

Apache Lucene & Apache Solr

Information Retrieval und Suchtechnologie

“Information retrieval (IR) is finding material (usually documents) of an unstructured nature (usually text) that satisfies an information need from within large collections (usually stored on computers).”

- Über 80% der in Firmen vorliegenden Information liegt in unstrukturierter, textueller Form vor.
- World Wide Web ist eine heterogene Sammlung von mit Hyperlinks verknüpften Dokumenten. Das vorherrschende Format ist dabei HTML, d.h. Text inkl. Markups für die Darstellung.
- Informationsexplosion: Digital verfügbare Datenmenge wächst dramatisch
- Der Benutzer an sich hat ein Informationsbedürfnis (Internetsuche, Unternehmenssuche, Spezialsuche)
- Problemstellung: Informationsbedürfnis des Nutzers zu stillen ohne den Nutzer mit Information zu überlasten z.B. nur die passenden Dokumente (Texte) oder Fakten zu liefern.

Gliederung

- Information Retrieval
- Was ist Apache Lucene?
- Wie funktioniert der Indizierungsprozess
- Woher stammt die Motivation
- Wie funktioniert Indizierungsprozess?
- Choose your weapon...
- Index
- Suchtypen
- Performance
- Was ist Apache Solr?
- Einbindung in Client - Server - Umgebung
- Dokumentdefinition
- Beispielabfragen
- Quellennachweis

Was ist Lucene?

- Stellt eine Bibliothek zur Informationsgewinnung dar
- Softwareprojekt der Apache Foundation (Apache Software License)
- Entwickelt von Doug Cutting
- Zentrale Idee ist die Verwendung eines Dokumentes mit Felder → Dadurch unabhängig von Dateitypen
- Es können diverse Dokumente (jedoch keine Bilder) indiziert werden, solange die Information darin extrahierbar ist
- Nutzer: Apple, DB Schenker, Disney, AOL, Eclipse Documentation, IBM, LinkedIn
- Geschichte: Arbeiten begannen bereits 1997, 2000: 0.01, 2001 Teil des Apache Projektes
- Aktuelle Version: 3.3

Lucene

Apache Solr

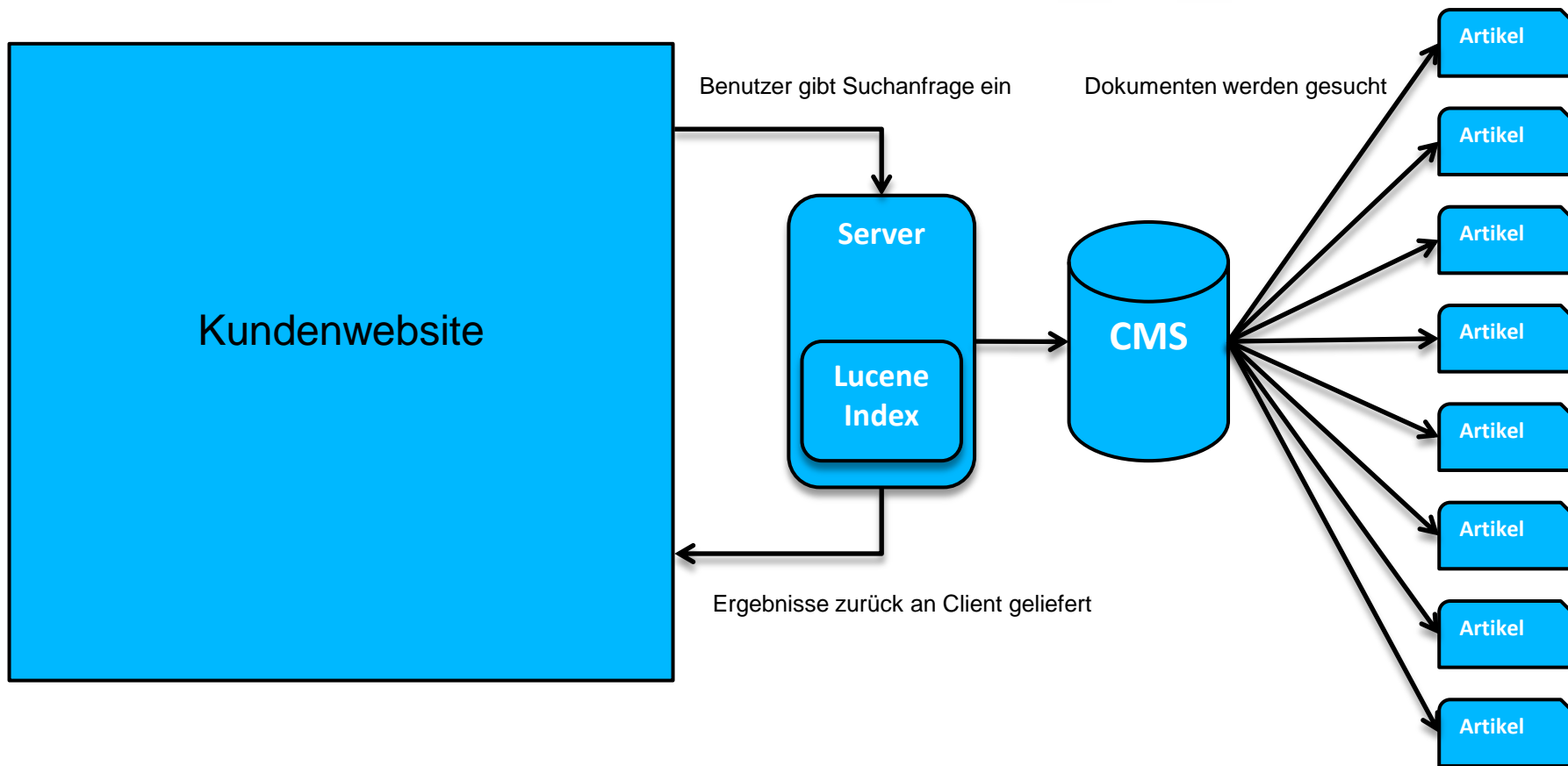
PyLucene

Apache Lucy

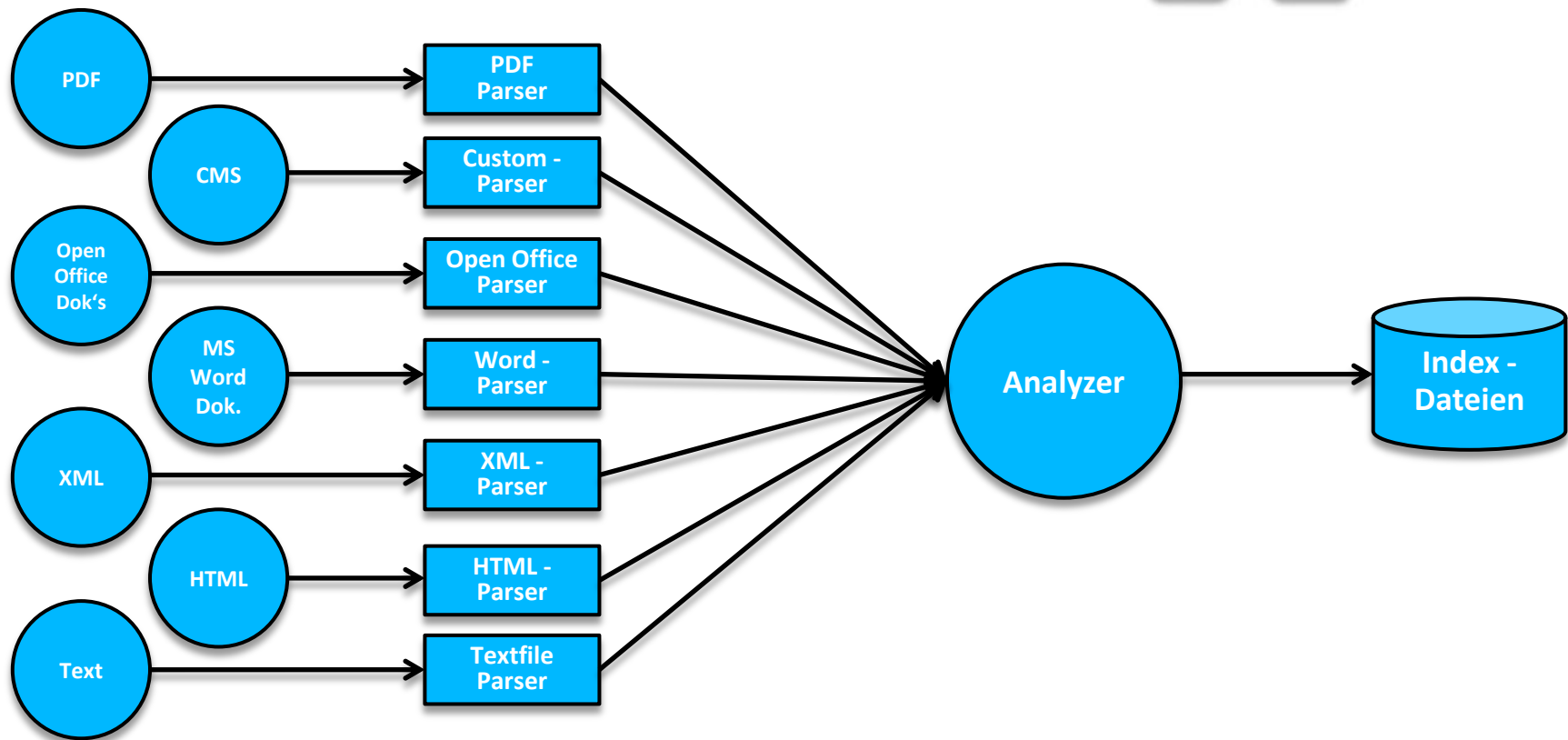
Lucene.net

nutch

Woher stammt die Motivation?



Indizierungsprozess



Welcher Analyzer soll verwendet werden?

Ziel

```
graph TD; Ziel[Ziel] --- Q1[Verwendung von Stopwords?]; Ziel --- Q2[Ist Groß- / Kleinschreibung wichtig?]; Ziel --- Q3[Index-Begriff : Einzahl ("Mensch")  
Suche: Mehrzahl ("Menschen")  
Treffer: Ja / Nein?]; Ziel --- Q4[Sollen Mailadressen, URLs erhalten bleiben?]; Ziel --- Q5[In welcher Sprache liegen die Dokumente vor?];
```

Verwendung
von
Stopwords?

Ist Groß- /
Kleinschreibung
wichtig?

Index-Begriff : Einzahl ("Mensch")
Suche: Mehrzahl ("Menschen")
Treffer: Ja / Nein?

Sollen Mailadressen,
URLs erhalten bleiben?

In welcher
Sprache liegen
die Dokumente
vor?

Analyzer – Beispiel

Analyzing "The quick brown fox jumped over the lazy dogs"

org.apache.lucene.analysis.whitespaceAnalyzer:

[The] [quick] [brown] [fox] [jumped] [over] [the] [lazy] [dogs]

org.apache.lucene.analysis.SimpleAnalyzer:

[the] [quick] [brown] [fox] [jumped] [over] [the] [lazy] [dogs]

org.apache.lucene.analysis.StopAnalyzer:

[quick] [brown] [fox] [jumped] [over] [] [lazy] [dogs]

org.apache.lucene.analysis.standard.StandardAnalyzer:

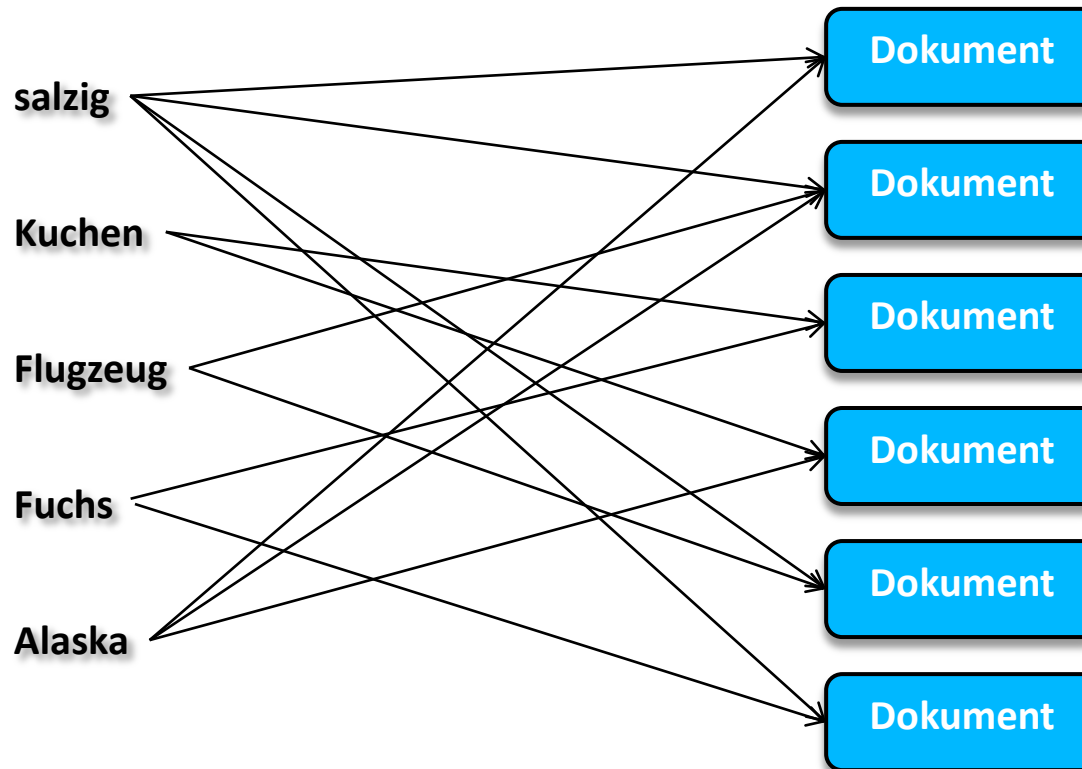
[quick] [brown] [fox] [jumped] [over] [lazy] [dogs]

org.apache.lucene.analysis.snowball.SnowballAnalyzer:

[quick] [brown] [fox] [jump] [over] [lazi] [dog]

Invertierter Index

Eine Liste von Ausdrücken die auf eine Anzahl von Dokumenten zeigt, in denen sie enthalten sind.



Indexstruktur

Index

Segment

Document

Feld Feld
Feld Feld

Document

Feld Feld
Feld Feld

Document

Feld Feld
Feld Feld

Segment

Document

Feld Feld
Feld Feld

Document

Feld Feld
Feld Feld

Document

Feld Feld
Feld Feld

Segment

Document

Feld Feld
Feld Feld

Document

Feld Feld
Feld Feld

Document

Feld Feld
Feld Feld

Suchmöglichkeiten

Platzhalter - Suche

T*ext*

Text*

T*ext

Te?t

Fuzzy Such

Text~

Boolsche Suche

AND

OR

NOT

Bereichssuche

[0..9]

[aida...epilog]

Einstellmöglichkeiten - Performance

mergeFactor

nach n eingelesen Dokument wird ein Segment auf die Platte geschrieben
nach n erstellten Segment werden diese Segmente zusammengefasst
Kleiner Wert: wenig RAM benötigt, Suche schneller, Indizieren langsam
Großer Wert: mehr RAM benötigt, Suche langsamer, Indizieren schneller
Min: 2, Standard: 10

minMergeDocs

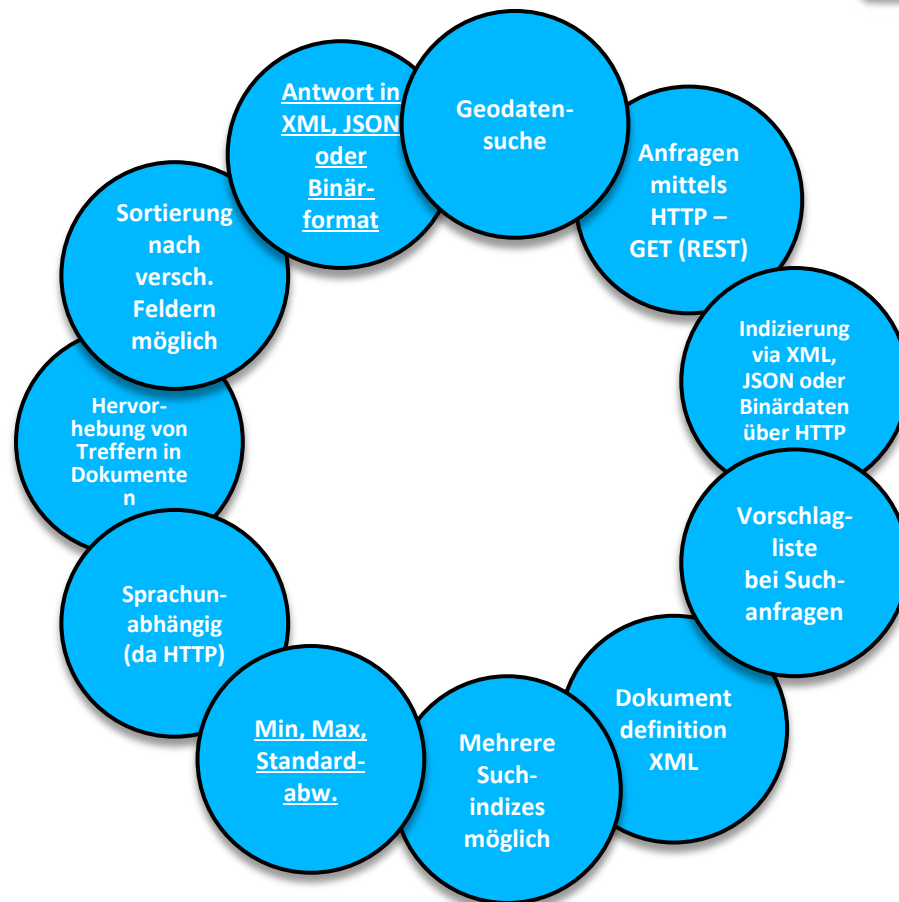
Mindestens n Dokumente müssen eingelesen werden, bevor diese in ein neues Segment geschrieben werden.
Großer Wert: Beschleunigt Indizieren, langsames Suchen

maxMergeDocs

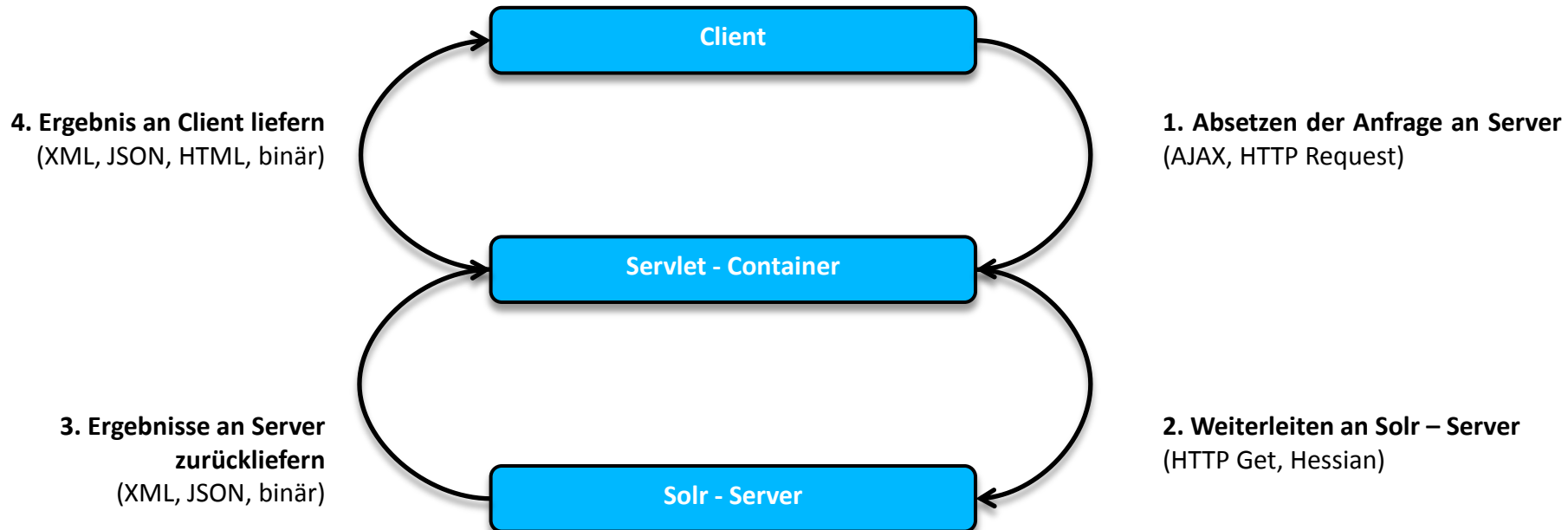
Bestimmt die maximale Anzahl von Dokumenten pro Segment.
Je höher der Wert, desto schneller die Suche da weniger Segmente durchsucht werden müssen
Standard: Integer.MAX_VALUE

Apache Solr

Was ist Solr?



Solr in einer Client – Server - Architektur



Dokumentdefinition

Datei: .../conf/schema.xml (Ausschnitt)

```
<field name="id" type="string" indexed="true" stored="true" required="true"/>
<field name="sku" type="text_en_splitting_tight" indexed="true" stored="true" omitNorms="true"/>
<field name="name" type="text_general" indexed="true" stored="true"/>
<field name="alphaNameSort" type="alphaOnlySort" indexed="true" stored="false"/>
<field name="manu" type="text_general" indexed="true" stored="true" omitNorms="true"/>
<field name="cat" type="string" indexed="true" stored="true" multiValued="true"/>
<field name="features" type="text_general" indexed="true" stored="true" multiValued="true"/>
<field name="includes" type="text_general" indexed="true" stored="true" termVectors="true" termPositions="true" termOffsets="true"/>
<field name="weight" type="float" indexed="true" stored="true"/>
<field name="price" type="float" indexed="true" stored="true"/>
<field name="popularity" type="int" indexed="true" stored="true"/>
<field name="inStock" type="boolean" indexed="true" stored="true"/>
```

indexed: Feld soll indizierbar (durchsuchbar & sortierbar)

stored: Feld soll im Index gespeichert werden

type: Datentyp

default: Standardwert, Wert nicht gesetzt

multiValued: Feld kann mehrere Werte pro Feld enthalten

required: Wert muss gesetzt sein

Datenabfrage - Beispiele

<SolrServer>/select?

q=video&fl=name,id,score

q=video&fl=name,id

q=video&sort=price asc

q=video&sort=inStock asc, price desc

q=video&wt=json

q=video&sort=price desc&fl=name,id,price

Quellenverzeichnis

<http://de.wikipedia.org/wiki/Levenshtein-Distanz>

<http://de.wikipedia.org/wiki/Solr>

<http://incubator.apache.org/lucy/>

http://lucene.apache.org/java/2_0_0/queryparsersyntax.html

<http://lucene.apache.org/solr/features.html>

<http://lucene.apache.org/solr/tutorial.html#Querying+Data>

<http://onjava.com/pub/a/onjava/2003/03/05/lucene.html>

<http://wiki.apache.org/lucene-java/PoweredBy>

<http://2003/07/30/LuceneIntro.html>

<http://www.christianherta.de/lehre/informationRetrieval/einfuehrung-suche-informationretrieval.pdf>

<http://www.fh-wedel.de/~si/seminare/ws02/Ausarbeitung/e.lucene/3.html>

<http://www.freiheitlich.org/2010/10/26/kurze-pause-%E2%80%93-groese-aufgaben/>

http://www.keldysh.ru/departments/dpt_10/lev.html