nodi e bilanciare il carico.
- **Sicurezza:** L'integrazione con Active Directory e Ranger garantisce autenticazione e autorizzazione robuste.
- **Alta disponibilità:** Zookeeper assicura che il sistema rimanga operativo anche in caso di guasti a uno o più nodi.
- **Gestione centralizzata:** Policy di sicurezza e metriche vengono gestite centralmente tramite Ranger e Ambari.

INSTALLAZIONE

Download

- Windows

<https://solr.apache.org/downloads.html>

Scarichiamo

Binary releases: solr-8.11.4.tgz

- Linux

```
wget https://dlcdn.apache.org/lucene/solr/8.11.4/solr-8.11.4.tgz
```

```
tar -xvzf solr- solr-8.11.4.tgz
```

- java -version
- Almeno la "1.8"

Antonio Scaiella
Gennaio 2025

AVVIO

Start

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

Stop

- `.\bin\solr stop -p 8983`

GUI

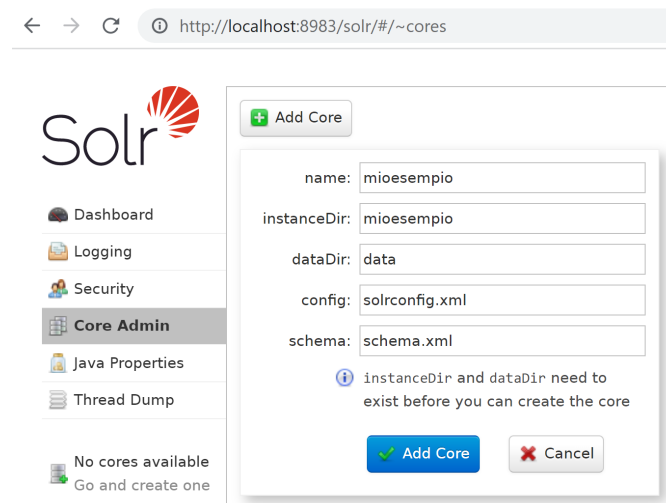
- `http://localhost:8983/solr/#/`

Creazione Core

- Command line:
 - Creazione
 - Da solr-8.11.4 digitiamo: `.\bin\solr create -c "nome core/collection"`
 - Da solr-8.11.4 digitiamo: `.\bin\solr create -c mioesempio`
 - Cancellazione
 - `.\bin\solr delete -c mioesempio`

Creazione Core

- Andiamo in `solr-8.11.4\server\solr`
- Creiamo un cartella con il nome del core
- Creiamo le sottocartelle:
 - «conf»
 - «data»
- Copiamo il contenuto di «`solr-8.11.4\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-8.11.4\example\exampledocs`
 - `java -jar -Dc=mioesempio post.jar *.xml`

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 (default field)

Specifica un campo predefinito dove ricercare

- ▶ fq (filter query)

Specifica un filtro a priori sulle query

- ▶ Fl (field list)

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

q
:

q.op
OR

fq
cat:"graphics card"  

sort
price asc

start, rows
0 50

q
name:Ca*

q.op
OR

fq

sort
price asc

start, rows
0 50

q
name:"iPod Cable "~5

q.op
OR



fq

sort
price asc

start, rows
0 50

q
name:*

q.op
OR

fq
price:[0 TO 399]  

sort
price desc

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



Dashboard

Logging

Security

Core Admin

Java Properties

Thread Dump

mioesempio

Overview

Analysis

Dataimport

Documents

Files

Ping

Request-Handler (qt)

/update

Document Type

JSON

Document(s)

```
{
  "id" : "44",
  "name" : "Antonio",
  "age" : 27,
  "Designation" : "Manager",
  "Location" : "Hyderabad",
},
{
  "id" : "45",
  "name" : "Robert",
```

Commit Within

1000

Overwrite

true

Submit Document

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();
}
```