Technische Hochschule Wildau
Technical University of Applied Sciences

Masterarbeit

zur Erlangung des akademischen Grades
Master

Technische Hochschule Wildau

Fachbereich Wirtschaft, Informatik, Recht

Studiengang Bibliotheksinformatik (M. Sc.)

Thema (deutsch): Konzept zur Verbesserung des Relevanzrankings und der Rechercheeinstiege im Discovery-System der Sächsischen Landesbibliothek - Staats- und Universitätsbibliothek Dresden

Thema (englisch): Concept for improving relevance ranking and research options in the discovery system of the Saxon State and University Library Dresden

Autor/in:	Ulrike Schünemann
Seminargruppe:	BIM/16
Betreuer/in:	Dipl.-Informatiker Sascha Szott
Zweitgutachter/in:	Dr. Frank Seeliger
Spätestmögliche Abgabe:	13.02.2022

Abstract

Um Möglichkeiten zur Verbesserung des Relevanzrankings im SLUB-Katalog aufzuzeigen, werden in der vorliegenden Arbeit die aktuellen Konfigurationen der Suche im SLUB-Katalog analysiert. Zur Ermittlung geeigneter Indexfelder für die Suche werden insbesondere die verwendeten Feldtypen und die Belegungshäufigkeit in den angesprochenen Solr-Cores untersucht. Darauf aufbauend werden Vorschläge zu Änderungen der Konfigurationen der Standardsuche „Alles“ des SLUB-Katalogs und spezialisierten Sucheinstiege gemacht.

Durch die prototypische Entwicklung einer Webapplikation, mit welcher sich Konfigurationen der Rankingberechnung zur Suchanfragezeit anpassen lassen, können die Auswirkungen multipler Faktoren, wie z.B. Auswahl der Indexfelder und Boost-Werte, die die Relevanzberechnung beeinflussen, schneller und unabhängig vom IT-Personal getestet, verglichen und dokumentiert werden.

In order to identify possibilities for improving the relevance ranking in the SLUB catalogue, this paper analyses the current configurations of the search in the SLUB catalogue. To determine suitable index fields for the search, the field types used and the field occupancy in the Solr cores addressed are examined in particular. Based on this, suggestions are made for changes to the configurations of the SLUB catalogue’s default search option "All fields" and specific search fields.

The prototypical development of a web application that can be used to adjust configurations of the ranking calculation at query time allows the effects of multiple factors, such as selection of index fields and boost values that influence the relevance calculation, to be tested, compared and documented more quickly and independently by the IT staff.

Inhaltsverzeichnis

1 Einleitung

1.1 Motivation und Ziele

Herzstück einer Bibliothek ist deren Bestand. Ohne ein Nachweissystem ist der Bestand einer Bibliothek nicht nutzbar. Für Privatbibliotheken reicht eventuell noch das Wissen des Besitzers aus, um das gewünschte Buch zu finden. Größere Bestände verlangen nach einem Nachweissystem. Über Listen-, Band- und Zettelkataloge kam es in der letzten Hälfte des 20. Jahrhunderts dann immer mehr zur Nutzung von Datenbanken zum Nachweis von Bibliotheksbeständen. Ab den 1980er wurde zunehmend die Chance genutzt, Bibliotheksbestände auch über das Internet recherchierbar zu machen. Es entstanden die ersten Online Public Access Catalogues (OPACs). Ein OPAC greift auf die Katalogisierungsdatenbank der Bibliotheken zu.

Katalogisierungsdatenbanken sind auf die Verzeichnung des Bestandes an selbständigen Werken (Bücher, Zeitschriften, Karten etc.) spezialisiert. Die zunehmende Menge an elektronischen Ressourcen bietet immer mehr Metadaten zu den Inhalten der elektronischen Bücher, Zeitschriften und Datenbanken (Kapitel, Artikel, Datensätze etc.). Diese Inhalte, sogenannte unselbständige Werke, könnten auch in den Katalogisierungsdatenbanken erfasst werden. Die riesige Menge dieser Daten und weitere Faktoren, wie z.B. wechselnde Zugriffsrechte bei Verlagen, durch die sich der Bestand an unselbständigen Werken ständig ändert, machen das aber uneffektiv bzw. unmöglich. Zur Literaturrecherche mussten deshalb mehr und mehr verschiedene Recherchesysteme genutzt werden, da der Bibliothekskatalog nur den Zugriff auf Daten der selbständigen Werke bot. Deutlich schneller erschien Nutzenden dagegen Websuchmaschinen wie Google, die auf eine einzige Anfrage scheinbar alle notwendigen Informationen lieferten.

Discovery-Systeme, die „neuen Bibliothekskataloge“, machen sich die Suchmaschinentechnologie zunutze. Sie greifen auf einen Index oder mehrere Indizes zu und sind dadurch in der Lage „… sofortigen Zugang zu einem möglichst breiten Spektrum an Bibliotheksinhalten zu bieten, einschließlich nicht nur lokaler Sammlungen, sondern

auch der riesigen Mengen an Materialien extern bereitgestellter Ressourcen.“[1] Ziel des Einsatzes von Discovery-Systemen ist es, den Nutzenden einen niederschwelligen Einstieg in die Recherche zu ermöglichen, der der Suche in den allgemein bekannten Websuchmaschinen sowohl in Optik als auch Funktionalität ähnelt.

Inzwischen wechseln immer mehr Bibliotheken zu Discovery-Systemen als Rechercheinstrument. Laut Internetrecherchen verwendeten im vergangenen Jahr 75 von 84 bei der Deutschen Bibliotheksstatistik[2] als Universitätsbibliothek aufgeführten Bibliotheken ein Discovery-System.

Dass Nutzenden jetzt (fast) alle Medientypen unter gleichen Bedingungen recherchieren können und ggf. auch sofort Zugriff haben, erhöht die Attraktivität der Discovery-Systeme im Vergleich zu den althergebrachten OPACs signifikant.[3]

Andererseits erwächst dadurch auch eine neue Anforderung an Bibliotheken: 2014 schrieb Dirk Lewandowski: „Macht haben Suchmaschinen vor allem dadurch, dass sie entscheiden, was ein Nutzer zu seiner Suchanfrage zu sehen bekommt, verstärkt durch die Entscheidung, an welcher Stelle und in welcher Darstellungsform die Ergebnisse angezeigt werden. Im Suchprozess gibt es zahlreiche Stellen, an denen das Design der Suchmaschine die Entscheidung des Nutzers für oder gegen bestimmte Ergebnisse beeinflusst.“[4]

Das gilt entsprechend auch für Discovery-Systeme. Mit der Festlegung der Eingabefaktoren, die die Berechnung des Rankings bestimmen, kann die Bibliothek entscheiden, welche Inhalte ihren Nutzenden präferiert angeboten werden.

Dazu äußert sich David Bade 2007 kritisch: „Ganze Klassen von Materialien können unabhängig von ihrer Relevanz an das Ende der Liste der abgerufenen Elemente fallen, z. B. ältere Materialien oder Materialien in anderen Sprachen, bei denen der Suchbegriff nur in einem niedrig gewichteten Feld erscheint“[5] und Dirk Lewandowski bemerkt 2010:

[1] Breeding (2012), S. 21
[2] https://www.bibliotheksstatistik.de, zuletzt geprüft am 28.01.2022
[3] Christensen et al. (2018), S. 22
[4] Lewandowski (2014), S. 231
[5] Bade (2007), S. 833.

„Studien zum Selektionsverhalten innerhalb der Trefferlisten zeigen eine sehr starke Fokussierung auf die ersten Trefferplätze … Nutzer erwarten einen schnellen Weg zu den Ergebnissen und sind nicht bereit, lange über die Formulierung einer Suchanfrage nachzudenken.“[6]

Die Sächsische Landesbibliothek - Staats- und Universitätsbibliothek Dresden (SLUB) bietet ihren Nutzenden seit 2010 ein Discovery-System als Rechercheinstrument an. Vom kommerziellen System Primo der Firma Ex Libris wechselte sie 2015 in die finc-Nutzergemeinschaft. Sowohl das Content Management System (CMS) TYPO3 für die Nutzeroberfläche als auch die zugrundeliegende Suchmaschine Apache Solr basieren auf Open Source.

Im Team „Katalog“ der SLUB gehen immer wieder Stellungnahmen ein, die darstellen, wie subjektiv Antworten auf Benutzeranfragen bewertet werden. Eine Aufgabe des Teams „Katalog“ ist es, diese Stellungnahmen zu analysieren und zu prüfen, ob und wie das Relevanzranking des SLUB-Katalogs im Sinne dieser Stellungnahmen verbessert werden kann.

Ziel dieser Masterarbeit ist die Erarbeitung eines Konzepts für die Verbesserung des Relevanzrankings und der Rechercheeinstiege im Discovery-System der SLUB, dem SLUB-Katalog. Grundlage für derartige Verbesserungen sind Kenntnisse über aktuelle Konfigurationen und Datenstrukturen oder, wie Anne Christensen 2014 in einem Vortrag feststellte:

> *„Wer die Hintergründe kennt, findet auch Antworten auf Fragen nach Ranking, Metadaten & Abdeckung.*
> *Wer die Hintergründe kennt, kann Discovery besser machen.“*[7]

Aus diesen Gründen werden in dieser Arbeit die aktuellen Konfigurationen der Katalogsuche analysiert und die verfügbaren Felder des finc-Solr-Schemas hinsichtlich

[6] Lewandowski (2010), S. 90 f.
[7] Christensen (2014)

ihrer Eignung als Suchfeld geprüft. Dafür werden sowohl die Feldtypen der verwendeten Indexfelder als auch deren Belegungshäufigkeit untersucht.

Aktuell können Änderungen an den Eingabefaktoren, welche die Berechnung des Rankings im SLUB-Katalog bestimmen, nur durch einen IT-Mitarbeiter ausgeführt werden, welcher Erfahrung mit der TYPO3-Erweiterung TYPO3-find hat. Zum Testen einer neuen Konfiguration muss ein Test-Frontend des SLUB-Katalogs erstellt werden. Sollen mehrere geänderte Konfigurationen verglichen werden, ist dies für jede Konfiguration zu tun. Jede Änderung der Konfiguration bedarf wieder des Einsatzes dieses IT-Mitarbeiters.

Die Optimierung der Konfigurationen wird als iterativer Prozess eingeschätzt. Neben der Analyse und Dokumentation der aktuellen Konfigurationen soll deshalb prototypisch eine Webapplikation entwickelt werden, mit welcher sich Konfigurationen der Rankingberechnung zur Suchanfragezeit anpassen lassen. Dadurch können die Auswirkungen multipler Faktoren, wie z.B. Auswahl der Indexfelder und Boost-Werte[8], die die Relevanzberechnung beeinflussen, schneller und unabhängig vom IT-Personal getestet, verglichen und dokumentiert werden.

Die Arbeit soll Möglichkeiten für Verbesserungen an den Rechercheeinstellungen des SLUB-Katalogs aufzeigen. Es werden dabei beispielhaft Vorschläge für Änderungen unterbreitet. Die Umsetzung dieser Vorschläge ist nicht Teil der vorliegenden Arbeit.

[8] Boost-Wert: Als Boost, Boost-Wert oder auch Boost-Faktor werden Werte bezeichnet, die unabhängig von der vom Ranking-Algorithmus zur Berechnung verwendeten Formel Einfluss auf den berechneten Relevanzwert (Score-Wert, vgl. Kapitel 2.2) nehmen.

1.2 Sächsische Landesbibliothek – Staats- und Universitätsbibliothek Dresden

1996 fusionierten die Sächsische Landesbibliothek und die Universitätsbibliothek der TU Dresden zur heutigen Sächsischen Landesbibliothek – Staats- und Universitätsbibliothek Dresden (SLUB). Mit 5,7 Millionen Bestandseinheiten[9] gehört sie zu den größten wissenschaftlichen Universalbibliotheken Deutschlands.

Als Landesbibliothek obliegt ihr die Sammlung und Archivierung der Veröffentlichungen über Sachsen und der in Sachsen erscheinenden Publikationen. Dieser Auftrag zur Sammlung der Pflichtexemplare ist im Sächsischen Gesetz über die Presse[10] festgeschrieben. In diesem Sinne baut die SLUB den Bestand der Sammlung Saxonica, dessen Grundstock im 18. Jahrhundert an der kurfürstlichen Bibliothek gelegt wurde, systematisch aus. Als Universitätsbibliothek versorgt sie das wissenschaftliche Personal und die Studierenden der TU Dresden, seit Jahren eine der Exzellenzuniversitäten Deutschlands, mit Literatur und fördert Bemühungen um den offenen Zugang zu wissenschaftlichen Informationen.[11] Außerdem ist sie das Koordinierungs- und Servicezentrum für die Bibliotheken im Freistaat Sachsen.

Ein weiteres Beispiel ihres Engagements für Wissenschaft und Kultur ist die Koordination des Landesdigitalisierungsprogramms des Freistaates Sachsen und die vier an der SLUB ansässigen Fachinformationsdienste. Neben arthistoricum.net, dem Fachinformationsdienst für Kunst, Fotografie und Design und musiconn, dem Fachinformationsdienst Musikwissenschaft betreut sie auch den Fachinformationsdienst Mobilitäts- und Verkehrsforschung (FID move) sowie den Fachinformationsdienst Materialwissenschaft und Werkstofftechnik (FID Materials Science).

[9] https://www.bibliotheksstatistik.de, zuletzt geprüft am 28.01.2022
[10] https://www.revosax.sachsen.de/vorschrift/4197-SaechsPresseG, zuletzt geprüft am 04.02.2022
[11] Sächsische Landesbibliothek - Staats- und Universitätsbibliothek Dresden (2019)

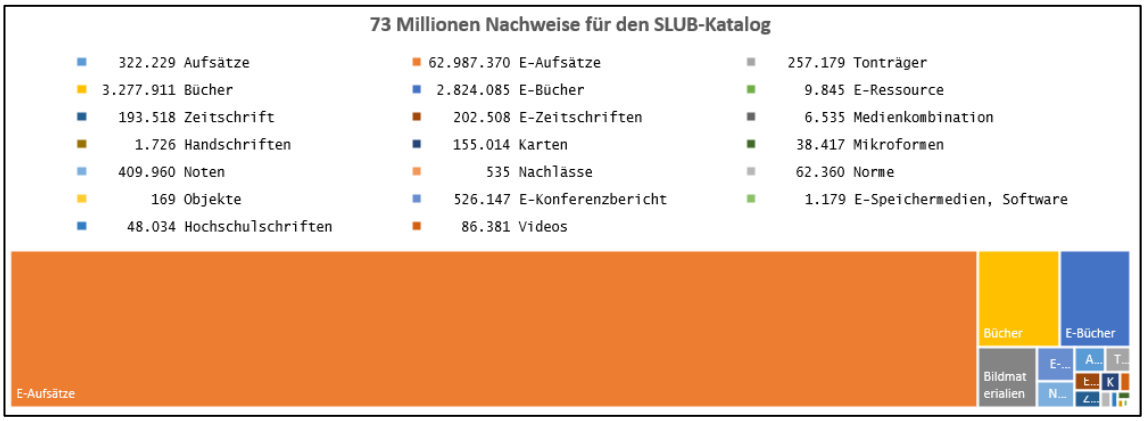


Abbildung 1: Medientypen im SLUB-Katalog, Stand Feb. 2022

Die Vielfalt der Aufgaben und Ausrichtungen spiegelt sich nicht nur in der Anzahl der im SLUB-Katalog, dem zentralen Nachweisinstrument der SLUB, recherchierbaren Datensätze, sondern auch in deren Heterogenität betreffs Herkunft der Daten und Medientypen wieder. Neben Büchern, Zeitungen, Zeitschriften und Karten weist die SLUB auch Noten und Fotos nach. Ebenso können Zugriffsdaten für Datenbanken, Volltextsammlungen und die Digitalen Sammlungen im Katalog gefunden werden Der SLUB-Katalog bietet aktuell 73 Millionen Nachweise aus 29 Datenquellen an (vgl. Abbildung 1 und Tabelle 1).

Datenquelle	Datensätze
AV-Portal der TIB-Hannover	17.480
BASE	506.411
Bayerische Staatsbibliothek / Musik	43.520
Crossref	53.596.158
Deutsche Fotothek	1.490.006
Deutsches Textarchiv	4.433
Directory of Open Access Books (DOAB)	49.024
Directory of Open Access Journals (DOAJ)	2.313.632
Diss online	265.040
Early English Books Online EEBO	144.356
Gallica Musik (Bibliothèque nationale de France, BNF)	55.755
GBV Musikdigitalisate	2.237
Handwörterbuch der musikalischen Terminologie	237
Hathi Trust	12.767
IEEE	1.069.743

Datenquelle	Datensätze
JSTOR	4.327.193
K10plus	6.861.210
Kalliope	535
Munzinger KLG	889
Music Treasures Consortium	3.215
Nationallizenzen (eBooks / Monographien)	717.094
Naxos Music Library	137.912
OECD iLibrary	26.881
Perinorm	62.360
Persèe	1.015.390
Primary Sources / Slavic Studies	2.088
Qucosa	22.273
Répertoire International des Sources Musicales (RISM)	92.682
SSOAR Social Science Open Access Repository	65.841

Tabelle 1: Datenquellen für den SLUB-Katalog

1.3 K10plus

Der K10plus ist eine vom Bibliotheksservice-Zentrum Baden-Württemberg (BSZ) und der Verbundzentrale des Gemeinsamen Bibliotheksverbund (VZG) gemeinsam betriebene Datenbank. Sie entstand durch die Vereinigung der Datenbanken der beiden Verbundsysteme Südwestdeutscher Bibliotheksverbund (SWB) und Gemeinsamer Bibliotheksverbund (GBV) und umfasst ca. 200 Millionen Bestandsnachweise.[12]

Die Bezeichnung K10plus leitet sich von den 10 beteiligten Bundesländern ab:

- Baden-Württemberg,
- Bremen,
- Hamburg,
- Mecklenburg-Vorpommern,
- Niedersachsen,
- Saarland,
- Sachsen,
- Sachsen-Anhalt,
- Schleswig-Holstein und
- Thüringen

Das „plus“ steht für zusätzliche Bibliotheken und Organisationen inner- und außerhalb dieser Bundesländer wie z.B. die Stiftung Preußischer Kulturbesitz und große Forschungseinrichtungen wie Leibniz-Institute, Helmholtz-Zentren oder Max-Planck-Institute.

Bibliotheken und Einrichtungen, die den oben genannten Verbünden angehören, verzeichnen ihre Bestände im CBS (das Zentrale Bibliothekssystem, eine Datenbank, hier der K10plus), indem sie einen sogenannten Lokalsatz (auch Lokaldatensatz oder Exemplarsatz) zu einem Titeldatensatz erstellen. Dadurch sigeln sie sich an diesen Titel an. Je nach Richtlinie des Verbundsystems und des durch die Bibliothek verwendeten Lokalsystems werden im Lokalsatz umfangreiche Angaben oder nur Basisinformationen gespeichert. Ein Lokalsystem oder Lokales Bibliothekssystem (LBS) verzeichnet lokale

[12] https://www.bszgbv.de/services/k10plus/, zuletzt geprüft am 02.02.2022

Daten der Bibliothek zu ihren Beständen, z.B. Inventarisierungs-, Erwerbungs- und Ausleihdaten.[13]

Die SLUB ist Mitglied des SWB und verwendet als Lokalsystem Libero der Firma Libero Library Management Solutions.

1.4 VuFind

VuFind gilt als eines der ersten auf Basis von Open Source Software entwickelten Discovery-Systemen. Der Ursprung des „von Bibliotheken für Bibliotheken konzipiert und entwickelt[en]“[14] Discovery-Systems liegt an der Villanova University in Pennsylvania (USA). Heute wird das Projekt durch eine internationale Community weiterentwickelt. Im Projektwiki werden fast 200 internationale Projekte erfasst, die VuFind als Plattform für ein Discovery-System einsetzen.[15] Dazu gehören auch viele sächsische Hochschulbibliotheken. VuFind ist eine PHP-Webapplikation, die Apache Solr als Standardsuchmaschine einsetzt. Dadurch stehen dem Discovery-System alle von Solr angebotenen Funktionen wie Ranking, Facettierung, Filter, Stemming etc. zur Verfügung. VuFind lieferte von Anfang an eine Möglichkeit zum Indexieren von MARC-Daten aus. Dafür standen ursprünglichen zwei PHP-Skripte zur Verfügung, die inzwischen durch das Open-Source-Tool SolrMarc ersetzt wurden. [16] Inzwischen wird VuFind standardmäßig mit SolrMarc ausgeliefert.[17]

1.5 SolrMarc

SolrMarc ist „ein konfigurierbares Java-basiertes Programm zum Indexieren von MARC-Datensätzen in einen Solr-Index“.[18] Um Werte aus den MARC-Feldern und -Unterfeldern zu extrahieren und daraus Indexeinträge zu erstellen, die einem Solr-Index hinzugefügt werden, verwendet SolrMarc ein konfigurierbares Skript. Spezifikationen für die Indexierung der MARC-Daten können in Konfigurationsdateien festgelegt werden. Dafür wird die Index Specification Language verwendet, die das Schreiben komplexer

[13] https://libero.com.au, zuletzt geprüft am 03.02.2022
[14] https://vufind.org/vufind/about.html, zuletzt geprüft am 29.01.2022
[15] https://vufind.org/wiki/community:installations, zuletzt geprüft am 29.01.2022
[16] Obenland (2017), S. 40–41
[17] https://vufind.org/wiki/indexing:solrmarc, zuletzt geprüft am 29.01.2022
[18] https://github.com/adjam/solrmarc/wiki/ConfiguringSolrMarc, zuletzt geprüft am 29.01.2022

Mappings vereinfacht. Außerdem ermöglicht SolrMarc die Einbindung Java-basierter Software-Komponenten, sogenannte CustomFunctions.

Die Grundlage für das Open-Source-Tool SolrMarc wurde von Wayne Graham, einem Programmierer des Colleges of William and Mary, Williamsburg, Virginia (USA) gelegt.[19] Inzwischen wird es, wie VuFind selbst, durch eine breite Community weiterentwickelt.

1.6 finc - find in catalog

Ab 2011 wurde das Projekt finc – **f**ind **in c**atalog an der Universitätsbibliothek Leipzig (UBL) aus EFRE-Mitteln gefördert. Dadurch wurde es möglich, an der UBL und 11 weiteren Hochschulbibliotheken Sachsens ein auf VuFind basierendes Discovery-System einzuführen. Seit 2012 sind die Systeme im produktiven Einsatz. Der zu Grunde liegende Solr-Index wird als finc-main bezeichnet. Im finc-main sind hauptsächlich bibliographisch selbständige Werke verzeichnet. Als Datenquelle hervorzuheben ist hier der K10plus, in welchem auch die Bestände der teilnehmenden Bibliotheken katalogisiert sind. Die Metadaten werden als MARC-Daten bezogen und von der UBL mit SolrMarc prozessiert.

Nach dem erfolgreichen Projektabschluss wurde das finc-Projekt in die finc-Nutzergemeinschaft überführt. Diese sichert seitdem unter der Führung der UBL den Dauerbetrieb der Kataloge und die dafür notwendige Infrastruktur.[20, 21]

Im Juli 2015 wurde, zusätzlich zum finc-main, zuerst durch fünf Bibliotheken der finc-Nutzergemeinschaft der finc-eigene aggregierte Artikelindex (finc-ai) in Betrieb genommen. Der Artikelindex umfasst über 90 Millionen Datensätze und weist eine sehr heterogene Datengrundlage auf. Beispielhaft seien hier die Datenquellen Crossref, JSTOR und DOAJ genannt. Für diese nicht in MARC vorliegenden Daten wird zur Konvertierung der Metadaten in das Indexschema das an der UBL entwickelte Werkzeug

[19] Tramullas und Garrido (2013), S. 90
[20] https://blog.ub.uni-leipzig.de/dem-katalog-unter-die-haube-geschaut, zuletzt geprüft am 28.01.2022
[21] Lazarus (2016)

„span“ eingesetzt. Dabei wird eine Zwischenrepräsentation, das Intermediate Schema, zur Vereinheitlichung verschiedener Ausgangsformate genutzt. Das Werkzeug „span“ ermöglicht eine schnelle Verarbeitung von Millionen von Datensätzen.[22]

Die SLUB ist seit 2015 Mitglied der finc-Nutzergemeinschaft und beteiligt sich aktiv an der Datenbereitstellung für die gemeinsam genutzten Solr-Indizes. Derzeit gibt es 24 Anwender. Dazu gehören sowohl wissenschaftliche Bibliotheken als auch Fachinformationsdienste innerhalb- und außerhalb Sachsens. [23]

[22] https://github.com/miku/span, zuletzt geprüft am 02.02.2022
[23] https://finc.info/anwender, zuletzt geprüft am 28.01.2022

2 Grundlagen

2.1 Information Retrieval

Die stetige Zunahme von Informationen stellte die Informationswissenschaft bereits in den 50er Jahren vor die Aufgabe, neue Methoden zum Wiederfinden und Bereitstellen von Informationen zu entwickeln. Ab diesem Zeitraum sind Forschungen zum Information Retrieval nachweisbar. Es wurde notwendig, Maschinen zur Unterstützung der Informationssuche zu befähigen. Das automatische Indexieren von Dokumenten durch textstatistische Verfahren geht auf Hans-Peter Luhn (1896 - 1964), einen der Pioniere des Information Retrieval, zurück.[24]

> *„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).“*[25]

Information Retrieval, Informationsrückgewinnung, bedeutet für Manning, Raghavan & Schütze (2008) in ihrer Definition die Befriedigung von Informationsbedürfnissen durch das Auffinden von meist unstrukturierten Daten (z.B. Texten) aus einer großen Datensammlung.[26] Beispiele für unstrukturierte Daten sind Volltexte von Artikeln, Inhalte digitalisierter Bücher, Inhaltsverzeichnisse oder Webseiten. Diese Objekte werden im Kontext von Information Retrieval i.A. als „Dokumente“ bezeichnet, eine zusammengehörende Menge von Dokumenten als „Kollektion“. Sie unterscheiden sich von den klar strukturierten Inhalten einer Datenbank, die z.B. Kontonummern oder Buchungsdaten beinhalten. Zu einem Dokument im Sinne von Information Retrieval können neben dem unstrukturierten Text auch beschreibende Informationen gehören. Diese werden als Metainformationen oder Metadaten bezeichnet, da sie Informationen zu Informationen enthalten. Titel, Abstracts oder gar Volltexte lassen sich nicht einfach abrufen wie Kontonummern oder Datumsangaben, da sich die Bedeutung von Worten nach dem jeweiligen Kontext richtet. Deshalb befasst sich Information Retrieval nach der

[24] Stock (2007), 38 f.
[25] Manning et al. (2008), S. 1
[26] Manning et al. (2008), S. 1

Auffassung von Salton mit der Repräsentation, der Speicherung, der Organisation und dem Zugriff auf Informationen bzw. Informationsobjekten:

> *„Information Retrieval (IR) is concerned with the representation, storage, organization, and accessing of information items. In principle no restriction is placed on the type of item handled in information retrieval.“*[27]

Als Antwort auf einen Informationsbedarf liefern Information Retrieval Systeme eine unbestimmte Anzahl von Dokumenten aus einer Kollektion. Dazu muss die natürlich sprachliche Anfrage in eine Repräsentation gewandelt („Formulierung der Suchanfrage“ vgl. Abbildung 2) werden, die mit den gespeicherten Repräsentationen der Dokumente verglichen werden kann. Die Umwandlung von Dokumenten in eine Repräsentation, die Indexierung, wird in Kapitel 2.3 beschrieben.

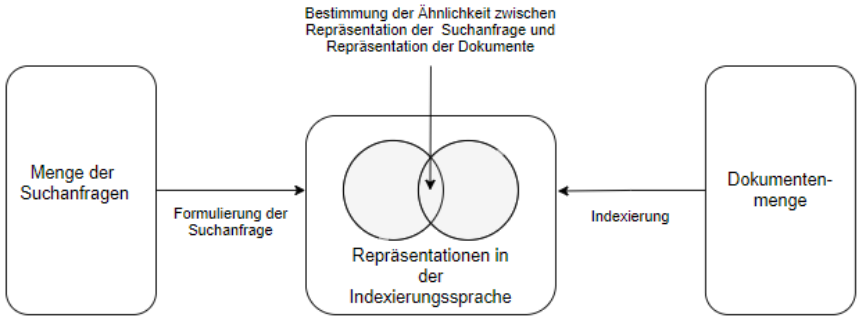


Abbildung 2: Funktionen eines Information Retrieval Systems nach Salton[28]

Um in einer Menge von Dokumenten Informationen zu finden, müssen Algorithmen entwickelt werden, welche die Repräsentation der Sucheingabe mit der Repräsentation des Dokuments vergleichen können. Bei ausreichender Ähnlichkeit wird das Dokument in die Treffermenge einbezogen.

[27] Salton und McGill (1983), S. 1
[28] Salton und McGill (1983), S. 12

> *„Understanding and modeling how people compare texts, and design computer algorithms to accurately perform this comparison, is at the core of information retrieval.“*[29]

Der Prozess des Abgleichs einer Anfrage mit der Datenmenge wird in einem Retrieval Modell abgebildet.[30] Durch die Möglichkeit, unstrukturierte Texte durch Algorithmen wieder auffindbar zu machen, können Datenmengen zugänglich gemacht werden, die sich jeder manuellen Erfassung in Katalogen oder Datenbanken entzogen hätten. Dabei muss mit vagen Anfragen ebenso umgegangen werden wie mit den Unsicherheiten bei der Antwort. Vagheit bei Anfragen entsteht dadurch, dass das Informationsbedürfnis nicht genau formuliert wird. Beispielsweise drückt die Eingabe eines Städtenamens noch nicht aus, ob Informationen zur Stadtgeschichte, zu Sehenswürdigkeiten oder zur Topographie etc. gesucht werden. Andererseits entstehen bei den Antworten Unschärfen z.B. durch Homonyme. Bei dem Begriff „Schloss“ kann es sich sowohl um ein Gebäude als auch um einen Schließmechanismus handeln.[31]

2.2 Relevanz und Relevanzranking

Nicht nur das Auffinden von Informationen, sondern auch deren Bewertung in Bezug auf die Anfrage ist Aufgabe der Information-Retrieval-Systeme. Durch die Bewertung der Ergebnisse werden diese in eine Rangfolge gebracht. Im Idealfall sollte das Dokument an erster Stelle stehen, welches das Informationsbedürfnis der Nutzenden am besten befriedigt. Relevanz ist ein wichtiges Konzept im Information Retrieval.[32]

„Relevanz (lat./ital.: re-levare ,,[den Waagebalken, eine Sache] wieder bzw. erneut in die Höhe heben‘‘) ist eine Bezeichnung für die Bedeutsamkeit und damit sekundär auch eine situationsbezogene Wichtigkeit, die jemand etwas in einem bestimmten Zusammenhang beimisst.“[33]

[29] Croft et al. (2010), S. 2
[30] Croft et al. (2010), S. 5
[31] https://fg-retrieval.gi.de/fachgruppe/ueber-die-fachgruppe, zuletzt geprüft am 27.06.2021
[32] Croft et al. (2010), 4 f.
[33] https://de.wikipedia.org/wiki/Relevanz, zuletzt geprüft am 12.01.2022

Hier wird die Schwierigkeit bei der Entwicklung von Ranking-Algorithmen sichtbar. Behnert und Borst beschreiben: „Da Relevanz von höchst subjektiver und dynamischer Natur ist, wird sie von jedem Menschen anders wahrgenommen.“[34]

Folgende Kriterien können neben der thematischen Komponente für den Nutzenden entscheidend sein:

- Sprache des Textes – Ist er zu verstehen?
- Aktualität des Textes – Sind die Informationen noch gültig?
- Zugänglichkeit des Textes – Ist der Text verfügbar (online / in der Bibliothek)?[35]

Außerdem interessiert ggf., ob

- der Text von der Person bereits gelesen wurde
- viele andere den Text gelesen haben
- der Text häufig zitiert wurde[36]

Zur Bestimmung der Relevanz von Dokumenten greifen Retrieval Modelle häufig auf die statistischen Eigenschaften der Texte zu. Dazu zählen

- Häufigkeiten eines Terms im Dokument
- Häufigkeiten eines Terms in der Dokumentkollektion
- Position des Terms im Dokument / Nähe von Termen
- Anzahl der Terme im Dokument (Dokumentlänge)

Für Retrieval Systeme sind Texte eine Menge von Termen (z.B. Wörtern).[37] Bereits in den 1970er Jahren vertrat Karen Spärck Jones die Ansicht, „… dass ein Anfrageterm, der in vielen Dokumenten vorkommt, kein gutes Unterscheidungsmerkmal ist und weniger gewichtet werden sollte als ein Term, der in wenigen Dokumenten vorkommt“[38]. 1972 schlug sie erstmals ein Maß für Term-Spezifität vor. Diese Angabe wird unter der Bezeichnung inverse Dokumentenhäufigkeit (inverse document frequency) fast überall zur Berechnung des Term-Gewichts eingesetzt.[39]

[34] Behnert und Borst (2015), S. 387
[35] Croft et al. (2010), 4 f.
[36] Croft et al. (2010), 4 f.
[37] https://webarchiv.ethz.ch/dbs/education/mmir/SS_02/Folien/kap1_1.pdf, zuletzt geprüft am 25.01.2022
[38] Robertson (2004), S. 503
[39] Robertson (2004), S. 503

Anders formuliert: Der Inhalt eines Dokuments wird durch die enthaltenen Terme beschrieben. Dafür sind Terme unterschiedlich gut geeignet. Terme, die in jedem Dokument vorkommen, haben wenig bis keine thematische Aussagekraft. Terme, die in nur wenigen Dokumenten vorkommen, dafür eine hohe. Sie heben das Dokument aus der Masse heraus. Kommt der sonst seltene Term mehrfach im Dokument vor, ist davon auszugehen, dass dieses Dokument eine hohe thematische Bedeutung für eine Anfrage zu diesem Term hat. Die Anzahl der Dokumente, die einen bestimmten Term enthalten, ist indirekt proportional zum Informationsgehalt des Terms.

Der Score-Wert drückt aus, wie gut ein Dokument zur Anfrage passt. In der Rangliste werden die Dokumente absteigend nach dem Score-Wert sortiert. Die Berechnung des Score-Werts erfolgt durch den Ranking-Algorithmus.

2.3 Suchmaschinen

Nach Croft (2010) sind Suchmaschinen „die praktische Anwendung von Information Retrieval Techniken auf große Textsammlungen“[40]. Darunter fallen für ihn Anwendungen wie

- Web- bzw. Internetsuche
- Desktopsuche (Personal Search)
- Unternehmenssuche (Enterprise Search)
- Vertikale Suchmaschinen
- Metasuchmaschinen
- Bibliothekssuchmaschinen

„Suchmaschine“ bezeichnete ursprünglich speziell zur Textsuche entwickelte Hardware. Ab den 80er Jahren ersetzt der Begriff mehr und mehr „Information Retrieval System“ und beschreibt damit den zugrundeliegenden Prozess. Suchmaschinen werden danach beurteilt, wie gut ihr Relevanzranking ist, ob sie Rückschlüsse auf die Qualität der Ergebnisse zulassen und wie ihre Interaktionen mit dem Informationssuchenden gestaltet werden.

[40] Croft et al. (2010), S. 6

Erwartungen an Suchmaschinen sind aber auch

- kurze Antwortzeiten
- Relevanzsortierung
- Ergebnisse, die thematisch zur Anfrage passen
- einfache Bedienung
- Benutzerfreundlichkeit
- geringe Netzwerkbelastung[41]

Die Basis einer Suchmaschine bildet deren Index. Meist werden invertierte Indizes verwendet. Die beiden grundlegenden Elemente eines invertierten Indizes sind das Lexikon (auch Wortschatz, vocabulary oder dictionary) und das Vorkommen (auch Häufigkeit oder Frequenz) der Terme. Bedingung für die Erstellung eines invertierten Indexes ist die Vergabe einer ID an jedes Dokument. Der invertierte Index entsteht, indem zu jedem Term, welcher in den Dokumenten einer Kollektion vorkommt, eine sortierte Liste der IDs der Dokumente gespeichert wird, welche den Term enthalten. Durch diese Zuordnung zwischen Term und Dokumenten-IDs können schnell alle Dokumente, die den gesuchten Term enthalten, identifiziert werden. Die Suche wird beschleunigt.

Document Collection		*Vocabulary*		*Inverted lists*
Doc.	*Text*	*Word*	*Freq.*	
1	fat rat eat cat	bat	3	⟨2, 5, 6⟩
2	fat rat eat bat	cat	3	⟨1, 3, 4⟩
3	mat cat eat rat	eat	6	⟨1, 2, 3, 5, 6, 8⟩
4	mat cat fat cat	fat	4	⟨1, 2, 4, 5⟩
5	fat rat eat fat mat bat	mat	4	⟨3, 4, 5, 7⟩
6	bat eat rat	rat	7	⟨1, 2, 3, 5, 6, 7, 8⟩
7	rat sat mat	sat	1	⟨7⟩
8	rat eat rat eat rat eat rat			

Abbildung 3: Example of an inverted file index (Table 7.1)[42]

Im Beispiel aus Abbildung 3 werden acht Dokumente indexiert: Im Lexikon („Vocabulary“) stehen eine Liste mit allen vorkommenden Termen („Word“) und die Angabe, in wie vielen Dokumenten die Terme vorkommen („Freq.“). Unter „Inverted lists“ sind die IDs der Dokumente aufgelistet, die den Term enthalten. Je nach den Anforderungen des Retrieval Modells können zusätzliche Angaben im invertierten Index gespeichert werden. Darunter fallen neben der bereits genannten Term-Frequenz z.B. die

[41] Schuldt (2015), S. 2
[42] Baeza-Yates et al. (2002), S. 197

Position des Terms im Dokument und die Term-Gewichtung**.** Das Speichern dieser zusätzlichen Angaben beschleunigt die Abfrage, muss aber gegen den höheren Speicherbedarf abgewogen werden.

In Abbildung 4 werden die beiden Basisprozesse einer Suchmaschine dargestellt.

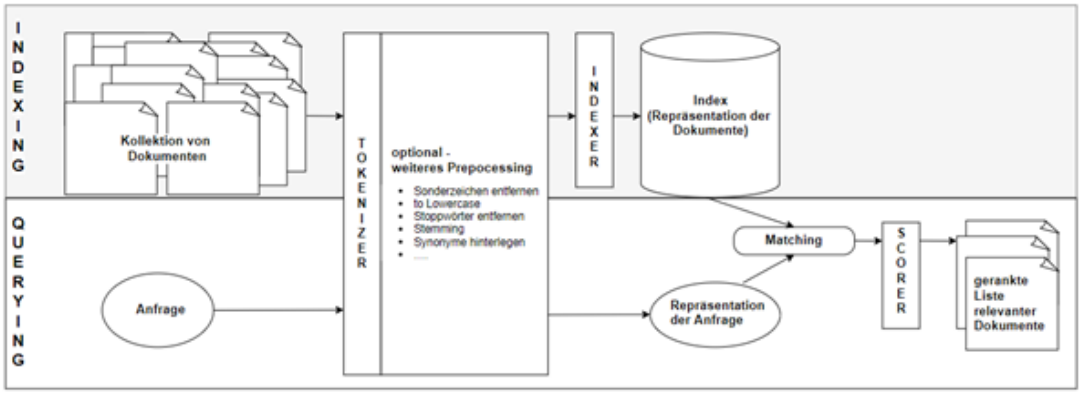


Abbildung 4: Basisprozesse einer Suchmaschine (eigene Darstellung)

Indexierung (Indexing, Indizierung) ist der Prozess, der aus den Dokumenten einer Kollektion deren Indexrepräsentation erstellt. Es werden verschiedene **Tokenizer** und **Filter** eingesetzt, um den Inhalt der Dokumente an festgelegten Trennzeichen (z.B. Leerzeichen) in einzelne Token zu trennen und die Tokens zu normalisieren. „Gewöhnlich sind Tokens einzelne Wörter oder Terme. Komplexe Strukturen, wie E-Mail-Adressen, Internetadressen (URLs), Phrasen oder Emoticons, können ebenso ein Token sein.“[43] Im Kapitel 2.1 wurde beschrieben, dass beim Information-Retrieval nicht mehr auf ein exaktes Matching der Suchbegriffe gezielt wird. Durch Vorverarbeitungsschritte wie die Umwandlung der Token in Kleinbuchstaben (Lowercasing), das Entfernen von Sonderzeichen, das Abschneiden von Wortendungen (Stemming) oder die Hinterlegung von Synonymen werden mehr Dokumente in die Treffermenge einbezogen.

Ist die Dokumentkollektion lokal gespeichert und muss nicht über eine Netzwerkschnittstelle angesteuert werden, kann der Prozess offline erfolgen.

[43] Wilhelm-Stein (2016), S. 13

Beispielsatz:	„Franz fährt in dem blauen Auto durch das Land, obwohl rote Autos auch fahren können.“

Beispielbearbeitung

Tokenizing (Trennen am Leerzeichen, Satzzeichen entfernen)	„Franz“, „fährt“, „in“, „dem“, „blauen“, „Auto“, „durch“, „das“, „Land“, „obwohl“, „rote“, „Autos“, „auch“, „fahren“, „können“
Stoppwörter entfernen	„Franz“, „fährt“, „blauen“, „Auto“, „Land“, „rote“, „Autos“, „auch“, „fahren“, „können“
Umwandlung von Umlauten	„Franz“, „fahrt“, „blauen“, „Auto“, „Land“, „rote“, „Autos“, „auch“, „fahren“, „konnen“
Lowercasing	„franz“, „fahrt“, „blauen“, „auto“, „land“, „rote“, „autos“, „auch“, „fahren“, „konnen“
Stemming	„franz“, „fahr“, „blau“, „auto“, „land“, „rot“, „auto“, „auch“, „fahr“, „konn“

Tabelle 2: Beispielbearbeitung eines Satzes vor der Indexierung

Beim Einsatz von Filtern muss die in den Dokumenten verwendete Sprache bedacht werden. Stoppwörter, Umlaute, Endungen werden jeweils sprachspezifisch sein.

Anfrageverarbeitung (Suche, Querying) ist der Prozess, bei welchem die über die Benutzeroberfläche an die Suchmaschine übergebene Anfrage mit den im Index repräsentierten Dokumenten verglichen wird. Dazu wird eine Repräsentation derselben erstellt, indem i.A. die gleichen Tokenizer und Filter wie bei der Indexierung durchlaufen werden. Anschließend werden über den invertierten Index die Dokumente ermittelt, die der Repräsentation der Anfrage ähnlich sind, also alle Terme oder eine festgelegte Mindestanzahl an Termen der Anfrage enthalten. Durch den **Scorer** wird für diese Dokumentrepräsentationen auf der Grundlage des verwendeten Retrieval Modells der Score-Wert berechnet, der Auskunft darüber gibt, wie relevant das Dokument für Anfrage ist. Die Liste der absteigend nach Relevanz sortierten Dokumente kann dann an eine Benutzeroberfläche übergeben werden.

2.4 Suchmaschine Solr

Auch Anfang 2022 ist Solr laut DB-Engines nach Elasticsearch und Splunk eine der populärsten Suchmaschinen.[44] Der plattformunabhängige Such-Server Solr ist ein Open-Source-Software-Projekt, das durch die Apache Software Foundation gefördert wird.

Der Solr zu Grunde liegende Lucene Core ist für die Erstellung des Index zuständig und stellt Suchalgorithmen für sortierte Trefferlisten bereit. Solr ergänzt diese Funktionen um Möglichkeiten zur Administration und Replikation. Außerdem bietet Solr weitere nützliche Funktionen wie Skalierbarkeit, Spellchecking, Highlighting, Facetting etc. Die mitgelieferte Administrationsoberfläche erleichtert die Konfiguration der Suchmaschine.

Solr besteht aus verschiedenen Komponenten, die modular aufgebaut sind. Für die verschiedenen Aufgaben stellt Solr unterschiedliche RequestHandler zur Nachnutzung bereit. Zur Indexierung der Daten werden je nach Ausgangsformat DataImportHandler oder UpdateRequestHandler angesprochen. Sie sind verantwortlich, dass Daten aus einer Datenquelle ausgelesen und aufbereitet werden. Sie erstellen den invertierten Index, durch den eine effiziente Suche ermöglicht wird.[45] [46]

Anfragen nimmt der Solr-Server vom Client durch einen HTTP-Request entgegen. RequestHandler der Klasse SearchHandler, die wiederum aus verschiedenen Search-Komponenten bestehen können, sind für die Verarbeitung der Suchanfragen zuständig. Festlegungen zu verwendeten RequestHandlern und deren Komponenten werden in der solrconfig.xml getroffen. Der ResponseWriter bestimmt, in welchem Format die Antwort, der Response ausgegeben wird. [47] [48]

[44] https://db-engines.com/de/ranking/suchmaschine, zuletzt geprüft am 02.01.2022
[45] Klose (2014), S. 17
[46] Klose (2014), S. 45
[47] Klose (2014), S. 17
[48] Klose (2014), S. 45

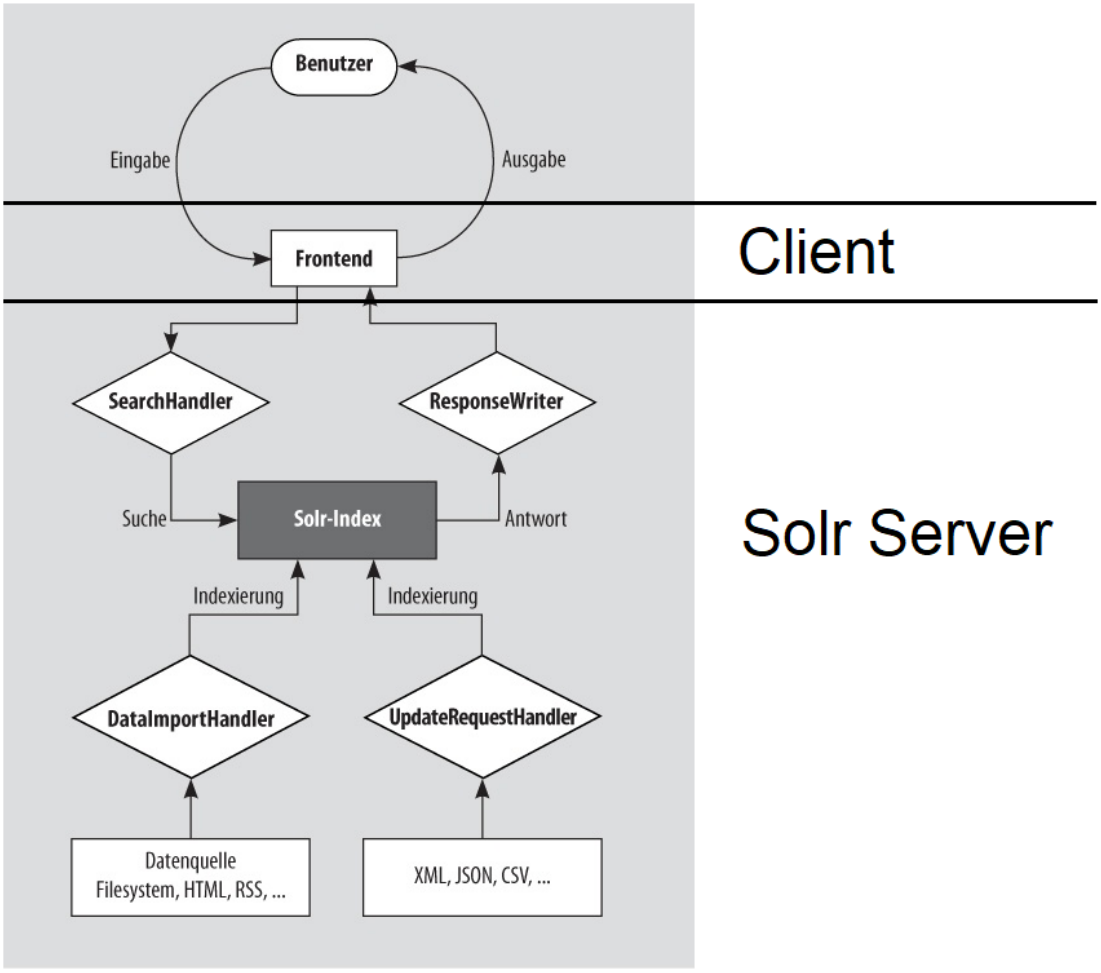


Abbildung 5: Diagramm einiger Solr-Komponenten[49]

2.4.1 Architektur einer Solr-Instanz

Innerhalb einer Solr-Instanz können mehrere voneinander unabhängige Solr-Cores betrieben werden. Jeder Core benötigt (neben einer eindeutigen Bezeichnung) eine solrconfig.xml, in welcher allgemeine Einstellungen zum Index wie z.B. UpdateHandler, ResponseWriter und RequestHandler festgelegt werden. In einer schema.xml kann definiert werden, wie der zugehörige Lucene-Index aufgebaut wird. Es ist aber auch möglich, Dokumentkollektionen ohne ein vorher festgelegtes Schema zu indexieren.

[49] Klose (2014), S. 16

2.4.2 SearchHandler

SearchHandler gehören zu den RequestHandlern. In der solrconfig.xml werden die verwendeten SearchHandler definiert und mit Voreinstellungen versehen. Die Definition eines Query Parsers, der durch den SearchHandler verwendet wird, legt fest, wie die übergebene Suchanfrage in ein durch den Solr-Server interpretierbares Query-Objekt gewandelt wird.

In der solrconfig.xml des Solr-Cores slub-kxp werden u.a. der Dismax und der Extended Dismax (eDismax) Query Parser als SearchHandler definiert (vgl. Anhang 2). Der zu verwendende Query Parser kann auch durch den Parameter **defType** übergeben werden.

Die Query Parser Dismax und eDismax bieten verschiedene Erweiterungen im Vergleich zum Lucene Query Parser. Dazu gehört die Möglichkeit, mehrere Felder festzulegen, auf welchen die Suche ausgeführt wird und diese mit unterschiedlichen Boost-Werten zu versehen. Boost-Werte beeinflussen die Berechnung der Relevanz, des Score-Wertes. Auslöser für die Anwendung eines Boost-Wertes ist das Auftreten definierter Merkmale bei einer Suche.

In den folgenden Abschnitten werden einige Funktionen des Dismax Query Parsers erläutert. Sie stehen alle auch im eDismax Query Parser zur Verfügung, der bei der Suche im SLUB-Katalog Anwendung findet.

2.4.2.1 Query Fields – Parameter qf

Der Parameter **qf** erlaubt es, die Suchterme auf mehreren Feldern gleichzeitig anzufragen. Dabei werden die jeweilig für das Feld geltenden Analysevorschriften berücksichtigt. Außerdem können Felder unterschiedlich gewichtet werden. So kann z.B. festgelegt werden, dass ein Treffer im Titelfeld mehr Gewicht hat als ein Treffer im Volltextfeld. [50]

2.4.2.2 Boost Query – Parameter bq

Mittels des Parameters **bq** (Boost Query) kann der Score-Wert für Dokumente beeinflusst werden, welche bestimmte Kriterien erfüllen. Der Suchterm in der Boost Query muss

[50] Klose (2014), S. 74–75

nicht einer der im Suchparameter **q** angegebenen Suchterme sein. Hier können zum Beispiel formale Kriterien, wie die Zugehörigkeit zu einer Unterkollektion oder Art des Dokumententyps den Score-Wert beeinflussen.[51] [52]

2.4.2.3 Boost Function – Parameter bf

Im Parameter **bf** können Funktionen, Function Queries, angegeben werden, die den Score-Wert der Treffer weiter beeinflussen. Daher die Bezeichnung Boost-Funktion.

Functions Queries berechnen Werte, die den Score-Wert beeinflussen anhand von Werten numerischer Felder. Sie nutzen dazu vordefinierte (i.A. mathematische) Funktionen.[53]

2.4.2.4 Phrase Fields – Parameter pf

Der Einsatz des Parameters **pf** (Phrase Fields) kann verwendet werden, um Treffern mehr Gewicht zu geben, die die Suchterme in gleicher Reihenfolge, d.h. als Phrase, beinhalten. Dazu werden, ähnlich wie im Parameter **qf,** Felder mit Boost-Werten aufgelistet. Tritt die Termreihenfolge der Suchanfrage als Phrase in den im Parameter **pf** gelisteten Feldern auf, wird der Score-Wert erhöht.[54], [55]

2.4.2.5 Phrase Slop – Parameter ps

Durch den Parameter **ps** wird der Abstand definiert, den die Suchterme in den im Parameter **pf** festgelegten Feldern haben dürfen. Der Defaultwert ist 0. Durch **ps**=0 wird nur eine exakte Phrase akzeptiert. [56]

2.4.2.6 Minimum Should Match – Parameter mm

Suchterme können obligatorisch, verboten oder optional sein. Obligatorische Suchterme werden durch das Präfix „+“ gekennzeichnet. Sie müssen im Dokument enthalten sein, damit es in die Trefferliste aufgenommen wird. Verbotenen Suchtermen ist ein „-“

[51] https://solr.apache.org/guide/6_6/the-dismax-query-parser.html, zuletzt geprüft am 07.01.2022
[52] Klose (2014), S. 206
[53] https://solr.apache.org/guide/7_7/function-queries.html, zuletzt geprüft am 01.02.2022
[54] Klose (2014), S. 206
[55] https://solr.apache.org/guide/6_6/the-dismax-query-parser.html, zuletzt geprüft am 07.01.2022
[56] Klose (2014), 76 u. 206

vorangestellt. Sie dürfen nicht im Dokument enthalten sein. Ist kein Präfix „+“ oder „-“ angegeben, ist der Suchterm optional.

Durch den Parameter **mm** (Minimum Should Match) kann festgelegt werden, wie viele der optionalen Suchterme das Dokument enthalten muss, damit dieses in die Trefferliste aufgenommen wird. Der Wert kann mit ganzen Zahlen oder prozentual angegeben werden. Beispielsweise bedeutet die Belegung **mm**=3, dass mindestens drei der optionalen Suchterme im Dokument enthalten sein müssen. Bei weniger als drei optionalen Suchtermen müssen alle enthalten sein. [57]

2.4.3 schema.xml

Eine schema.xml wird genutzt, um festzulegen, welche Felder Verwendung finden, welches Format sie haben und wie ihre Inhalte interpretiert werden. Jedem Feld wird ein Feldtyp zugewiesen. Darin wird die Verarbeitung der Inhalte bei der Indexierung und bei der Abfrage festgelegt. Feldtypen gehören zu einer bestimmten Solr-Klasse, die bereits eine Aussage über die Feldinhalte trifft.[58] Es stehen u.a. Klassen für Text-, Zahlen und Datumsfelder zur Verfügung.

Das Beispiel in Abbildung 6 zeigt die Konfiguration des Feldtyps textSpellShingle in der solrconfig.xml der für die Finc-Nutzergemeinschaft genutzten Solr-Installationen. Es gehört zur Klasse der Textfelder (`class="solr.TextField"`).

```
<!-- More advanced spell checking field. -->
<fieldType name="textSpellShingle" class="solr.TextField" positionIncrementGap="100">
  <analyzer type="index">
    <tokenizer class="solr.ICUTokenizerFactory"/>
    <filter class="solr.ICUFoldingFilterFactory"/>
    <filter class="solr.StopFilterFactory" ignoreCase="true" words="stoppwoerter_de-en-kurz.txt"/>
    <filter class="solr.ShingleFilterFactory" maxShingleSize="2" outputUnigrams="false"/>
    <filter class="solr.RemoveDuplicatesTokenFilterFactory"/>
  </analyzer>
  <analyzer type="query">
    <tokenizer class="solr.ICUTokenizerFactory"/>
    <filter class="solr.ICUFoldingFilterFactory"/>
    <filter class="solr.StopFilterFactory" ignoreCase="true" words="stoppwoerter_de-en-kurz.txt"/>
    <filter class="solr.ShingleFilterFactory" maxShingleSize="2" outputUnigrams="false"/>
    <filter class="solr.RemoveDuplicatesTokenFilterFactory"/>
  </analyzer>
</fieldType>
```

Abbildung 6: Auszug schema.xml, Konfiguration Feldtyp textSpellShingle

[57] https://solr.apache.org/guide/6_6/the-dismax-query-parser.html, zuletzt geprüft am 07.01.2022
[58] https://solr.apache.org/guide/7_2/field-type-definitions-and-properties.html, zuletzt geprüft am 25.01.2022

Im vorliegenden Fall werden die gleichen Tokenisierungs- und Filterschritte für die Indexierung `<analyzer type="index">` und die Anfrageverarbeitung `<analyzer type="query">` verwendet. Die wichtigste Aufgabe des ICUTokenizers (Zeile 4 bzw.11) ist die Auftrennung des Inhalts an Wortgrenzen zu Tokens. Danach erfolgt mittels des ICUFoldingFilters (Zeile 5 bzw. 12) eine Unicode-Zeichen-Normalisierung.

Unicode (Universal Character Encoding) ist ein Standard zum Kodieren von Schriftzeichen. Umlaute können in Unicode unterschiedlich dargestellt werden, als zusammengesetztes Zeichen oder als Zeichenfolge. Die unterschiedlichen Darstellungen nach sind Unicode-Standard als identisch anzusehen. Für Suchen ist es effizienter, durch eine vorangestellte Normalisierung entweder alle Zeichen zusammengesetzt oder getrennt zu verarbeiten.[59]

Die StopFilterFactory wendet die Liste stoppwoerter_de-en-kurz.txt (vgl. Anhang 5) zum Entfernen von Stoppwörtern an (Zeile 6 bzw. 13). Mittels des ShingleFilters (Zeile 7 bzw. 14) werden N-Gramme mit einer Höchstlänge von zwei Tokens gebildet. Die Original-Unigramme, die eingehende Folge von Tokens, werden dabei nicht weitergereicht. Die entstandenen Tokens werden zuletzt mit Hilfe des RemoveDuplicatesTokenFilter (Zeile 8 bzw. 15) dedupliziert. Dieser Filter ist sehr gut zur Phrasensuche geeignet.

2.4.3.1 Statische und dynamische Felder

Jedes Solrfeld benötigt eine eigene Definition. In der Definition müssen mindestens der Name, Feldtyp sowie die Angaben „indexed“ und „stored“ (vgl. Tabelle 3) festgelegt werden.

[59] https://www.w3.org/International/questions/qa-html-css-normalization.de, zuletzt geprüft am 25.01.2022

Attribut	Beschreibung	
indexed	Angabe der Indexierung des Feldwertes	
	true	Wert des Feldes ist durchsuchbar. Es existiert ein Index für das Feld.
	false	Wert des Feldes ist nicht durchsuchbar
stored	Angabe zum Abspeichern des ursprünglichen Feldwertes vor der Indexierung.	
	true	Werte des Feldes werden gespeichert und sind dadurch anzeigbar
	false	Werte des Feldes werden nicht gespeichert -> nicht anzeigbar
multiValued	default ist „false“ (keine Mehrfachbelegung möglich), das Attribut muss nicht unbedingt gesetzt werden	
required	Pflichtfeldangabe, default ist „false“ (Feld muss nicht in jedem Datensatz existieren), das Attribut muss nicht unbedingt gesetzt werden	

Tabelle 3: Auswahl wichtiger Attribute zur Felddefinition

Es werden zwei Arten von Feldern unterschieden:

1. **Statische Felder**: Der komplette Name wird in der Definition festgelegt.
   Die Definition beginnt mit <field name= ….

```
23  <field name="spellingShingle" type="textSpellShingle" indexed="true" stored="true" multiValued="true"/>
```

Abbildung 7: Beispieldefinition des statischen Felds spellingShingle

2. **Dynamische Felder:** Ein Basisname wird durch einen der regulären Ausdrücke „_*“ oder „*_“ ergänzt. Auf diesem aufbauend können verschiedene Felder mit gleichen Eigenschaften aber unterschiedlichen Namen verwendet werden. Das verschlankt die schema.xml.
   Die Definition beginnt mit <dynamicField name=„beispiel_*“ …

```
26  <dynamicField name="barcode_*" type="code" indexed="true" stored="true" multiValued="true"/>
```

Abbildung 8: Beispieldefinition des dynamischen Felds barcode

2.4.3.2 CopyFields

In den **CopyField-Definitionen** ist eher eine Art Bildungsvorschrift enthalten. Über die Attribute „source“ (Name des zu kopierenden Feldes) und „dest“ (Name des Zielfeldes) wird festgelegt, der Inhalt welchen Feldes vor der Analyse in ein anderes Feld kopiert wird. Der Inhalt kann dadurch auf unterschiedliche Weise verarbeitet werden. Die Ziel- und die Quellfelder der CopyFields müssen vorher als statische oder dynamische Felder definiert worden sein.

Die Bildungsvorschrift beginnt mit <copyField source=„ …..

```
30  <copyField source="author" dest="spellingShingle"/>
```

Abbildung 9: Bildungsvorschrift des CopyFields spellingShingle

2.4.4 Filter Query – Parameter fq

Filter Queries dienen dazu, die Menge der zurückgegebenen Dokumente durch formale Kriterien zu beschränken. Sie nehmen keinen Einfluss auf das Scoring. Der Parameter **fq** kann in einer Anfrage mehrfach verwendet werden. Dokumente werden nur in die Trefferliste aufgenommen, wenn sie allen Kriterien, die in den Parametern **fq** festgelegt wurden, entsprechen.[60]

2.4.5 Sharding – Parameter shards

Apache Solr bietet die Möglichkeit, zu groß gewordene Indizes auf mehrere Server zu verteilen. Das Aufteilen auf mehrere Server wird Sharding genannt. Die daraus entstehenden Solr-Shards können für Suchanfragen durch eine gemeinsame URL angesprochen werden, indem die Adresse der Solr-Shards in den Parameter **shards** der URL eingetragen wird. Die Anfrage wird auf den eingetragenen Solr-Shards ausgeführt. Dadurch entsteht je Solr-Shard eine Ergebnismenge. Die Ergebnismengen aus den Anfragen an die Shards werden zu einer Ergebnismenge vereinigt und nach dem Score-Wert (oder einem anderen, festzulegenden Sortierparameter) sortiert. Dabei ist zu beachten, dass die Berechnung des Score-Wertes auf Grund der Bedingungen jedes einzelnen Shards erfolgt. Sind die Dokumente nicht gleichmäßig auf den Shards verteilt, können Dokumente, die der Anfrage an sich weniger ähnlich sind (vgl. Kap. 2.2) einen guten Score-Wert erhalten, weil der Suchterm auf dem einen Shard viel seltener ist als auf dem anderen.[61]

[60] https://solr.apache.org/guide/6_6/common-query-parameters.html, zuletzt geprüft am 02.01.2022
[61] Klose (2014), S. 228–229

3 Situationsanalyse

Seit 2015 liegt dem SLUB-Katalog die auf Apache Lucene basierende Suchmaschine Solr zugrunde. Zur Gestaltung ihrer Webseiten nutzt die SLUB das freie Content-Management-System TYPO3. Der SLUB-Katalog greift dabei u.a. auf die Erweiterungen TYPO3-Find zurück. Die Extension TYPO3-Find bietet eine Frontendlösung für Solr-Indizes. Sie wurde 2013 an der SUB Göttingen von Sven-S. Porst und Ingo Pfennigstorf[62] entwickelt.

Der SLUB-Katalog weist unter einer Suchoberfläche z.Z. ca. 73 Millionen digitale und physische Objekte nach (Stand 2.1.2022). Darunter sind neben Büchern, Zeitungen, Zeitschriften, Karten, Noten und Fotos auch Zugriffsdaten für Datenbanken, Volltextsammlungen und die Digitalen Sammlungen der SLUB Dresden. Für Sonderbestände wie z.B. Handschriften werden derzeit noch spezielle Nachweissysteme genutzt.

3.1 finc-Solr-Schema

Die durch die finc-Nutzergemeinschaft und für den SLUB-Katalog verwendeten Solr-Cores basieren auf dem sogenannten finc-Solr-Schema, dessen Grundzüge in einer schema.xml (vgl. Anhang 3) definiert sind. Basis des finc-Solr-Schemas ist das VuFind-Schema.[63] Dieses Schema wurde für die Kooperation der finc-Nutzergemeinschaft um fast 200 Felder erweitert. Eine frei oder für alle Mitglieder der finc-Nutzergemeinschaft zugängliche Dokumentation des finc-Solr-Schemas liegt bisher nicht vor. Für die durch die SLUB verwendeten Felder des finc-Solr-Schemas wurde mit der Dokumentation im Intranet der SLUB begonnen. Das Team Datenmanagement arbeitet stetig an der Vervollständigung und Pflege dieser Dokumentation.

Ein Teil der zusätzlichen Felder beinhaltet Lokalinformationen einzelner Finc-Teilnehmerbibliotheken für Barcodes, Standorte, Signaturen etc. Des Weiteren wurden zusätzliche Facettenfelder z.B. zur Erschließung von Filmen (facet_937) oder Musikalien

[62] https://extensions.typo3.org/extension/find, zuletzt geprüft am 02.01.2022
[63] https://vufind.org/wiki/development:architecture:solr_index_schema, zuletzt geprüft am 06.01.2022

(facet_937d, facet_937e, facet_937f) eingeführt. Hier handelt es sich meist um dynamische Felder. Die Häufigkeit der Belegung indexierter Felder, welche für die SLUB von Bedeutung sind, wurde mittels des LukeRequestHandlers ermittelt und steht im Anhang 10 zur Verfügung.

Auf welche Solrfelder bei der Suche zurückgegriffen wird, ist u.a. auch von deren Feldtyp abhängig. Im Anschluss werden die im finc-Solr-Schema verwendeten Feldtypen kurz vorgestellt, die für die Suche im SLUB-Katalog von Bedeutung sind. Der Feldtyp zu den verwendeten Solrfeldern wird folgend entweder im Text genannt, in Klammern hinter der Feldbezeichnung angegeben oder in einer zugehörigen Tabelle aufgeführt. Ein Überblick kann auch in der Tabelle im Anhang 10 gewonnen werden.

3.1.1 Feldtyp string

Beim Feldtyp string werden die eingegebenen Werte 1:1 übernommen.

	Öko-Haus Walden
DefaultAnalyzer	Öko-Haus Walden

3.1.2 Feldtyp textProper

Bei der Indexierung und bei der Queryanalyse wird die Eingabe durch den ICUTokenizer an Wortgrenzen (Leerzeichen, Kommata, Semikolon) aufgetrennt und in einzelne Tokens zerlegt. Durch den WordDelimiterFilter erfolgen weitere Aufsplittungen bevor durch den ICUFoldingFilter eine Unicode-Zeichen-Normalisierung und ein Lowercasing durchgeführt wird. Dubletten werden durch den RemoveDuplicatesTokenFilter zum Schluss entfernt.

	Däubler-Gmelin, Herta				**Max Frisch**	
ICUTokenizer	Däubler	Gmelin	Herta		Max	Frisch
WordDelimiterFilter	Däubler	Gmelin	Herta		Max	Frisch
ICUFoldingFilter	daubler	gmelin	herta		max	frisch
RemoveDuplicatesTokenFilter	daubler	gmelin	herta		max	frisch

Tabelle 4: Beispielindexierung von Namen mit Feldtyp textProper

	Dresden grüner durch ökologische Übergänge				
ICUTokenizer	Dresden	grüner	durch	ökologische	Übergänge
WordDelimiterFilter	Dresden	grüner	durch	ökologische	Übergänge
ICUFoldingFilter	dresden	gruner	durch	okologische	ubergange
RemoveDuplicatesTokenFilter	dresden	gruner	durch	okologische	ubergange

Tabelle 5: Beispielindexierung mit Feldtyp textProper

3.1.3 Feldtyp text

Beim Feldtyp text werden zusätzlich zu den beim Feldtyp textProper durchgeführten Verarbeitungsschritten Tokenizierung (ICUTokenizer), Aufsplittungen durch den WordDelimiterFilter, Unicode-Zeichen-Normalisierung und ein Lowercasing (ICUFoldingFilter) der KeywordMarkerFilter und der SnowballPorterFilter durchlaufen. Dadurch werden die Tokens auf ihre Wortstammformen zurückgeführt (Bsp.: ökologisch → ökolog). Danach werden Dubletten durch den RemoveDuplicatesTokenFilter entfernt.

	Dresden grüner durch ökologische Übergänge				
ICUTokenizer	Dresden	grüner	durch	ökologische	Übergänge
WordDelimiterFilter	Dresden	grüner	durch	ökologische	Übergänge
ICUFoldingFilter	dresden	gruner	durch	okologische	ubergange
KeywordMarkerFilte	dresden	gruner	durch	okologische	ubergange
SnowballPorterFilter	dresd	grun	durch	okolog	ubergang
RemoveDuplicatesTokenFilter	dresd	grun	durch	okolog	ubergang

Tabelle 6: Beispielindexierung mit Feldtyp text

3.1.4 Feldtyp code

Durch den KeywordTokenizer wird das gesamte Textfeld als ein einziges Token übernommen. Anschließend werden alle Buchstaben in Kleinbuchstaben gewandelt (LowerCaseFilter). In diesem Feld kann nur der komplette String gefunden werden.

	(DE-14)1997 8 024095 001		**Hist.Sax.C.3,misc.2**
KeywordTokenizer	(DE-14)1997 8 024095 001		Hist.Sax.C.3,misc.2
LowerCaseFilter	(de-14)1997 8 024095 001		hist.sax.c.3,misc.2

Tabelle 7: Beispielindexierung mit Feldtyp code

3.1.5 Feldtyp codetokenized

Der WhitespaceTokenizer trennt die eingehenden Werte bei Leerzeichen in einzelne Token. Der LowerCaseFilter wandelt anschließend alle Buchstaben in Kleinbuchstaben um. Alle Satzzeichen bleiben erhalten.

	VD18 10540334-ddd			**Best.-Nr.. RWA 108-20**		
WhitespaceTokenizer	VD18	10540334-ddd		Best.-Nr..	RWA	108-20
LowerCaseFilter	vd18	10540334-ddd		best.-nr..	rwa	108-20

Tabelle 8: Beispielindexierung mit Feldtyp codetokenized

3.1.6 Feldtyp isn

Der PatternTokenizer mit dem Muster `pattern="^(\S*)\s*.*$" group="1"` verarbeitet nur die erste Gruppe von Zeichen, die keine Leerräume (non whitespace character) sind. Der LowerCaseFilter formt alle Buchstaben zu Kleinbuchstaben. Der PatternReplaceFilter löscht anhand des Musters `pattern="[^0-9x]" replacement="" replace="all"` alle Zeichen außer Ziffern und x.

	VD17 14:003620R		**979-0-20650-873-8**		**55.048435.3**
PatternTokenizer	VD17		979-0-20650-873-8		55.048435.3
LowerCaseFilter	vd17		979-0-20650-873-8		55.048435.3
PatternReplaceFilter	17		9790206508738		550484353
LengthFilter	17		9790206508738		550484353

Tabelle 9: Beispielindexierung mit Feldtyp isn

3.1.7 Feldtyp callnumberSearch

Sowohl bei der Indexierung als auch bei der Queryanalyse bleiben Satzzeichen erhalten. Leerzeichen werden durch die Einstellungen des PatternReplaceFilters über das Muster `pattern="(\s)" replacement=""` entfernt. Anschließend wird durch den KeywordTokenizer das gesamte Textfeld als einziges Token übernommen und durch den ICUFoldingFilter erfolgt die Unicode-Zeichen-Normalisierung und das Lowercasing.

	NZ 14720 U53 D7-1		**Hist. Sax. C. 3, misc. 2**
PatternReplaceCharFilter	NZ14720U53D7-1		Hist.Sax.C.3,misc.2
KeywordTokenizer	NZ14720U53D7-1		Hist.Sax.C.3,misc.2
ICUFoldingFilter	nz14720u53d7-1		hist.sax.c.3,misc.2

Tabelle 10: Beispielindexierung mit Feldtyp callnumberSearch

3.2 Katalogkonzept

Eine detaillierte Anforderungserhebung aus der Planungszeit des neuen Katalogs (ca. 2015) hinsichtlich des Relevanzrankings konnte nicht ermittelt werden.

Der SLUB-Katalog bietet seinen Nutzenden in der „Einfachen Suche“ z.Z. neun verschiedene inhaltlich begrenzte Suchfelder und die Standardsuche „Alles“. Bei den Suchfeldern kann z.B. gezielt in den Bereichen Titel, Personen/Institutionen, Schlagworten, Signaturen, Verlagsangaben und Reihen gesucht werden. Bei der Suche „Alles“ werden alle diese Bereiche und zusätzlich weitere, wie z.B. Volltexte, Beschreibungen, Zusammenfassungen und IDs in die Recherche einbezogen. Bis auf das Suchfeld „ISBN/ISSN/ISMN“ können die begrenzten Suchfelder in der „Erweiterten Suche“ miteinander kombiniert werden. Eine Verfeinerung der Suche ist z.B. über die Phrasensuche möglich, indem der Suchbegriff in Hochkommata gesetzt wird. Zur Erweiterung der Treffermenge wird die Rechtstrunkierung mittels * angeboten. Für alle Sucheinstiege können zur Verbesserung der Ergebnisse auch die Booleschen Operatoren AND, OR und NOT eingesetzt werden.

Weitere Verfeinerungen der ursprünglichen Benutzeranfrage sind über 10 Facetten möglich. Angeboten werden folgende Filter:

- **Zugang:** Auswahl zwischen digitalen und physischen Medien
- **Medientyp:** Bücher, Aufsätze, Zeitschriften ….
- **Erscheinungsjahr** (Zeitstrahl)
- **Verfügbarkeit:** OpenAccess, Freihand verfügbar, Magazinbestellung …
- **Standort:** Zweigbibliotheken der SLUB, die ein Exemplar aus der Treffermenge besitzen
- **Sprache**
- **Fachgebiet**
- **Person/Institution**
- **Musikalische Ausgabeform**
- **Kollektion:** (Verbunddaten SWB, Deutsche Fotothek, SpringerOnline …)

Erläuterungen zur Suche finden sich auf der Homepage der SLUB unter https://www.slub-dresden.de/hilfe-zum-katalog/hilfe-suchtipps.

Nach der initialen Einführung 2015 wurde die Konfiguration mehrfach überarbeitet. Dies geschah hauptsächlich auf der Grundlage von Anforderungen aus der bibliothekarischen Auskunftstätigkeit oder durch direkte Nutzeranfragen. Nach wie vor kommt es zu Anfragen, weil erwartete Medien nicht angezeigt werden, das Ranking der Treffer nicht sinnvoll erscheint oder angezeigte Medien nicht oder nur schwer in Zusammenhang mit der Rechercheanfrage gebracht werden können.

3.3 Suchkonfigurationen

Die Kommunikation zwischen Client und Solr erfolgt über HTTP-Requests. Über eine URL wird der Server angesprochen, auf dem die Solr-Installation liegt. Es folgen die Angaben des Standardverzeichnisses solr und des Solr-Cores. Im Anschluss werden die Anforderungsparameter formuliert.

http:// <server> /solr/ <core> / <Anforderung mit Parametern>

Die Parameter der Anforderung rufen als Erstes den RequestHandler auf. Im Falle einer Anfrage geschieht das durch „select?“

```
https://beispielserver.de/solr/slub-kxp/select?<Parameter>
```

Abbildung 10: Beispielaufruf für eine Suche auf dem Solr-Core slub-kxp, „URL-Rumpf“

An diesen „URL-Rumpf“ werden weitere Parameter angehängt, die die Optionen für die Suche festlegen. In Tabelle 11 sind die Parameter genannt, die bei der Suche im SLUB-Katalog verwendet werden.

Die Gestaltung dieser Parameter wird für den SLUB-Katalog durch die TYPO3-Find-Extension gesteuert. Komponenten dafür werden in der Konfiguration der TYPO3-Find-Extension (EXT:slub_katalog_config/Configuration/TypoScript/Plugin, Anhang 7) und weiteren Dateien wie der AdvancedQuery.php (Anhang 8) oder der SearchHandler.php (Anhang 9) festgelegt.

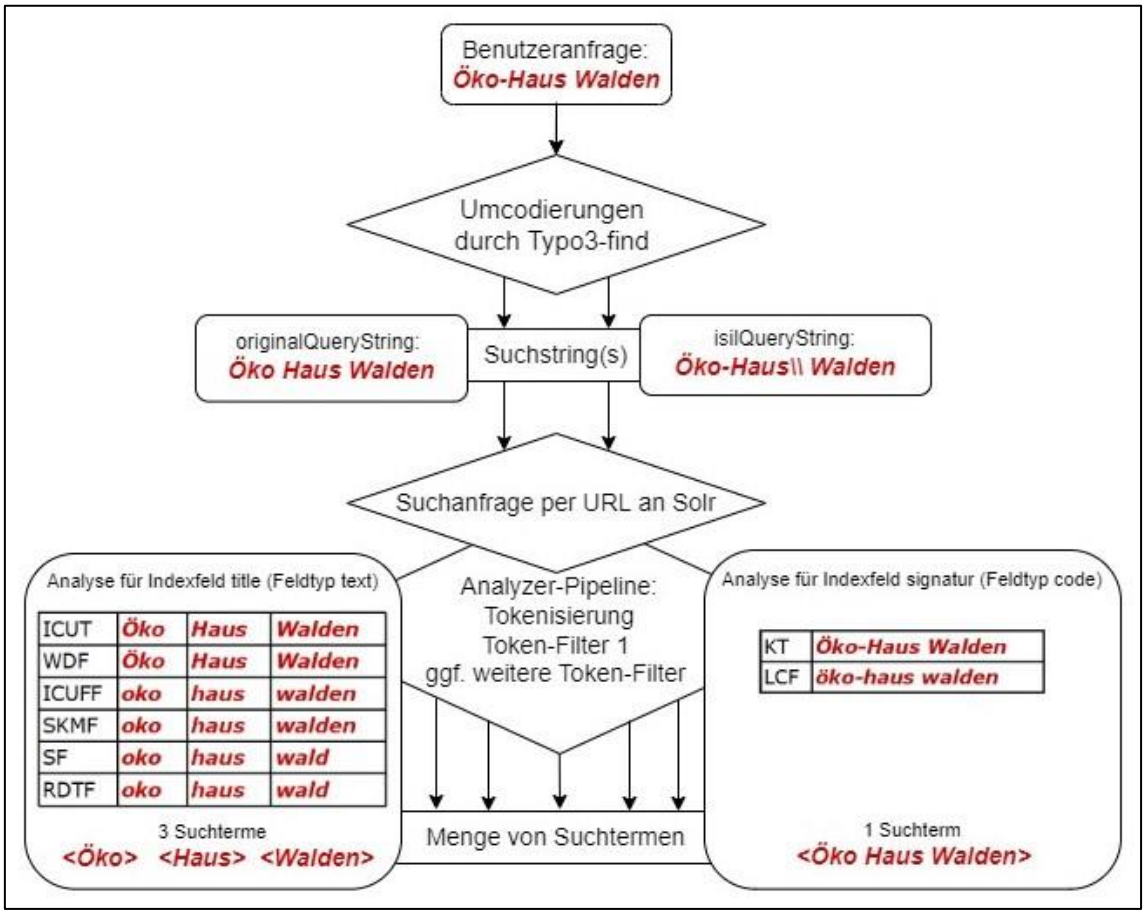


Abbildung 11: Von der Benutzeranfrage zum Suchterm

Am Beispiel der **Benutzeranfrage** <<Öko-Haus Walden>> sind in Abbildung 11 die verschiedenen Verarbeitungsschritte dargestellt, die die **Benutzeranfrage** von der Eingabe in den Suchschlitz des SLUB-Katalogs bis zur Anfrageverarbeitung auf dem Solr durchläuft.

TYPO3-find übersetzt den übergebenen String für die Weitergabe an den Solr in unterschiedliche Suchstrings. Für die Standardsuche „Alles“ im SLUB-Katalog entstehen der originalQueryString und der isilQueryString (vgl. Kapitel 3.3.4).

Der/die **Suchstring**(s) werden im Parameter **q**, der Suchanfrage, innerhalb der URL an den Solr übergeben. Der Query Parser wandelt die eingehende Anfrage in ein für den Solr interpretierbares Query-Objekt. Die Suchstrings durchlaufen dabei eine Analyzer-Pipeline, die aus einem Tokenizer und i.A. einem oder mehreren Filtern besteht. Aus dem Suchstring entsteht dadurch eine Menge von **Suchtermen**.

Bis auf die im Parameter **q** übergebene Suchanfrage, die sich von Suche zu Suche verändert, bleiben die anderen Einstellungen für die Standardsuche im SLUB-Katalog gleich. Das betrifft z.B. die Festlegungen zum Response-Format, zu den verwendeten Shards oder zu der Verwendung der Facetten, aber auch die im Parameter **fq** hinterlegten FilterQueries. Aus den spezialisierten Suchen heraus werden an die URL zusätzlich die Parameter zur Definition des zu verwendenden Query Parsers **defType** und für die Boost Query **bq** gehängt.

Parameterbelegung für SLUB-Katalog	Erläuterung	
omitHeader=true	Wenn „true“, werden verschiedene Informationen des ResponseHeaders unterdrückt	
wt=json	Festlegung des Response-Formats	
json.nl=flat	Steuert das Ausgabeformat von NamedLists. NamedLists finden bei der Facettierung Anwendung.	
q= ……	Enthält den zu bildenden QueryString, Ausführungen in Kapitel 3.3.3 und 3.3.5	
start=0	Definiert den Index des ersten zu zeigenden Treffer	
rows=20	Voreinstellung der pro Seite angezeigten Trefferanzahl	
fl=*, score	Legt fest, welche Felder im Response zurückgegeben werden SLUB-Katalog: alle Felder und Score-Wert	
sort=score+desc	Voreinstellung der Sortierung der Treffermenge, SLUB-Katalog: absteigend nach Score-Wert	
fq= ……	Enthält die FilterQueries, kann mehrfach auftreten Ausführungen in Kapitel 3.3.2	
shards= ……	Enthält die URLs der abgefragten Solr-Shards Ausführungen in Kapitel 3.3.1	
hl=true	Festlegung, ob Highlighting gewünscht	
hl. …	Weitere Festlegung zum Highlighting	
facet=	Festlegung, ob Facetten angeboten werden	
facet. … / **f.** …	Weitere Festlegungen zur Facettierung	
defType=edismax	Legt den SearchHandler für die Boost Query fest	Anwendung in URL nur bei den spezialisierten Sucheinstiegen
bq=	enthält die Boost Query	

Tabelle 11: Übersicht über verwendete Parameter [64] [65] [66] [67] [68]

[64] https://solr.apache.org/guide/6_6/common-query-parameters.html, zuletzt geprüft am 02.01.2022
[65] https://solr.apache.org/guide/6_6/response-writers.html, zuletzt geprüft am 02.01.2022
[66] https://solr.apache.org/guide/6_6/distributed-requests.html, zuletzt geprüft am 02.01.2022
[67] https://solr.apache.org/guide/6_6/highlighting.html, zuletzt geprüft am 02.01.2022
[68] https://solr.apache.org/guide/6_6/faceting.html, zuletzt geprüft am 02.01.2022

Als Highlighting (Parameter **hl**) wird die optische Hervorhebung der gefundenen Suchterme bezeichnet. Dadurch kann verdeutlicht werden, warum ein Treffer in der Trefferliste erscheint.

Bei der Facettierung (Parameter **facet**) werden Ergebnisse anhand bestimmter Merkmale zu Gruppen zusammengefasst und bieten dem Nutzer dadurch gezielte Filtermöglichkeiten.

3.3.1 Parameter shards – die verwendeten Indizes

Der SLUB-Katalog spricht über die URL im Parameter **shards** drei Solr-Cores an, die auf getrennten Solr-Installationen laufen. Es handelt sich hier nicht um Teile eines Index, der auf Grund der Größe geteilt wurde, sondern um eine Aufteilung nach einem formal-inhaltlichen Profil. Der Solr-Core finc-main enthält als größten Baustein Daten aus dem K10plus, aber auch zahlreiche weitere Kollektionen wie Daten aus dem freizugänglichen französischen Online-Archiv Persée oder den Nationallizenzen. Es handelt sich hier vorrangig um Metadaten von selbständigen Publikationen. Der Solr-Core finc-ai, der Artikelindex, enthält die meisten Dokumente. Er verzeichnet hauptsächlich Dokumente auf der Artikelebene, z.B. aus den Datenquellen Crossref, DOAJ (Directory of Open Access Journals) und BASE (Bielefeld Academic Search Engine). Der Solr-Core slub-kxp enthält durch die SLUB prozessierte Daten z.B. aus der Deutschen Fotothek und dem AV-Portal der TIB Hannover.

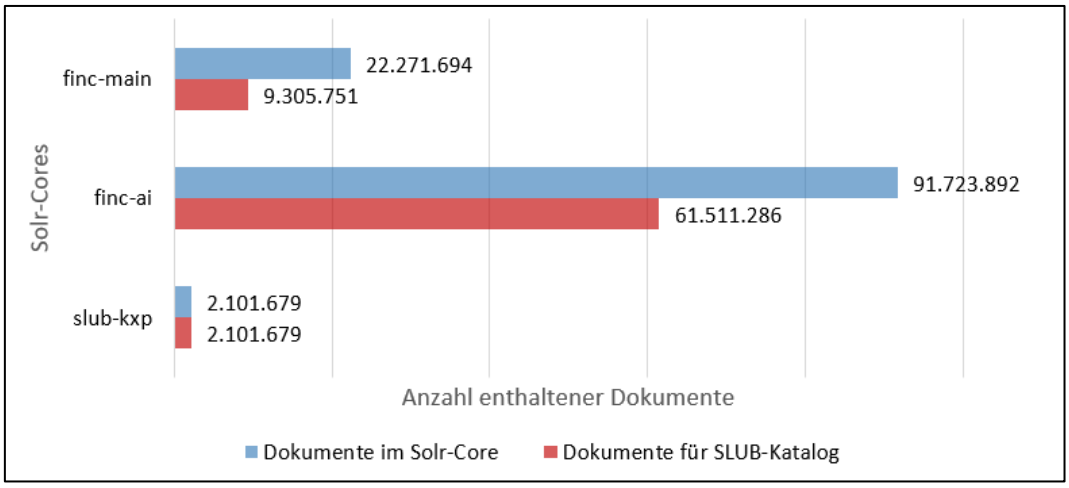


Abbildung 12: Belegung der verwendeten Solr-Cores, Stand 2.1.2022

3.3.2 Parameter fq – Filter Queries

In der Konfiguration der TYPO3-Find-Extension sind unter „additional Filters“ die Angaben für die zu verwendenden Filter Queries definiert. Diese werden in der URL im Parameter **fq** an den Solr übergeben.

```
1486    additionalFilters {
1487        1 = institution:DE-14
1488        2 = format_de14:["" TO *]
1489        3 = !mega_collection:("Genios (Wirtschaftswissenschaften)" OR "Genios (Recht)" OR "Genios (Tech
1490    }
```

Abbildung 13: Auszug aus EXT:slub_katalog_config/Configuration/TypoScript/Plugin, Stand: 19.11.2021

Für die Suche im SLUB-Katalog gelangen nur Dokumente in die Treffermenge, für welche gilt:

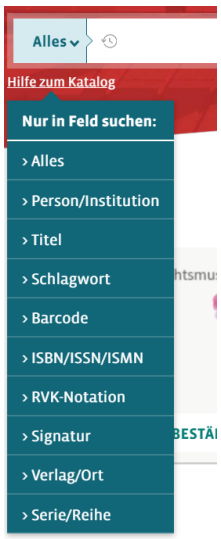
Solrfeld	**Bedingung**	
institution	muss „DE-14“ enthalten	
format_de14	muss belegt sein	
mega_collection	darf folgende Werte nicht enthalten:	Genios (Wirtschaftswissenschaften) Genios (Recht) Genios (Technik) Genios (Sozialwissenschaften) Genios (Psychologie)

Tabelle 12: FilterQueries, EXT:slub_katalog_config/Configuration/TypoScript/Plugin, Stand: 19.11.2021

3.3.3 Sucheinstiege im SLUB-Katalog

Sucheinstiege, welche das Katalog-Frontend „SLUB-Katalog“ nutzen kann, werden im Abschnitt Query Fields der Setups der TYPO3-Find-Extension (Zeile 1772-1912) festgelegt. Verwendet werden für den SLUB-Katalog aktuell die Standardsuche „Alles“ und neun spezialisierte Sucheinstiege (vgl. Abbildung 14 und Tabelle 13). Zusätzlich wird eine erweiterte Suche angeboten.

Die Standardsuche „Alles“ wird in Kapitel 3.3.4, die spezialisierten Sucheinstiege werden im Kapitel 3.3.5 hinsichtlich der verwendeten Felder und Parameter untersucht.



Identifier	Name	Im SLUB-Katalog angeboten als	In erweiterter Suche
10	default	Alles	-
20	author	Person/Institution	ja
30	rvk_facet	RVK-Notation	ja
50	topic	Schlagwort	ja
70	signatur	Signatur	
80	barcode	Barcode	
100	title	Titel	ja
130	ident	ISBN/ISSN/ISMN	ja
130	imprint	Verlag/Ort	ja
140	series2	Serie/Reihe	ja

Abbildung 14: Ausschnitt aus dem SLUB-Katalog
Tabelle 13: aktuell im SLUB-Katalog verwendete Sucheinstiege

3.3.4 Standardsuche

Im Abschnitt „components“ der Konfiguration der TYPO3-Find-Extension (Anhang 7) werden Festlegungen zu den Bestandteilen des originalQueryString für die Standardsuche getroffen.

Festgelegt werden der eDismax als DismaxHandler (Zeile 1493) und der Wert für Minimum Should Match (vgl. Kapitel 3.2.5) wird durch **mm**=3 definiert (Zeile 1496-1497).

Die im Parameters **bq** für den SLUB-Katalog getroffenen Festlegungen haben eher formalen Charakter (vgl. Abbildung 15). Dokumente, die im Feld *access_facet* die Belegung „Electronic Resources“ haben, erhalten keinen Boost, dagegen wird die Belegung mit „Local Holdings“ stark mit dem Wert 50.000 geboostet.

Es ist zu beobachten, dass bei vielen elektronischen Ressourcen aus dem finc-ai (z.B. aus der Datenquelle Crossref) nur spärliche Metadaten vergeben sind. Da Felder mit viel Inhalt, also lange Felder, weniger Einfluss auf den Score-Wert haben als kürzere Felder, würden ohne diesen Boost für die Zugangsart „Local Holdings“ vielfach schlecht beschriebene Dokumente aus elektronischen Kollektionen die oberen Rankingplätze bei

Anfragen einnehmen, während die ausführlich beschriebenen Dokumente aus den Bibliothekskatalogen die hinteren Plätze belegen.

Eine Aufwertung erfahren alle Medientypen (Feld *format_de14*) außer dem Medientyp Nachlass.

```
1503            30 {
1504                name = bq
1505                value = (access_facet:"Electronic Resources"^0 OR access_facet:"Local Holdings"^50000.0) -format_de14:"Nachlass"^999
1506            }
```

Abbildung 15: Parameter bq, Auszug aus EXT:slub_katalog_config/Configuration/TypoScript/Plugin, Stand: 19.11.2021

Die Belegung des Parameters **qf** (Query Fields) wird im Bereich DismaxFields (Zeile 1509-1544) vorbereitet. Zu verwendende Felder und deren Boost erhalten für die weitere Verarbeitung durch die AdvancedQuery.php IDs.

ID	Feld	Boost-Wert	inhaltliche Beschreibung des Feldes
100	title_full_unstemmed	1950	vollständige Titelaussage inklusive Autorenangabe/Verfasserwiederholung usw.
110	title_alt		alternative Titel (jede Form davon, inkl. Paralleltitel)
120	title_short	500	Kurztitel
200	author	750	Hauptautoren, die wichtigsten Autoren
210	author2	500	Nebenautoren bzw. Autoren mit Nebeneintragungen
220	author_corporate	200	Körperschaft als Autor, auch weitere Körperschaften
230	author_corporate2	200	weitere Körperschaften mit Nebeneintragungen
240	author_ref		Varianten eines Namens aus den Normdaten
320	publishPlace		Erscheinungsort
333	publisher		Verlag
410	marc024a_ct_mv		Produktcodes, Standardnummern, DOIs, VD16,-17,-18-Angaben
411	marc535_cns_mv		Signatur des Originals der Reproduktion
420	marc028a_ct_mv		Verlags-, Produktions- und Bestellnummer
421	callnumber_de14		Signaturen SLUB Dresden, Grundsignatur aus Lokalsystem

*Tabelle 14: Auszug aus EXT:slub_katalog_config/Configuration/TypoScript/Plugin, Identifier <800, **Teil 1,** Stand: 19.11.2021*

ID	Feld	Boost-Wert	inhaltliche Beschreibung des Feldes
600	topic_ref		Schlagwörter in finc-Quellen werden über UBL-Anreicherung in MARC 950a ergänzt
650	series2		Titel der Serie(n)
660	music_heading		Besetzung, Form und Gattung von Musikalien
700	description	0,000001	Beschreibende Zusammenfassung des Objekts
710	contents	0,000001	Details, Inhaltsverzeichnis
720	footnote	0,000001	Fußnoten und sonstige Anmerkungen
750	publishDateSort		Wert aus publishDate (Erscheinungsjahr), nur der erste dort angegebene Wert

*Tabelle 15: Auszug aus EXT:slub_katalog_config/Configuration/TypoScript/Plugin, Identifier <800, **Teil 2,** Stand: 19.11.2021*

ID	Feld	Boost-Wert	inhaltliche Beschreibung des Feldes
1100	isbn		Internationale Standardbuchnummer, nur die gültige ISBN
1200	issn		Internationale Standardnummer für fortlaufende Sammelwerke, nur gültige ISSN
1230	ismn		Internationale Standardmusiknummer
1240	rsn_id_str_mv		RSN, ID aus Lokalsystem mit Sigel-Präfix Präfix für SLUB: (DE-14)
1250	barcode		Barcodes aus Lokalsystem mit Sigel-Präfix Präfix für SLUB: (DE-14)
1260	signatur		Signatur aus Lokalsystem mit Sigel-Präfix Präfix für SLUB: (DE-14)
1270	record_id		Lieferanten-Identifier (original ID aus der Quelle)
1280	zdb		ID in der ZDB
1290	swb_id_str		SWB-PPN, ehemalige ID im SWB
1291	kxp_id_str		KXP-PPN, ID im K10plus
1295	finc_id_str		IDs aus alten Versionen; u.a. für Link-Resolver alter Permalinks
1330	geographic		geografische Schlagwörter
1430	collection		Kürzel oder Name der Sammlung, die diesen Original-Datensatz enthält
1730	fulltext	0,000001	Volltext

Tabelle 16: Auszug aus EXT:slub_katalog_config/Configuration/TypoScript/Plugin, Identifier >800, Stand: 19.11.2021

Durch die Funktion `cleanParameter` der AdvancedQuery.php werden in der übergebenen Benutzeranfrage verschiedene Satz- und Sonderzeichen durch Blanks ersetzt, bevor der Suchstring in den originalQueryString eingefügt wird.

```
133     private function cleanParameter($queryParameter) {
134             return str_replace([':','?', ';', '-', '!', '&', '-', '(', ')', '+', '=', '$', '[', ']', '.', ',', '"', '‘', '’'],' ',$queryParameter);
135     }
```

Abbildung 16: Funktion cleanParameter, Auszug aus AdvancedQuery.php, Stand: 18.06.2021

Ausnahme bilden hier Benutzeranfragen, die in Anführungszeichen gesetzt wurden. Für die Phrasensuche (s. Kapitel 3.3.4.1) wird die Eingabe zwischen den Anführungszeichen als ein einziger Term behandelt und unverändert übergeben.

Letzter Baustein für die Standardsuche ist der isilQueryString. Er wird in Zeile 1916 bis 1926 definiert und enthält sieben weitere eDismaxQueries, welche mit OR an den originalQueryString angeschlossen und auch untereinander mit OR verbunden werden. Für den isilQueryString wird die eingegebene Benutzeranfrage anders vorverarbeitet. Vor dem Einfügen des Suchstrings in die eDismaxQueries anstelle des Platzhalters **%1$s** (vgl. Abbildung 17) des isilQueryStrings werden lediglich Leerzeichen und runde Klammern dreifach maskiert.

```
(_query_:"{!edismax qf=\"barcode rsn_id_str_mv\"}\\(DE-14\\)%1$s*")
OR (_query_:"{!edismax qf=\"signatur\"}\\(DE-14\\)%1$s")^10000 OR
(_query_:"{!edismax qf=\"callnumber_de14\"}%1$s")^10000 OR
(_query_:"{!edismax qf=\"marc024a_ct_mv\"}%1$s") OR
(_query_:"{!edismax qf=\"marc028a_ct_mv\"}%1$s") OR
(_query_:"{!edismax qf=\"marc535_cns_mv\"}%1$s") OR
(_query_:"{!edismax qf=\"issn\"}%1$s")^500000
```

Abbildung 17: :Einstellungen isilQueryString, Auszug aus EXT:slub_katalog_config/Configuration/TypoScript/Plugin, Stand 19.11.2022

Für den isilQueryString werden aktuell folgende Felder in die Abfrage einbezogen:

Feldname	Feldtyp	Beschreibung	Präfix	Boost-Wert
barcode	code	Barcodes aus Lokalsystem	(DE-14)	
rsn_id_str_mv	String	RSN, ID im Lokalsystem		
signatur	code	Signatur aus Lokalsystem	(DE-14)	10.000
callnumber_de14	code	Signaturen SLUB Dresden, Grundsignatur aus Lokalsystem		10.000
marc024a_ct_mv	codetokenized	Produktcodes, Standardnummern, DOIs, VD16,-17,-18-Angaben		
marc028a_ct_mv	codetokenized	Verlags-, Produktions- und Bestellnummer		
marc535_cns_mv	callnumberSearch	Signatur des Originals der Reproduktion		
issn	isn	ISSN		500.000

Tabelle 17: im isilQueryString verwendete Felder mit Feldtyp, Beschreibung, Präfix und Boost-Werten

Die Bildung des QueryStrings in Parameter **q** hängt von der Art der eingegebenen Benutzeranfrage ab. Im Folgenden werden die drei möglichen Varianten beschrieben.

3.3.4.1 Phrasensuche

Steht die gesamte Benutzeranfrage in Anführungszeichen, wird der originalQueryString aus den Solrfeldern gebildet, deren IDs kleiner als 800 (Tabelle 14 und Tabelle 15) sind. Der isilQueryString findet keine Anwendung.

```
q=(_query_:"{!edismax qf=\"title_full_unstemmed^1950 title_alt
title_short^500 author^750 author2^500 author_corporate^200
author_corporate2^200 author_ref publishPlace publisher
marc024a_ct_mv marc535_cns_mv marc028a_ct_mv callnumber_de14
topic_ref series2 music_heading description^0.000001
contents^0.000001 footnote^0.000001 publishDateSort\" mm=\'3\'
bq=\'(access_facet:\"Electronic Resources\"^0 OR
access_facet:\"Local Holdings\"^50000.0) -
format_de14:\"Nachlass\"^999\'}\"August der Starke\"")
```

Abbildung 18: Bsp. Phrasensuche nach <<„August der Starke“>>

Die Benutzeranfrage wird dabei ohne Änderung inklusive der Anführungszeichen übernommen. Im Suchstring werden die Anführungszeichen maskiert.

3.3.4.2 Benutzeranfrage beginnt mit einem Wort

Beginnt die Benutzeranfrage mit Buchstaben oder nur einer Ziffer (die Art der folgenden Terme ist unerheblich), wird der originalQueryString aus den Solrfeldern gebildet, deren IDs kleiner als 800 (Tabelle 14 und Tabelle 15) sind. Der in Zeile 1916 bis 1926 definierte isilQueryString wird im Parameter **q** angehängt.

```
q=(_query_:"{!edismax qf=\"title_full_unstemmed^1950 title_alt
title_short^500 author^750 author2^500 author_corporate^200
author_corporate2^200 author_ref publishPlace publisher
marc024a_ct_mv marc535_cns_mv marc028a_ct_mv callnumber_de14
topic_ref series2 music_heading description^0.000001
contents^0.000001 footnote^0.000001 publishDateSort\" mm=\'3\'
bq=\'(access_facet:\"Electronic Resources\"^0 OR
access_facet:\"Local Holdings\"^50000.0) -
format_de14:\"Nachlass\"^999\'}Schloß Dresden August") OR
(_query_:"{!edismax qf=\"barcode rsn_id_str_mv\"}\\(DE-14\\)Schloß\\
Dresden\\ August*") OR (_query_:"{!edismax qf=\"signatur\"}\\(DE-
14\\)Schloß\\ Dresden\\ August")^10000 OR (_query_:"{!edismax
qf=\"callnumber_de14\"}Schloß\\ Dresden\\ August")^10000 OR
(_query_:"{!edismax qf=\"marc024a_ct_mv\"}Schloß\\ Dresden\\
August") OR (_query_:"{!edismax qf=\"marc028a_ct_mv\"}Schloß\\
Dresden\\ August") OR (_query_:"{!edismax
qf=\"marc535_cns_mv\"}Schloß\\ Dresden\\ August") OR
(_query_:"{!edismax qf=\"issn\"}Schloß\\ Dresden\\ August")^500000
```

Abbildung 19: Bsp. Suche nach <<Schloß Dresden August>>

3.3.4.3 Benutzeranfrage beginnt mit einer Zahl

Beginnt die Benutzeranfrage mit mindestens zwei Ziffern (die Art der folgenden Terme ist unerheblich), werden alle DismaxFields (Tabelle 14, Tabelle 15 und Tabelle 16) für den originalQueryString verwendet und zusammen mit dem isilQueryString im Parameter **q** zur Übergabe an den Solr in die URL eingefügt.

```
q=(_query_:"{!edismax qf=\"title_full_unstemmed^1950 title_alt
title_short^500 author^750 author2^500 author_corporate^200
author_corporate2^200 author_ref publishPlace publisher
marc024a_ct_mv marc535_cns_mv marc028a_ct_mv callnumber_de14
topic_ref series2 music_heading description^0.000001
contents^0.000001 footnote^0.000001 publishDateSort isbn issn ismn
rsn_id_str_mv barcode signatur record_id zdb swb_id_str kxp_id_str
finc_id_str geographic collection fulltext^0.000001
publishDateSort\" mm=\'3\' bq=\'(access_facet:\"Electronic
Resources\"^0 OR access_facet:\"Local Holdings\"^50000.0) -
format_de14:\"Nachlass\"^999\'}0519 4555 Relikten") OR
(_query_:"{!edismax qf=\"barcode rsn_id_str_mv\"}\\(DE-14\\)0519-
4555\\ Relikten*") OR (_query_:"{!edismax qf=\"signatur\"}\\(DE-
14\\)0519-4555\\ Relikten")^10000 OR (_query_:"{!edismax
qf=\"callnumber_de14\"}0519-4555\\ Relikten")^10000 OR
(_query_:"{!edismax qf=\"marc024a_ct_mv\"}0519-4555\\ Relikten") OR
(_query_:"{!edismax qf=\"marc028a_ct_mv\"}0519-4555\\ Relikten") OR
(_query_:"{!edismax qf=\"marc535_cns_mv\"}0519-4555\\ Relikten") OR
(_query_:"{!edismax qf=\"issn\"}0519-4555\\ Relikten")^500000
```

Abbildung 20: Bsp. Query Suche nach <<0519-4555 Relikten>>

3.3.5 Spezialisierte Sucheinstiege

Der SLUB-Katalog bietet z.Z. neun spezialisierte Sucheinstiege an (vgl. Kapitel 3.3.3). Da diese teilweise für die erweiterte Suche verwendet werden, kann die Boost Query **bq** hier nicht in der Query enthalten sein. Sie würde sonst bei der erweiterten Suche mehrfach ausgeführt. Deshalb wird sie bei den spezialisierten Sucheinstiegen, ähnlich wie die FunctionQuery **fq**, jeweils außerhalb des Parameters **q** übergeben. Für die Boost Query **bq** muss zusätzlich auch der eDismax als SearchHandler im Parameter **defType** festgelegt werden. Es wird die gleiche Boost Query **bq** wie in der Standardsuche verwendet (vgl. Kapitel 3.3.3).

3.3.5.1 Sucheinstieg Person/Institution

Für die Suche nach Person/Institution sollen mittels des Parameters **qf** mehrere Felder mit unterschiedlichen BoostFaktoren in die Query aufgenommen werden. Deshalb wird hier bereits in der Query der eDismax als SearchHandler definiert.

```
_query_:"{!edismax qf=\"author^300 author_id^100
author_corporate^300 author2^300 author_corporate2^300
author_fuller^150 author_ref^500 author_corporate_ref^500
author_orig^300 author2_orig^300 author_corporate_orig^300
author_corporate2_orig^100\"}%s"
```

Abbildung 21: Sucheinstieg Person/Institution, Queryeinstellung aus EXT:slub_katalog_config/Configuration/TypoScript/Plugin, Stand: 19.11.2021

Den höchsten Boost erhalten die Felder mit Angaben aus Normdaten (*author_ref*, *author_corporate_ref*), den niedrigsten die Felder, die die Namen in Originalschrift oder die ID aus der Gemeinsamen Normdatei verzeichnen.

Solrfeld	Boost	Feldtyp	Feldinhalt
author	300	textProper	Hauptautoren, die wichtigsten Autoren
author_ref	500	textProper	Varianten eines Namens aus den Normdaten
author_orig	300	textProper	Autor in Originalschrift
author_fuller	150	textProper	Vollständige Form des/der Namen(s) des Hauptautors
author2	300	textProper	Nebenautoren bzw. Autoren mit Nebeneintragungen
author2_orig	300	textProper	weitere Autorennamen in Originalschrift
author_corporate	300	textProper	Körperschaft als Autor, auch weitere Körperschaften
author_corporate_ref	500	textProper	Verweisungsform aus den Normdaten für Körperschaften und Kongresse
author_corporate_orig	300	textProper	Körperschaft als Autor in Originalschrift
author_corporate2	300	textProper	weitere Körperschaften mit Nebeneintragungen
author_corporate2_orig	100	textProper	weiter Körperschaft als Autor in Originalschrift
author_id	100	String	GND-ID der Verfasser

Tabelle 18: Felder der aktuellen Suche Personen/Institutionen mit BoostFaktor, Feldtyp und Beschreibung

Bis auf das Feld *author_id* (String) sind alle verwendeten Felder vom Feldtyp textProper, der im Kapitel 3.1.2 erläutert wurde. Bei diesem Feldtyp werden alle Namensteile einzeln indexiert (vgl. Tabelle 19).

eingehende Werte:	**Frisch, Max**			**Max Frisch**	
ICUTokenizer	Frisch	Max		Max	Frisch
WordDelimiterFilter	Frisch	Max		Max	Frisch
ICUFoldingFilter	frisch	max		max	frisch
RemoveDuplicatesTokenFilter	frisch	max		max	frisch

Tabelle 19: Beispielindexierung <<Frisch, Max>> und <<Max Frisch>> mit Feldtyp textProper

Aus diesem Grund werden bei der Suche in Personen/Institutionen nach „Manfred Frisch“ auch Medien angezeigt, die als Personen „Max Frisch“ und „Manfred Meschede“ o.ä. enthalten (vgl. Abbildung 22).

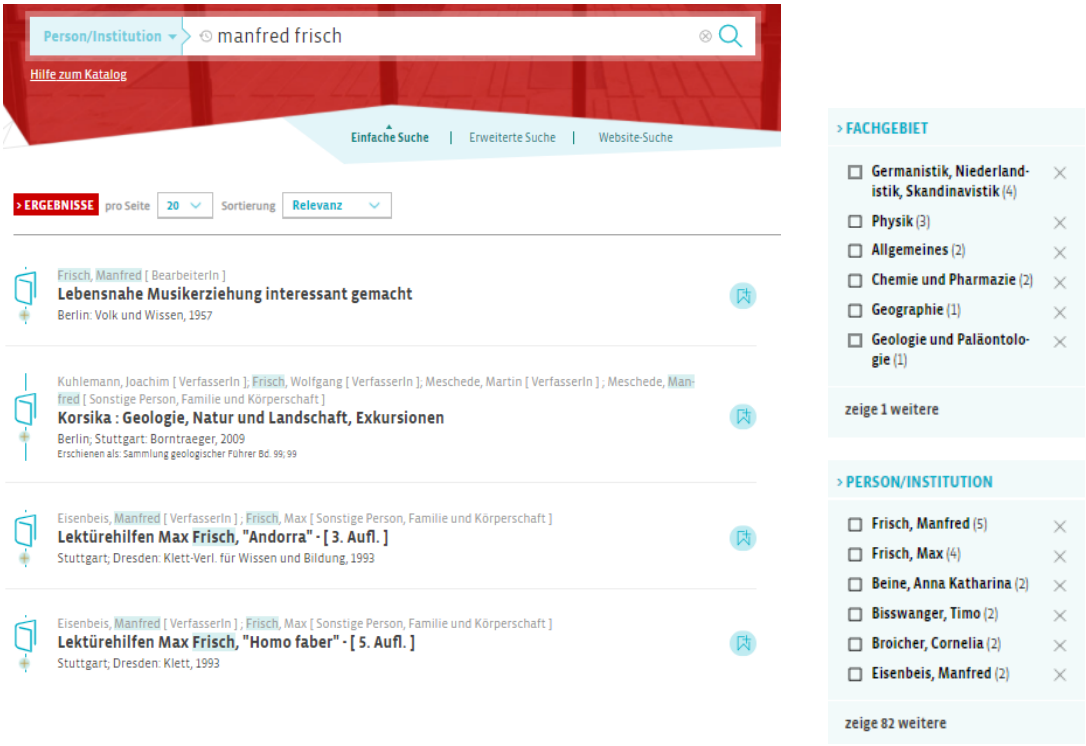


Abbildung 22: Ausschnitt Trefferliste Personensuche <<manfred frisch>> im SLUB-Katalog mit Facettenangebot

Die gesuchte Person bzw. Institution kann über die Facette genau ausgewählt werden.

3.3.5.2 Sucheinstieg Titel

Titel können sich aus mehreren Bestandteilen (Hauptsachtitel, Zusatz zum Sachtitel) zusammensetzen. Des Weiteren gibt es Neben- und Paralleltitel, die bei der Suche berücksichtigt werden müssen.

```
query = {!edismax qf=\"title_full_unstemmed title_alt\"}%s
```

Abbildung 23: Sucheinstieg Titel, Queryeinstellung aus EXT:slub_katalog_config/Configuration/TypoScript/Plugin, Stand: 19.11.2021

Aktuell werden die Felder *title_full_unstemmed* (textProper) und *title_alt* (text) für die Suche verwendet. Keines der beiden Felder wird durch einen Boost besonders berücksichtigt. Im Feld *title_full_unstemmed* ist der vollständige Titel inklusive der Verantwortlichkeitsangabe enthalten. Das Feld *title_alt* enthält zusätzliche Titel und Titelinformationen.

3.3.5.3 Sucheinstieg Schlagwort

Die Schlagwortsuche wird auf dem Feld *topic_unstemmed* (textProper) durchgeführt, welches die Originalform der Schlagwörter ohne Stammformreduktion enthält.

```
query = topic_unstemmed:%s
```

Abbildung 24: Sucheinstieg Schlagwort, Queryeinstellung aus EXT:slub_katalog_config/Configuration/TypoScript/Plugin, Stand: 19.11.2021

3.3.5.4 Sucheinstieg Barcode

Barcodes im bibliothekarischen Kontext dienen der Identifizierung jeweils eines Exemplars eines Bibliotheksbestandes, z.B. zur Verbuchung von Ausleihdaten. Das Feld *barcode* (code) wird bei der Prozessierung der K10plus-Daten für einige finc-Bibliotheken mit Daten aus dem Lokalsystem Libero angereichert. Zur Identifizierung der bibliotheksspezifischen Barcodes wird je nach Herkunft als Präfix das Sigel der besitzenden Bibliothek in Klammern vor den Wert des Barcodes gesetzt. Barcodes der SLUB beginnen im Feld *barcode* deshalb grundsätzlich mit dem Präfix „(DE14)“.

Zur gezielten Barcodesuche wird eine Abfrage auf das Feld *barcode* gestartet. Dem Suchstring muss zwingend das Präfix „(DE-14)“ vorangestellt werden. In diesem Feld kann nur der komplette String gefunden werden.

```
query = barcode:"(DE-14)%s"
```

Abbildung 25: Sucheinstieg Barcode, Queryeinstellung aus EXT:slub_katalog_config/Configuration/TypoScript/Plugin, Stand: 19.11.2021

3.3.5.5 Sucheinstieg ISBN/ISSN/ISMN

Die Suche nach International Standard Numbers für Bücher, Serien oder Musikalien (ISBN, ISSN, ISMN) erfolgt über eine eDismax-Abfrage der Felder *isbn, ismn* und *issn*. Alle Felder haben den Feldtyp isn. Keines der Felder wird geboostet.

```
query = {!edismax qf=\"isbn issn ismn\"}%s
```

Abbildung 26: Sucheinstieg ISBN/ISSN/ISMN, Queryeinstellung aus EXT:slub_katalog_config/Configuration/TypoScript/Plugin, Stand: 19.11.2021

Teilweise gehen in das Feld *ismn* auch andere Standardnummern ein. Gründe sind z.B. Katalogisierungsfehler und die Prozessierungsregeln. Durch die für den Feldtyp isn festgelegten Analyseregeln werden dann ggf. nur Bruchstücke der eigentlichen Inhalte des Feldes übernommen. Im ersten Beispiel der Tabelle 9 ist zu sehen, welchen Indexeintrag eine VD17-Nummer durch die Indexierung in einem Feld des Typs isn erhält.

3.3.5.6 Sucheinstieg RVK-Notation

Das Feld *rvk_facet* (code) enthält die RVK-Notation als Text. Die SLUB nutzt die RVK-Notation als Aufstellungssystematik des Freihandbestandes.

```
query = rvk_facet:%s
```

Abbildung 27: Sucheinstieg RVK-Notation, Queryeinstellung aus EXT:slub_katalog_config/Configuration/TypoScript/Plugin, Stand: 19.11.2021

RVK-Notationen können auf Grund des Feldtyps code nur als kompletter String gesucht werden.

3.3.5.7 Sucheinstieg Signatur

Die Anfrage nach der Signatur erfolgt auf die Solrfelder *signatur* (code), *callnumber_de14* (code) und *marc535_cns_mv* (callnumberSearch) ohne Boost für ein bestimmtes Feld.

```
query = signatur:\(DE-14\)%1$s OR callnumber_de14:%1$s OR
marc535_cns_mv:%1$s
```

Abbildung 28: Sucheinstieg Signatur, Queryeinstellung aus EXT:slub_katalog_config/Configuration/TypoScript/Plugin, Stand: 19.11.2021

Das Feld *signatur* wird bei der Prozessierung der K10plus-Daten für die SLUB und die UBL mit Daten aus dem Lokalsystem Libero angereichert. Zur Identifizierung der bibliotheksspezifischen Signaturen wird je nach Herkunft als Präfix das Sigel der besitzenden Bibliothek in Klammern vor den Wert der Signatur gesetzt. Signaturen der SLUB beginnen deshalb grundsätzlich mit „(DE14)“. Dem Suchstring muss deshalb für eine Anfrage auf dem Feld *signatur* das Präfix „(DE14)“ vorangestellt werden. Die Möglichkeit zur Suche nach Grundsignaturen über das Feld *callnumber_de14* ist zum Auffinden von Zeitschriften von Bedeutung. Das Feld callnumber_de14 ist ein dynamisches, speziell für die SLUB indexiertes Feld. Erkennbar ist das am Postfix „_de14“ der Feldbezeichnung. Das Feld *marc535_cns_mv* enthält bei Reproduktionen die Signatur des Originals.


```
{
  "id": "0-216742951",
  "callnumber_de14": [
    "1997 8 024095"
  ],
  "marc535_cns_mv": [
    "96 T 1435"
  ],
  "signatur": [
    "(DE-14)1997 8 024095 001",
    "(DE-14)1997 8 024095"
  ]
},
```


Abbildung 29: Ausschnitt aus Datensatz id 0-216742951, Feldbelegung callnumber_de14, marc535_cns_mv, signatur

3.3.5.8 Sucheinstieg Verlag/Ort

Unter dem Sucheinstieg Verlag/Ort kann speziell nach Verlagen oder Verlagsorten recherchiert werden. Dafür werden die Felder *publisher* und *publishPlace* mittels eDismaxSearchHandler abgefragt. Beide Felder sind vom Feldtyp textProper.

```
query = {!edismax qf=\"publisher publishPlace\"}%s
```

Abbildung 30: Queryeinstellung aus EXT:slub_katalog_config/Configuration/TypoScript/Plugin, Stand: 19.11.2021

3.3.5.9 Sucheinstieg Serie/Reihe

Die Abfrage erfolgt auf dem Feld *series2* vom Feldtyp text. Das Feld enthält laut Beschreibung die Titel von Serien und die zugehörige Zählung.

```
query = {!edismax qf=\"series2\"}\"%s\"
```

Abbildung 31: Queryeinstellung aus EXT:slub_katalog_config/Configuration/TypoScript/Plugin, Stand: 19.11.2021

4 Möglichkeiten zu Verbesserung der Standardsuche „Alles“

Bei der Standardsuche „Alles“ wird viel Gewicht auf die Erfüllung von „known-item searches“, dem Suchen und Finden bereits bekannter Dokumente gelegt. Das spiegelt sich u.a. im hohen Boosting für die Felder *title_full_unstemmed* und *issn* wieder. Den Nutzenden der SLUB soll durch die Standardsuche für alle Benutzeranfragen der „passende“ Treffer als erster angezeigt werden, auch auf Benutzeranfragen zu Signaturen, Barcodes, Standardnummern und Personen. Das für diese speziellen Suchen gesonderte Sucheinstiege bereitstehen, wird den Nutzenden nicht aktiv vermittelt.

Abbildung 32: Screenshot https://www.slub-dresden.de/, 20.01.2022

In Kapitel 3 wurden Untersuchungen zum aktuellen Stand des SLUB-Katalogs gemacht. Als Grundlage dazu dienten die Beschreibungen der verwendeten Feldtypen in den Kapiteln 3.1.1 bis 3.1.7 und die Untersuchung der Belegungshäufigkeit der Felder (Anhang 10). Kapitel 3.3 untersuchte die Suchkonfiguration der Standardsuche und der speziellen Sucheinstiege.

Anhand dieser Analysen werden in diesem Kapitel Vorschläge für Veränderungen in den Einstellungen der Standardsuche angeboten.

4.1 FilterQueries – nicht Benötigtes ausschließen

Im SLUB-Katalog sollen nur Medien abgebildet werden, die für die SLUB angesigelt[69] sind, d.h., das Feld *institution* enthält „(DE-14)“. Außerdem sollen nur Medien dargestellt

[69] Ansigeln: Bibliothekssigel sind eindeutige Identifikatoren für Bibliotheken. Die Kennzeichnung in gemeinsamen Nachweissystemen, dass eine Bibliothek ein Exemplar des Titels besitzt bzw. einen bestimmten Datensatz verwenden möchte, wird als ansigeln bezeichnet.

werden, bei welchen das Format im Frontend angezeigt werden kann. Das bedeutet, dass das Feld *format_de14* belegt ist. Die beiden ersten Einstellungen der FilterQueries **fq** werden deshalb übernommen (vgl. Kapitel 3.3.2).

Die durch die SLUB genutzten Solr-Cores enthalten aktuell Daten aus fast 40 Datenquellen (u.a. K10plus, Persée, Nationallizenzen, Deutsche Fotothek). Die Datenquellen können über das Feld *source_id* identifiziert werden. Das Feld *mega_collection* enthält entweder die Bezeichnung der Datenquelle oder eine differenzierende Angabe, falls es zur Datenquelle Unterkollektionen gibt.

Aktuell sollen über die Einstellungen in der FilterQuery **fq** fünf Unterkollektionen aus der Trefferliste ausgeschlossen werden. Die Einstellung bezieht sich auf ein zeitlich begrenztes Anzeigeproblem aus dem Jahr 2021 und kann gelöscht werden, da es in *mega_collection* keine Belegungen mehr mit diesen Bezeichnungen gibt.

```
    additionalFilters {
        1 = institution:DE-14
        2 = format_de14:["" TO *]
        3 = !mega_collection:("Genios (Wirtschaftswissenschaften)"
OR "Genios (Recht)" OR "Genios (Technik)" OR "Genios
(Sozialwissenschaften)" OR "Genios (Psychologie)")
    }
```

Abbildung 33: Vorschlag für additionalFilters in den EXT:slub_katalog_config/Configuration/TypoScript/Plugin

4.2 DismaxParameter

4.2.1 Mindestanzahl enthaltener Suchterme – Minimum Should Match mm

Die Einstellungen des für die Suche im SLUB-Katalog verwendeten SearchHandlers haben zu Folge, dass alle Suchterme als obligatorisch gelten. Diese Behandlung wird nur geändert, wenn in der Benutzeranfrage Terme mittels der booleschen Operatoren OR oder „NOT“ verbunden werden.

Da der Parameter mm nur für Suchterme Anwendung findet, welche optional sind, hat der Parameter mm unter den aktuellen Bedingungen nur Auswirkung bei Benutzeranfragen, die mehrere Terme mit OR verbinden. Insofern kann der Parameter mm vernachlässigt werden.

4.2.2 Mehr Aktualität – Boost Function bf auf publishDateSort

Durch den Einsatz einer Boost Function **bf** = recip(rord(publishDateSort), m, a, b) kann die Aktualität der Medien mit einem Boost versehen werden.

Diese Boost Function addiert zum eigentlichen Score-Wert einen Wert, der sich umgekehrt proportional zu *publishDateSort* verhält, wobei für m, a und b natürliche Zahlen eingesetzt werden müssen, um das Gewicht der Boost Function zu bestimmen.

$$bf = \frac{a}{m\left(rord(publishDateSort)\right) + b}$$

Die Funktion rord(publishDateSort) bewirkt, dass der Wert aus PublishDateSort von einem fiktiven Höchstwert abgezogen wird.

Die Höhe der Werte m, a, b wurde in Anlehnung des Beispiels in Graininger (2014)[70] versuchsweise mit 1, 100, 100 angesetzt. Um die Auswirkungen verschiedener Werte testen zu können, sollte es in einer nächsten Version der Webapplikation RankingTest möglich sein, die Werte m, a und b zu variieren.

Beispiel: Die Suche nach dem genauen Titel „Gray's Atlas der Anatomie“ liefert fünf Treffer. Alle haben den gleichen Score-Wert 14,884034, sind also für die Anfrage von gleicher Bedeutung. Die Aktualität wird bisher nicht berücksichtigt. Verwendet man die obengenannte Boost Function **bf** = recip(rord(publishDateSort),1,100,100), ändert sich die Sortierung, da nun auf den ursprünglichen Score-Wert der Wert aus der Boost Function **bf** aufgeschlagen wird.

[70] Grainger und Potter, Kap. 15.1.2.

ohne Berücksichtigung der Aktualität	**unter Einsatz von recip(rord(publishDateSort),1,100,100)**

{
"title":"Gray's Atlas der Anatomie",
"publishDateSort":"**2017**",
"score":14.884034},
{
"title":"Gray's Atlas der Anatomie",
"publishDateSort":"**2013**",
"score":14.884034},
{
"title":"Gray's Atlas der Anatomie",
"publishDateSort":"**2017**",
"score":14.884034},
{
"title":"Gray's Atlas der Anatomie",
"publishDateSort":"**2009**",
"score":14.884034},
{
"title":"Gray's Atlas der Anatomie",
"publishDateSort":"**2021**",
"score":14.884034}]
},
	{
	"title": "Gray's Atlas der Anatomie",
	"publishDateSort": "**2021**",
	"score": 14.958816},
	{
	"title": "Gray's Atlas der Anatomie",
	"publishDateSort": "**2017**",
	"score": 14.958281},
	{
	"title": "Gray's Atlas der Anatomie",
	"publishDateSort": "**2017**",
	"score": 14.958281},
	{
	"title": "Gray's Atlas der Anatomie",
	"publishDateSort": "**2013**",
	"score": 14.957822},
	{
	"title": "Gray's Atlas der Anatomie",
	"publishDateSort": "**2009**",
	"score": 14.957424
	}

Tabelle 20: Beispielabfrage nach <<“Gray's Atlas der Anatomie“>> ohne und mit Boost Function

Das Feld *publishDateSort* ist vom Feldtyp string. Durch die Prozessierungsvorschriften wird es in den verwendeten Solr-Cores grundsätzlich mit ganzen Zahlen belegt. Allerdings ist das Feld nicht zu 100% belegt (vgl. Anhang 10). Deshalb sollte die Funktion um die Prüfung if(exists(publishDateSort)…) erweitert werden, die den Boost-Wert auf „0“ setzt, wenn das Feld nicht belegt ist.

Um aktuellen Ausgaben mehr Gewicht zu verleihen, könnte diese Boost Function eingefügt werden.

```
bf =
if(exists(publishDateSort),recip(rord(publishDateSort),1,100,100),0)
```

4.2.3 Boost Queries bq

Nach den aktuellen Konfigurationen werden physische Bestände (access_facet = "Local Holdings") den Online-Ressourcen (access_facet = "Electronic Resources") vorgezogen, um die Nachteile der unterschiedlichen Feldlängen auszugleichen (vgl. Kapitel 3.3.3).

```
bq = access_facet:"Electronic Resources"^0 OR access_facet:"Local
Holdings"^50000.0
```

Aus verschiedenen Aussagen in den Dokumentationsseiten des Teams „Katalog“ der SLUB geht andererseits hervor, dass inzwischen eher Onlineangebote auf den vorderen Plätzen der Trefferliste erwartet werden. Durch die aktuell gesetzten Werte ist das kaum möglich.

Die bisherigen Einstellungen sind eine Behelfslösung, um Dokumente mit geringen Metadaten in der gerankten Trefferliste nach unten zu schieben. Ausführlich beschriebene Dokumente aus Bibliothekskatalogen, allen voran Daten aus dem K10plus, sollten dagegen im Ranking angehoben werden.

Datenquellen und deren Unterkollektionen können über die Felder *source_id* bzw. *mega_collection* identifiziert werden. In der Boost Query **bq** könnten Kollektionen, deren Daten als besonders wichtig erachtet werden, über die Felder *source_id* oder *mega_collection* mit positiven BoostFaktoren versehen werden. Möglich wäre auch ein negatives Boosting von Kollektionen, deren Metadaten als besonders gering eingeschätzt werden.

Variante 1: Dokumente aus dem Bibliothekskatalogisierungssystem K10plus (source_id=0) mit einem positiven Boost versehen:

```
bq = source_id:"0"^posBoost
```

Variante 2: Dokumente aus der Kollektion Crossref (source_id=49) und JSTOR (source_id=55) mit einem negativen Boost versehen:

```
bq = source_id:"49"^negBoost OR source_id:"55"^negBoost
```

Auch die Benachteiligung einzelner Medientypen sollte überprüft und dokumentiert werden, da zur Eingrenzung der Medientypen eine Facette zur Verfügung steht. Es gibt derzeit nur ca. 500 Dokumente mit der Belegung „Nachlass“ im Feld *format_de14*.

```
bq = -format_de14:"Nachlass"^999
```

Vereinzelt treten Forderungen nach einer Rangfolge von Medientypen auf. Diese Forderungen sind als eher subjektiv zu beurteilen. Da Medientypen über eine Facette wählbar sind, sollte diese Forderung unberücksichtigt bleiben.

4.2.4 PhraseFields pf

In Kapitel 2.4.2.4 wird der Parameter **pf** (PhraseFields) beschrieben, durch den Treffer mehr Gewicht erhalten, die die Suchterme in der Reihenfolge der Eingabe, d.h. als Phrase, beinhalten. Durch die Verwendung des Felds *title* mit einem positiven Boost im Parameter **pf** rücken Medieneinheiten, deren Titel die Benutzeranfrage als Phrase enthält, in der Trefferliste nach oben.

In Tabelle 21 ist die Beispielsuche nach <<Barockstadt Dresden>> abgebildet. Für das Ergebnis in der linken Spalte wurde die aktuelle Suchkonfiguration der Standardsuche „Alles“ verwendet. In der rechten Spalte wurde dieser Konfiguration der Parameter **pf = title^3000** hinzugefügt. Es ist deutlich zu sehen, dass in der rechten Spalte die Titel im oberen Bereich stehen, die „Barockstadt Dresden“ als Phrase enthalten.

Da in der aktuellen Suchkonfiguration insgesamt sehr hohe Boost-Werte eingesetzt werden (vgl. Kapitel 3.2), wurde für das Beispiel der Boost-Wert mit 3000 ebenfalls hoch gewählt, um eine Veränderung im Ranking zu bewirken.

aktuelle Suchkonfiguration des Standardsuche Alles	**mit zusätzlichem Parameter pf = title^3000**
Canaletto und die Barockstadt Dresden	Canaletto und die Barockstadt Dresden
Moderne in Dresden: Spurensuche in einer Barockstadt	Barockstadt Dresden und das Elbtal
Dresden: Stolze Barockstadt an der Elbe	Barockstadt Dresden und das Elbtal
Barockstadt Dresden und das Elbtal	Barockstadt Dresden und das Elbtal
Bahn-Renaissance für die Barockstadt: Dresdens Eisenbahnknoten erhält eine grundlegende Verjüngungskur	Barockstadt Dresden und das Elbtal
Stadtplan Barockstadt Dresden 1755: mit Straßen- und Gebäudeverzeichnis sowie einem etwas früheren Panoramabild und historischem Begleittext = Map of the baroque city of Dresden in 1755	Bahn-Renaissance für die Barockstadt: Dresdens Eisenbahnknoten erhält eine grundlegende Verjüngungskur
Barockstadt Dresden und das Elbtal	Stadtplan Barockstadt Dresden 1755: mit Straßen- und Gebäudeverzeichnis sowie einem etwas früheren Panoramabild und historischem Begleittext = Map of the baroque city of Dresden in 1755
Barockstadt Dresden und das Elbtal	Moderne in Dresden: Spurensuche in einer Barockstadt
Barockstadt Dresden und das Elbtal	Dresden: Stolze Barockstadt an der Elbe
Dresden-Archiv	Dresden-Archiv

Tabelle 21: Beispielabfrage nach Barockstadt Dresden, Treffer 1-10 ohne und mit Boost auf Phrase in Feld title

```
pf = title^posBoost
```

4.3 Felder für originalQueryString und isilQueryString

In diesem Kapitel werden die Vorschläge für Veränderungen im originalQueryString und im isilQueryString erläutert. Sie sind in Tabelle 23 und Tabelle 24 zusammengefasst.

4.3.1 author_corporate2

Das Feld *author_corporate2* ist in den verwendeten Indizes nicht mehr belegt. Es wurde aus den Prozessierungsregeln entfernt und kann deshalb auch aus der Suche entfernt werden.

4.3.2 topic_ref und topic

Das Feld *topic_ref* (text), welches derzeit im originalQueryString verwendet wird, ist nur in ca. 3% der Datensätze befüllt. Alle Datensätze, bei welchen *topic_ref* belegt ist, stammen aus dem Index finc-main. Das bedeutet, dass bei der Standardsuche Schlagwörter nur zu einem geringen Teil einbezogen werden. Eine wesentlich bessere Belegung weist das Feld *topic* (text) mit durchschnittlich 66,86% auf. Das Feld *topic_ref* sollte durch das Feld *topic* ergänzt werden.

4.3.3 geographic und fulltext

Bisher werden die Felder *isbn, issn, ismn, rsn_id_str_mv, barcode, signatur, record_id, zdb, swb_id_str, kxp_id_str, finc_id_str, geographic, collection* und *fulltext* nur in den originalQueryString aufgenommen, wenn die Benutzeranfrage mit mindestens zwei Ziffern beginnt. Die Anweisung ist folgerichtig für die Felder *swb_id_str, kxp_id_str* und *finc_id_str*, welche ausschließlich eine einzelne Zahl enthalten können.

Die Felder *geographic* und *fulltext* sollten ständig im originalQueryString enthalten sein, da sie mehrheitlich Wörter enthalten. Ggf. muss ihr Gewicht durch einen negativen BoostFaktor abgeschwächt werden.

4.3.4 rsn_id_str_mv und barcode

Die Felder *rsn_id_str_mv* (code) und *barcode* (String) beinhalten Werte aus dem Lokalsystem. Zur Identifizierung wird bei der Prozessierung als Präfix das Sigel der besitzenden Bibliothek in Klammern vor die RSN (*rsn_id_str_mv*) bzw. den Barcode (*barcode*) gesetzt. Die Werte der Felder *rsn_id_str_mv* und *barcode* der für die SLUB-angesigelten Datensätzen enthalten im Indexeintrag deshalb immer das Präfix „(DE-14)“. Auf Grund der verwendeten Feldtypen können Inhalte nur gefunden werden, wenn das Präfix dem gesuchten Term vorangestellt wird. Dadurch entsteht der Suchterm <<(DE-14)Term>>. Diese Variante der Suche wird bereits im isilQueryString durchgeführt und kann nur dort zum Ziel führen. Die Felder *rsn_id_str_mv* und *barcode* können deshalb aus dem originalQueryString entfernt werden.

4.3.5 signatur

Die Signaturen enthalten häufig neben alphanumerischen Zeichen auch Satz- und Leerzeichen. Da das Feld *signatur* vom Feldtyp code ist, werden die eingehenden Werte als ein einziges Token indexiert. Die Satz- und Leerzeichen bleiben erhalten. Alle Buchstaben werden in Kleinbuchstaben umgewandelt (vgl. Kapitel 3.1.4).

Signaturen aus dem Bestand der SLUB erhalten bei der Prozessierung der K10plus-Daten das Präfix „(DE-14)“. Um Signaturen der SLUB zu finden, muss deshalb auch dem gesuchten Term für das Feld *signatur* das Präfix „(DE-14)“ vorangestellt werden. In der Definition des isilQueryString ist das bereits umgesetzt.

Es muss zusätzlich beachtet werden, dass es auch für andere Datenquellen Belegungen im Feld *signatur* gibt, allerdings ohne das Präfix „(DE-14)“. Diese Signaturen können aktuell im SLUB-Katalog nicht gefunden werden. Durch die Erweiterung des isilQueryStrings um die Abfrage zum Feld *signatur* ohne Präfix kann diese Möglichkeit geschaffen werden.

Für den originalQueryString wird die Benutzeranfrage an Satz- und Leerzeichen getrennt. Aus typischen Signaturen entstehen mehrere Suchterme. Abschnitte von Signaturen können im Feld Signatur nicht gefunden werden. Aus dem originalQueryString kann das Feld *signatur* entfernt werden.

4.3.6 record_id, collection

Die Felder *record_id* und *collection* enthalten teilweise Satz- und Leerzeichen. Sie sind vom Feldtyp String (vgl. Kapitel 3.1.1) und werden als ein einiges Token inklusive der Satz- und Leerzeichen indexiert. Bei der Umcodierung der Benutzeranfrage für den originalQueryString werden alle Satzzeichen durch Leerzeichen ersetzt. Deshalb sind die Chancen gering, damit hier Treffer zu finden. Diese Felder sollten (auch) in den isilQueryString integriert werden.

4.3.7 zdb

Es besteht der Wunsch, Zeitschriften durch die Eingabe der ZDB-ID in der Standardsuche auffindbar zu machen. Das Feld *zdb* ist vom Feldtyp string. ZDB-IDs bestehen grundsätzlich aus einer Ziffernfolge an welche durch Bindestrich eine Prüfziffer angeschlossen ist. Die Suche über den originalQueryString wird immer erfolglos sein, da der Bindestrich bei der Übergabe der Sucheingabe entfernt wird. Das Feld *zdb* muss über den isilQueryString abgefragt werden.

4.3.8 isbn, ismn, isbn_isn_mv und issn_isn_mv

Bei Feldern des Feldtyps isn werden für die Indexierung und die Analyse des Suchterms alle Zeichen außer Ziffern und „x“ entfernt. Das gilt auch für eventuell zur Trunkierung eingesetzte Zeichen wie „?“ oder „*“. Ein Treffer wird nur erzielt, wenn Suchterm und der indexierte Term komplett übereinstimmen. Erfolgt die Abfrage beispielsweise einer ISBN mit Bindestrichen über die Standardsuche, werden die Bindestriche für den originalQueryString durch Leerzeichen ersetzt. Dadurch werden vier oder fünf Teilstücke der ISBN als Suchterme an die Felder des originalQueryStrings übergeben. Auf das Feld ISBN werden diese Anfragen erfolglos sein. Die Felder *isbn* und *ismn* sollten zusätzlich zu dem bereits enthaltenen Feld *issn* in den isilQueryString aufgenommen werden. Aus dem originalQueryString können sie entfernt werden.

Im Dezember 2021 wurden zwei weitere dynamische Felder für die Indexierung von ISBN und ISSN in die Prozessierung der MARC-Daten aufgenommen. Die Felder *isbn_isn_mv* und *issn_isn_mv* verzeichnen nicht die ISBN oder ISSN der vorliegenden Medieneinheit, sondern weitere ISBN / ISSN, die im Original-Datensatz verzeichnet sind. Dazu gehören u.a. gelöschte und falsche ISBN / ISSN sowie im Original-Datensatz verzeichnete ISBN / ISSN von Ausgaben, welche zur vorliegenden Medieneinheit in Beziehung stehen. Die Felder *isbn_isn_mv* und *issn_isn_mv* sollten in den isilQueryString eingefügt werden. Da die Treffer mit der exakten ISBN / ISSN zuerst angezeigt werden sollen, müssen die Felder *isbn* und *ismn* geboostet werden. Ob der BoostFaktor für das Feld *issn* weiterhin mit 500.000 angegeben wird, sollte in einer erweiterten Version der Webapplikation RankingTest getestet werden.

```
queryModifier.isilQueryString = (_query_:"{!edismax qf=\"barcode
rsn_id_str_mv\"}\\(DE-14\\)%1$s*") OR (_query_:"{!edismax
qf=\"signatur\"}\\(DE-14\\)%1$s")^10000 OR (_query_:"{!edismax
qf=\"signatur\"}%1$s") OR (_query_:"{!edismax
qf=\"callnumber_de14\"}%1$s")^10000 OR (_query_:"{!edismax
qf=\"marc024a_ct_mv\"}%1$s") OR (_query_:"{!edismax
qf=\"marc028a_ct_mv\"}%1$s") OR (_query_:"{!edismax
qf=\"marc535_cns_mv\"}%1$s") OR (_query_:"{!edismax qf=\"isbn
issn^5000000 ismn zdb isbn_isn_mv issn_isn_mv\"}%1$s") OR
(_query_:"{!edismax qf=\"collection record_id\"}%1$s")
```

Abbildung 34: Vorschlag für den neuen isilQueryString

In Tabelle 23 und Tabelle 24 werden die Vorschläge zur Änderung der Konfiguration des originalQueryStrings und des isilQueryStrings zusammengefasst. Sie enthält neben der Spalte mit der Bezeichnung der Solrfelder zur schnellen Orientierung eine Spalte mit einer kurzen Beschreibung. Die Spalte „enthalten in“ ist in drei Teile gegliedert, den isilQueryString und zwei Varianten des originalQueryStrings:

A – Grundform (Benutzeranfrage beginnt mit einem Wort (vgl. Kap. 3.3.4.2))

B – zusätzlich, wenn Benutzeranfrage mit mindestens zwei Ziffern beginnt (vgl. Kap.3.3.4.3)

Die Bedeutung der verwendeten Kennzeichnung ist in Tabelle 22 erläutert.

X	bleibt im jeweiligen QueryString enthalten
+(DE-14)	dem Suchstring muss das Präfix (DE-14) vorangestellt werden
(**X**)	kann aus dem originalQueryString entfernt werden
(—)	aus dem originalQueryString entfernen, da nur im isilQueryString sinnvoll
+	zusätzlich in den isilQueryString bzw. Variante des originalQueryStrings aufnehmen

Tabelle 22: Erläuterung der Kennzeichnungen in Tabelle 23 und Tabelle 24

Solrfeld	Beschreibung	enthalten in		
		originalQueryString		isilQuery String
		A	B	
author	Hauptautoren, die wichtigsten Autoren	**X**		
author_corporate	Körperschaft als Autor, auch weitere Körperschaften	**X**		
author_corporate2	weitere Körperschaften mit Nebeneintragungen	entfällt, da unbelegt		
author_ref	Varianten eines Namens aus den Normdaten	**X**		
author2	Nebenautoren bzw. Autoren mit Nebeneintragungen	**X**		
barcode	Barcodes aus Lokalsystem mit Sigel-Präfix	(—)		**X** **+(DE-14)**
callnumber_de14	Signaturen SLUB Dresden, Grundsignatur aus Lokalsystem	(**X**)		**X**
collection	Kürzel oder Name der Sammlung, die diesen Original-Datensatz enthält	(**X**)		**+**
contents	Details, Inhaltsverzeichnis	**X**		
description	Beschreibende Zusammenfassung des Objekts	**X**		
finc_id_str	IDs aus alten Versionen, u.a. für Link-Resolver alter Permalinks		**X**	
footnote	Fußnoten und sonstige Anmerkungen	**X**		
fulltext	Volltext	**+**		
geographic	geografische Schlagwörter	**+**		
isbn	Internationale Standardbuchnummer, nur die gültige ISBN	(**X**)		**+**
isbn_isn_mv	gelöschte/falsche ISBN, im Original-Datensatz zusätzlich verzeichnete ISBN			**+**
ismn	Internationale Standardmusiknummer	(**X**)		**+**
issn	Internationale Standardnummer für fortlaufende Sammelwerke, nur gültige ISSN	(**X**)		**X**
issn_isn_mv	gelöschte/falsche ISSN, im Original-Datensatz zusätzlich verzeichnete ISSN			**+**

Tabelle 23: Vorschläge für zu verwendende Felder im originalQueryString und isilQueryString, Teil1

Solrfeld	Beschreibung	enthalten in		
		originalQueryString		isilQuery String
		A	B	
kxp_id_str	KXP-PPN, ID im K10plus		X	
marc024a_ct_mv	Produktcodes, Standardnummern, DOIs, VD16,-17,-18-Angaben	X		X
marc028a_ct_mv	Verlags-, Produktions- und Bestellnummer	X		X
marc535_cns_mv	Signatur des Originals der Reproduktion	X		X
music_heading	Besetzung, Form und Gattung von Musikalien	X		
publishDateSort	Wert aus publishDate (Erscheinungsjahr), nur der erste dort angegebene Wert	X		
publisher	Verlag	X		
publishPlace	Erscheinungsort	X		
record_id	Lieferanten-Identifier (original ID aus der Quelle)			+
rsn_id_str_mv	RSN, ID im Lokalsystem	(—)		X +(DE-14)
series2	Titel der Serie(n),	X		
signatur	Signatur aus Lokalsystem	(—)		X +(DE-14)
signatur	Signatur aus Fremdsystemen	(—)		+
swb_id_str	SWB-PPN, ehemalige ID im SWB		X	
title_alt	alternative Titel (jede Form davon inkl. Paralleltitel)	X		
title_full_unstemmed	vollständige Titelaussage inklusive Autorenangabe/ Verfasserwiederholung usw.	X		
title_short	Kurztitel	X		
topic	Schlagwörter	+		
topic_ref	Schlagwörter in finc-Quellen werden über UBL-Anreicherung in MARC 950a ergänzt	(X)		
zdb	ID in der ZDB	(—)		+

Tabelle 24: Vorschläge für zu verwendende Felder im originalQueryString und isilQueryString, Teil II

5 Möglichkeiten zur Verbesserung der spezialisierten Sucheinstiege

Die aktuellen Einstellungen der spezialisierten Sucheinstiege werden in Kapitel 3.3.5 erläutert. In den folgenden Abschnitten werden nur die Sucheinstiege besprochen, zu welchen Änderungsvorschläge gemacht werden.

5.1 Titelsuche

Bei einer Titelsuche wird erwartet, auf eine Anfrage Treffer zu erhalten, welche die Terme der Sucheingabe in einem Titel (z.B. Hauptsachtitel, Zusatz zum Sachtitel, Paralleltitel, Einheitssachtitel) enthalten. Es ist davon auszugehen, dass Treffer eine höhere Bedeutung für die Benutzeranfrage haben, wenn die Terme der Benutzeranfrage im Hauptsachtitel enthalten oder/und die Termreihenfolge der Benutzeranfrage derjenigen des Titels gleicht.

Aktuell werden die Felder *title_full_unstemmed* (textProper) und *title_alt* (text), welches zusätzliche Titel und Titelinformationen enthält, für die Suche verwendet. Im Feld *title_full_unstemmed* ist der vollständige Titel inklusive der Verantwortlichkeitsangabe enthalten. Die Verantwortlichkeitsangabe enthält u.a. Personen und Institutionen, die für den Inhalt der betreffenden Medieneinheit verantwortlich sind. Diese Angaben sind nicht Teil des Titels. Die Verwendung des Felds *title_full_unstemmed* für den Sucheinstieg Titel kann zu Problemen führen, wenn Suchbegriffe der Benutzereingabe in der Verantwortlichkeitsangabe enthalten sind.

Beispiel: Gesucht wird ein Stadtplan der Stadt Seelow in Brandenburg. Der Benutzer wählt den Sucheinstieg „Titel“ und gibt <<Seelow>> in den Suchschlitz ein. Durch die Anfrage auf das Feld *title_full_unstemmed* mit dem Term „Seelow“ befindet sich innerhalb der ersten 10 Treffer auch das Buch „Halldór-Laxness-Werkausgabe“, da in der Verantwortlichkeitsangabe der Herausgeber Hubert Seelow genannt ist. Dieser Treffer ist für die obengenannte Benutzeranfrage nicht relevant.

	Halldór-Laxness-Werkausgabe hrsg. von Hubert Seelow						
ICUTokenizer	Halldór	Laxness	Werkausgabe	hrsg	von	Hubert	Seelow
WordDelimiterFilter	Halldór	Laxness	Werkausgabe	hrsg	von	Hubert	Seelow
ICUFoldingFilter	halldor	laxness	werkausgabe	hrsg	von	hubert	seelow
RemoveDuplicatesTokenFilter	halldor	laxness	werkausgabe	hrsg	von	hubert	seelow

Tabelle 25 Beispielindexierungen von Feld title_full_unstemmed (Feldtyp textProper)

Deshalb soll das Feld *title_full_unstemmed* nicht mehr als Suchfeld im Sucheinstieg Titel verwendet werden.

Eine Lösungsmöglichkeit besteht in der Einführung eines neuen CopyFields *title_unstemmed* vom Feldtyp textProper. Die Einführung neuer Felder wird in Abstimmung mit der UBL getroffen. Nach der Einführung müssten alle drei Solr-Cores neu indexiert werden.

Eine weitere Möglichkeit wäre, statt des Feldes *title_full_unstemmed* die Felder *title* und *title_short* zu verwenden und zusätzlich PhraseFields (vgl. Kap. 4.2.4) zu definieren.

Die Titelsuche sollte um die Felder *title_new* und *title_old* ergänzt werden, damit auch Vorgänger- und Nachfolgertitel gefunden werden. Den Feldern *title* und *title_short* sollte durch BoostFaktoren jeweils mehr Gewicht gegeben werden.

Hier sollte auch über den Einsatz der in Kapitel 4.2.3 beschriebenen Boost Function **bf** nachgedacht werden, um bei gleichem Titel eine Sortierung absteigend nach Erscheinungsjahr zu erreichen.

```
query = {!edismax qf=\"title^Boost title_short^Boost title_alt
title_new und title_old\" pf=\"title^Boost\"}%s
```

Abbildung 35: Vorschlag für eine neue Titelsuche

5.2 Personen/Institutionen

Wie in Kapitel 4.3.1 beschrieben, werden die Felder *author_corporate2* und *author_corporate2_orig* nicht mehr belegt. Sie sollten aus dem Parameter **qf** entfernt

werden. Hinzugefügt werden könnte die Abfrage nach *author_additional*. In diesem Feld werden die Verantwortlichkeitsangaben beigefügter Werke erfasst.

```
_query_:"{!edismax qf=\"author^300 author_id^100
author_corporate^300 author2^300 author_fuller^150 author_ref^500
author_corporate_ref^500 author_orig^300 author2_orig^300
author_corporate_orig^300 author_additional\"}%s"
```

Abbildung 36: Vorschlag für Änderungen Suche Personen/Institutionen

Die Boost-Werte der aktuellen Konfiguration wurden vorerst übernommen.

5.3 ISBN/ISSN/ISMN

In Kapitel 4.3.8 wurde beschrieben, dass im Dezember 2021 die Felder *isbn_isn_mv* und *issn_isn_mv* neu eingeführt wurden, da der Wunsch bestand, auch über die Angabe einer ISBN / ISSN einer anderen Ausgabe bzw. fehlerhaft verzeichneten Standardnummern zu Treffern zu gelangen. Die beiden Felder sollten zusätzlich in den Parameter **qf** aufgenommen werden. Durch die Vergabe eines BoostFaktors an die Felder *isbn, ismn* und *ismn* gelangen Datensätze mit korrekten Standardnummern im Ranking nach oben.

```
query = {!edismax qf=\"isbn^Boost issn^Boost ismn^Boost isbn_isn_mv
und issn_isn_mv\"}%s
```

Abbildung 37: Vorschlag für Änderungen Suche ISBN/ISSN/ISMN

5.4 Signatur

Für die Signaturensuche gibt es die Anforderung, Signaturen auch auffinden zu können, wenn im Suchfeld zusätzliche Leerzeichen zwischen den Signaturbestandteilen eingegeben werden.

Signaturen bestehen meistens aus alphanumerischen Zeichen, Satzzeichen und Leerzeichen. Die aktuell für den Sucheinstieg Signatur verwendeten Felder *signatur* und *callnumber_de14* sind vom Feldtyp code. Deren gesamter Inhalt wird durch den KeywordTokenizer als ein einziges Token übernommen. Anschließend werden alle Buchstaben durch den LowerCaseFilter in Kleinbuchstaben gewandelt (vgl. Kapitel 3.1.4). Die Verarbeitung des Suchterms erfolgt ebenso. Das bedeutet, dass bei der

Eingabe auf die korrekte Verwendung von Leerzeichen geachtet werden muss, um Treffer zu erzielen.

Abhilfe könnten hier z.B. CopyFields der Felder *signatur* und *callnumber_de14* mit Feldtyp callnumberSearch schaffen. In Abbildung 38 wird gezeigt, dass bei der Verwendung des Feldtyps callnumberSearch (vgl. 3.1.7) ohne Rücksicht auf Leerzeichen gesucht werden kann.

	NZ 14720 U53 D7-1	**Hist.Sax.C.3,misc.2**	**Hist. Sax. C. 3, misc. 2**
PatternReplaceCharFilter	NZ14720U53D7-1	Hist.Sax.C.3,misc.2	Hist.Sax.C.3,misc.2
KeywordTokenizer	NZ14720U53D7-1	Hist.Sax.C.3,misc.2	Hist.Sax.C.3,misc.2
ICUFoldingFilter	nz14720u53d7-1	hist.sax.c.3,misc.2	hist.sax.c.3,misc.2

Abbildung 38: Verarbeitung von Beispielsignaturen durch eine callnumberSearch-Feld

Bis zur Einführung neuer Felder wird die Konfiguration für den Sucheinstieg Signatur aus Kapitel 3.3.5.7 beibehalten.

5.5 Neuer Sucheinstieg Zeitschriften

Um die Suche nach Zeitschriften nutzerfreundlicher zu gestalten, sollte ein neuer Sucheinstieg geschaffen werden, in welchem nur der Medientyp Zeitschrift angezeigt wird. Die Filter Queries sind deshalb wie folgt umzustellen:

```
    additionalFilters {
        1 = institution:DE-14
        2 = format_de14:"Journal E-Journal"
```

Abbildung 39: Vorschlag für additionalFilters für Sucheinstieg Zeitschriftensuche

Der Medientyp „Zeitschrift“, unabhängig davon, ob er als elektronisches oder physisches Medium vorliegt, weicht bei der Beurteilung der Aktualität von anderen Medien ab. Ausschlaggebend ist hier nicht das Erscheinungsjahr, sondern i.A. der nicht abgeschlossene Erscheinungszeitraum. Bei der Erfassung von Zeitschriften in Datenquellen wie dem K10plus oder der ZDB wird durch ein „-“ am Ende des Erfassungsfeldes angezeigt, dass die Zeitschrift weiterhin erscheint. Diese Kennzeichnung sagt aus, das aktuelle Bände vorliegen können. Der Erscheinungsverlauf

wird im Feld *dateSpan* (string) abgebildet. Einen Boost sollten deshalb Treffer erhalten, deren Wert im Feld dateSpan mit „-“ endet.

```
bq = dateSpan:"-"^Boost
```

Präzise Abfragen zu vorhandenen Zeitschriftenjahrgängen wären nur möglich, wenn auch die Bestandsnachweise von Zeitschriftenjahrgängen indexiert würden.

6 Webapplikation für Queryanpassungen

6.1 Anforderungen an den Prototypen der Webapplikation

Die Webapplikation soll das Suchverhalten im SLUB-Katalog möglichst genau nachbilden. Dafür muss sie in der Lage sein, bei Eingabe einer Benutzeranfrage eine URL zu bilden. Diese muss die gleichen konstanten Parameter und den gleichen QueryString enthalten, der bei der dieser Benutzeranfrage an den SLUB-Katalog gebildet worden wäre.

Zusätzlich soll sie die Möglichkeit bieten, Werte, die Einfluss auf das Ranking haben, zu verändern. Wichtig ist die Dokumentation der getesteten Konfigurationen. Deshalb soll die Webapplikation eine Möglichkeit zum Speichern der Beurteilungen und Erkenntnisse bieten.

6.1.1 Funktionale Anforderungen

Lfd. Nr.	**Beschreibung der Anforderung**
FA_01	Es muss ein Feld zur Eingabe der Benutzeranfrage geben.
FA_02	Es sollen Felder zu Eingabe von BoostFaktoren für alle in der Standardsuche „Alles“ verwendeten Indexfelder vorhanden sein.
FA_03	Es soll ein Feld zur Eingabe des Minimum Should Match mm vorhanden sein
FA_04	Es soll ein Feld zur Eingabe der BoostFaktoren innerhalb der verwendeten Boost Function bf vorhanden sein.
FA_05	Die Angaben von Benutzeranfrage, BoostFaktoren und des Parameters Minimum Should Match sollen bei der Nutzung der Defaultwerte zur gleichen Anfragebearbeitung führen, die der SLUB-Katalog verwendet und eine Treffermenge liefern.
FA_06	Die Treffer sollen mit geeigneten Feldern angezeigt werden.
FA_07	Es soll sichtbar sein, in welchen Felder die Terme der Benutzeranfrage gefunden wurden.
FA_08	Die Anzahl der Treffer pro Seite soll begrenzt sein.
FA_09	Ergeben sich mehrere Trefferseiten, soll zwischen den Seiten navigiert werden können.

FA_10	Es soll möglich sein, die Treffer in Bezug auf ihre Relevanz zur Benutzeranfrage zu beurteilen
FA_11	Es soll möglich sein, die Beurteilung der Treffer in einer Datei inklusive der verwendeten Konfiguration und der Benutzeranfrage zu speichern.
FA_12	Es soll möglich sein, den Namen der zu speichernden Datei festzulegen.

Tabelle 26: Funktionale Anforderungen an die Webapplikation

6.1.2 Nicht-funktionale Anforderungen

Lfd. Nr.	**Beschreibung der Anforderung**
NFA1	Die Webapplikation soll auch durch Mitarbeiter der SLUB mit geringen Programmierkenntnissen bedient werden können.
NFA2	Die Webapplikation soll in allen gängigen Browsern funktionieren.

Tabelle 27: Nicht-Funktionale Anforderungen an die Webapplikation

6.2 Konzeption der Webapplikation

Auf Grund der aktuellen, auch durch die Corona-Pandemie bedingten, angespannten Personalsituation in der SLUB konnte der Prototyp der Webapplikation nicht wie geplant innerhalb der TYPO3-find-Umgebung entwickelt werden. Die Entscheidung fiel deshalb zugunsten einer Single-Page-Webapplikation mittels HTML5 und jQuery, die die Basisfunktionen der Webapplikation enthält. Diese dient zur Vorstellung der Idee bei ausgewählten Mitarbeitenden der SLUB und soll bei entsprechender Akzeptanz als Basis für eine Migration der Webapplikation in die TYPO3-find-Umgebung verwendet werden.

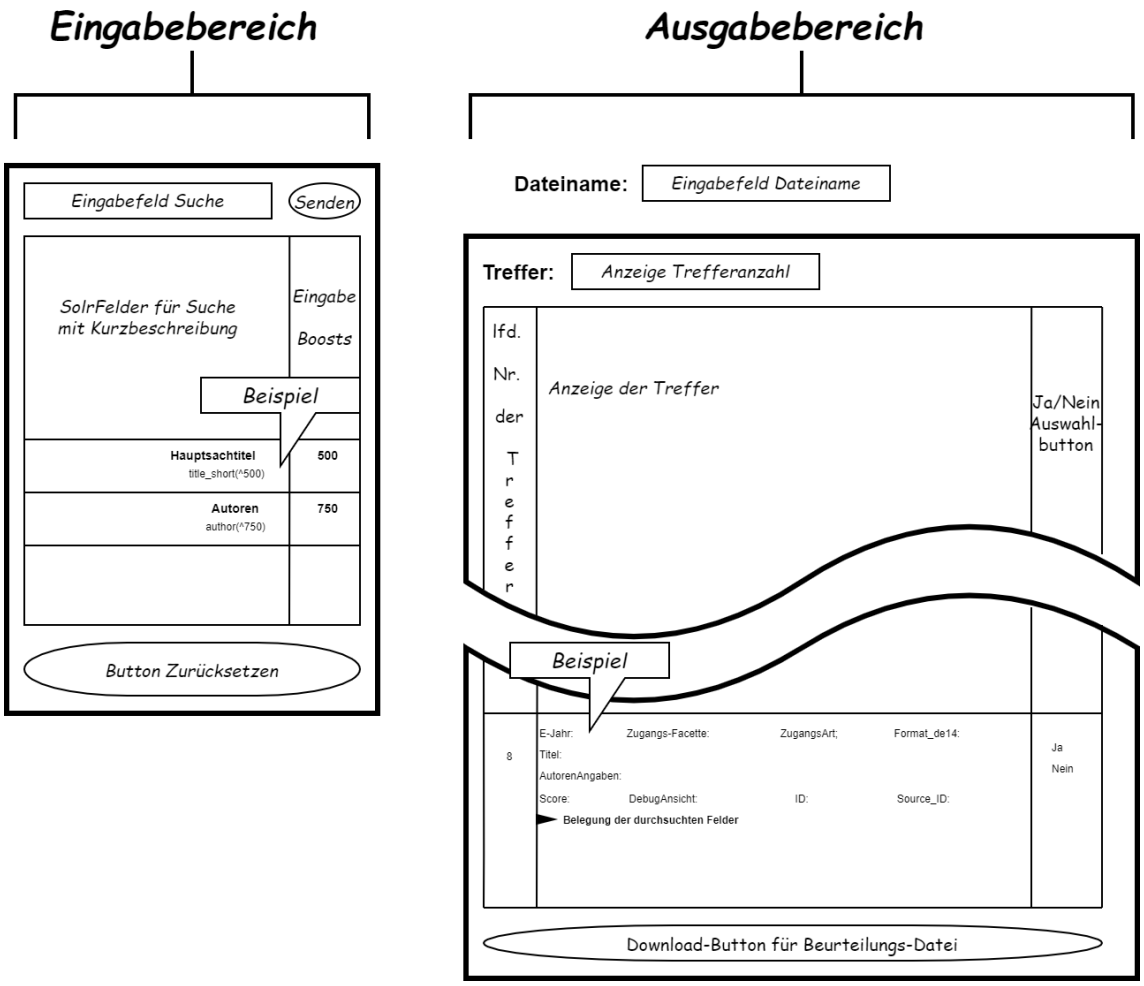


Abbildung 40: Skizze 1 Webapplikation RankingTest

Der **Eingabebereich** soll ein HTML-Formular zur Übergabe der Benutzeranfrage und der für die Anfrage zu verwendenden BoostFaktoren an die JavaScript-Logik enthalten. Die reinen Solrfeld-Bezeichnungen müssen mit einer Kurzbeschreibung versehen sein, um den Testpersonen die Bedeutung des Feldes zu erläutern. Die Werte der BoostFaktoren der aktuellen Konfiguration werden im HTML-Formular als Default hinterlegt. Sie werden gleichzeitig auch in der Feldbeschreibung angegeben, um ein manuelles Rücksetzen eines Wertes zu ermöglichen. Für den Prototyp der Webapplikation wurden zunächst neun Werte aus den aktuellen Konfigurationen des SLUB-Katalogs gewählt:

Solrfeld	Erläuterung
title_full_unstemmed	Hauptsachtitel, Zusatz zum Titel, Verfasserangabe
title_short	Hauptsachtitel
author	Autoren
author2	Nebenautoren bzw. Autoren mit Nebeneintragungen
author_corporate	Körperschaft als Autor, auch weitere Körperschaften
access_facet = "Electronic Resources"	Zugangsfacette, es handelt sich um eine elektronische Ressource
access_facet = "Local Holding"	Zugangsfacette, es handelt sich um eine lokal vorliegende Medieneinheit
-format_de14:"Nachlass"	Medientyp ist nicht „Nachlass“
mm	Minimum Should Match

Tabelle 28: Solrfelder und –Parameter zur Änderung der BoostFaktoren

Zum Auslösen der Benutzeranfrage wird ein Button „Senden“ benötigt. Ein Button „Zurücksetzen“ soll die Herstellung des Ausgangszustands aller Formularfelder ermöglichen.

Da aus dem Ausgabebereich die Speicherung der Bewertung erfolgen muss, wird dieser Bereich in einem weiteren HTML-Formular realisiert. Zuerst soll die Gesamtzahl der erzielten Treffer angezeigt werden. Es folgt die Anzeige einer begrenzten Anzahl Treffer. Diese sollten fortlaufend nummeriert sein und wichtige Angaben enthalten, die eine grobe Übersicht über Treffer geben, anhand derer seine Relevanz für die Anfrage beurteilt werden könnte. Folgende Werte werden für wichtig erachtet:

gewünschte Anzeige / Information	verwendetes Solrfeld
Erscheinungsjahr	publishDateSort
Zugangsfacette (Schieberegler des SLUB-Katalogs)	access_facet
weitere Information zum Zugang	facet_avail
Angabe über Format der Medieneinheit	format_de14
Hauptsachtitel, Zusatz zum Titel, Verfasserangabe	title_full
Erscheinungsvermerk (Ort, Verlag, Jahr)	imprint_str_mv
Personen (Autoren, Herausgeber, Beteiligte…)	author, author2
Körperschaften	author_corporate
Score-Wert	score
Link zur Anzeige des Treffers im SLUB-Katalog	id
Link zur Debug-Anzeige des Treffers im SLUB-Katalog	id
Identifier der Datenquelle	source_id
in welchen Feldern wurde(n) der/die Term(e) der Benutzeranfrage gefunden?	

Tabelle 29: Angaben, die die Anzeige der Treffer enthalten sollten

Zur Anzeige, in welchen Feldern der/die Term(e) der Benutzeranfrage gefunden wurden, wird zu jeder Trefferanzeige eine Liste der durchsuchten Felder angezeigt. Die Liste wird als erweiterbare Anzeige realisiert, um die Anzeige übersichtlicher zu halten.

Es werden alle für die Standardsuche verwendeten Felder in die Liste aufgenommen. Felder, die nicht angezeigt werden können, weil sie in der schema.xml mit dem Attribut stored =„false“ gekennzeichnet sind, werden mit einem Hinweis versehen. Folgende Felder und Hinweise werden aufgenommen:

- author
- author2
- author_ref (stored =„false“ – keine Anzeige möglich)
- author_corporate
- title_full_unstemmed
- title_short
- title_alt
- publisher
- publishPlace
- publishDateSort
- topic_ref (stored =„false“ – keine Anzeige möglich)

- series2
- contents
- description
- footnote
- marc024a_ct_mv
- marc028a_ct_mv
- marc535_cns_mv
- music_heading
- callnumber_de14
- issn

- ***Folgende Felder werden nur durchsucht, wenn die Benutzeranfrage mit Ziffernfolge beginnt***
  - collection
  - fulltext (stored =„false“ – keine Anzeige möglich)
  - geographic
  - signatur
  - barcode
  - isbn
  - ismn
  - issn
  - record_id
  - kxp_id_str
  - finc_id_str
  - rsn_id_str_mv
  - swb_id_str
  - zdb

Die Bewertungsmöglichkeit für jeden Treffer soll über Radiobuttons ermöglicht werden. Zur Übersichtlichkeit der Trefferanzeige wird aus dem Response eine dreispaltige Tabelle aufgebaut, die folgende Werte enthält:

Spalte	Inhalt
1	laufende Nummer
2	Angaben zum Treffer (vgl. Tabelle 29)
3	Radiobuttons Ja/Nein, kein Defaultwert

Tabelle 30: Tabelle über Inhalt der Trefferanzeige

Für die geplante Speicherung der Bewertungen wird ein Feld zur Eingabe eines Dateinamens und ein Button zum Absenden der Bewertung benötigt. Für den Dateinamen wird der Defaultwert „test“ vorgegeben.

Durch die Möglichkeit des Downloads einer festlegbaren Anzahl Treffer inklusive der Suchanfrage, verwendeter Parameter und Bemerkungen in einer CSV-Datei können

Beurteilungen in vergleichbarer Weise dokumentiert werden. Die Angaben können als Tabelle zur Dokumentation z.B. im Intranet verwendet werden.

6.3 Bewertung des Prototyps RankingTest

Im Intranet der SLUB wurde eine Nutzungsanleitung inklusive kurzer Beschreibung der Ziele der Webapplikation für einen begrenzten Personenbereich bereitgestellt (vgl. Anhang 1). Zwei Bibliothekarinnen des Teams „Katalog“ und einer Fachreferentin für Kunst konnten für erste Tests gewonnen werden. Die grundsätzliche Idee der Webapplikation erhielt ein positives Echo, zumal an der SLUB mehrere Suchoberflächen verwendet werden, die auf Solr-Cores zugreifen. Neben dem SLUB-Katalog sind das z.B. der Katalog des Fachinformationsdiensts Kunst, arthistoricum.net und das Regionalportal Saxorum. Für einen gewinnbringenden Einsatz wurden in der Auswertung deutlich erweiterte Funktionalitäten formuliert.

- Die Liste der Felder, für welche die Boost-Werte einstellbar sind, soll alle für die Suche benutzten Felder umfassen.
- Für einen effektiven Einsatz sollte die Applikation enger an das jeweilige Frontend angepasst werden. Mindestens ist die Angleichung der Bezeichnung der in der Webapplikation verwendeten Felder und der Bezeichnungen im Katalog-Frontend gewünscht.
- Die angezeigte Treffermenge sollte größer bzw. individuell anpassbar sein.
- Es soll eine Blätter- oder Navigationsfunktion innerhalb der Treffermenge geben.

Die Bewertungsmöglichkeit mit „ja/nein“ wurde unterschiedlich bewertet: Ein Textfeld für eine kurze Bewertung scheint besser geeignet. Andererseits wurde bereits die Möglichkeit des Downloads der Angaben der Treffer zusammen mit den Boost-Einstellungen als nützlich angesehen.

6.4 Anwendung der Webapplikation

Die Qualität eines Bibliothekskatalogs wird u.a. daran gemessen, ob er Medien, die inhaltlich zur Benutzeranfrage passen, in ausreichendem Maß und einer für den Anfragenden sinnvollen Sortierung anzeigt. Die Bildung der Treffermenge ist der erste Schritt zur Anzeige des Suchergebnisses. Sie wird u.a. durch die Auswahl der

durchsuchten Indexfelder und die Angaben, mit welcher Genauigkeit die Suchanfrage zu beantworten ist, beeinflusst. In Kapitel 4 und 5 wurden Vorschläge für die Neugestaltung der Standardsuche und einiger spezialisierter Sucheinstiege im SLUB-Katalog erarbeitet. Im nächsten Schritt müssen die Angaben der Boost-Werte für die Parameter wie Query Fields **qf**, Boost Function **bf** und Boost Query **bq** festgelegt werden.

Zur Unterstützung dieses iterativen Prozesses wurden die Vorschläge aus Kapitel 4 in einer erweiterten Version als RankingTest_Version2 umgesetzt. Die Anzahl der Felder, deren Boost-Werte verändert werden können, wurde stark erweitert (Abbildung 41). Die bisherigen Boost-Werte wurden in dieser Version als Default übernommen.

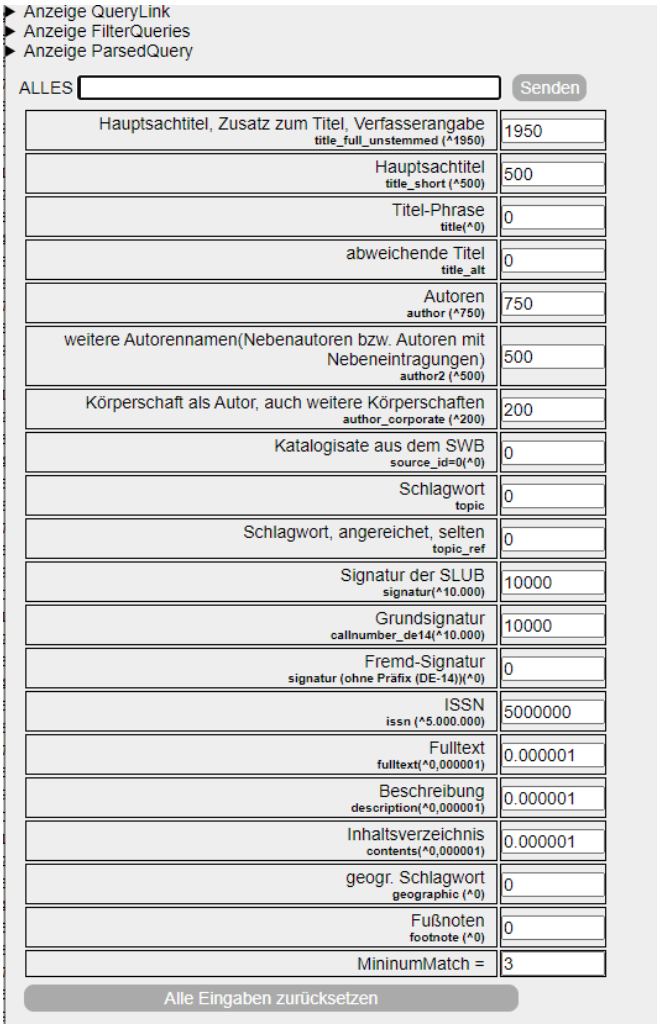


Abbildung 41: Eingabeformular Webapplikation RankingTest_Version2

Beurteilungen können in der RankingTest_Version2 als Text angegeben werden (vgl. Abbildung 42).

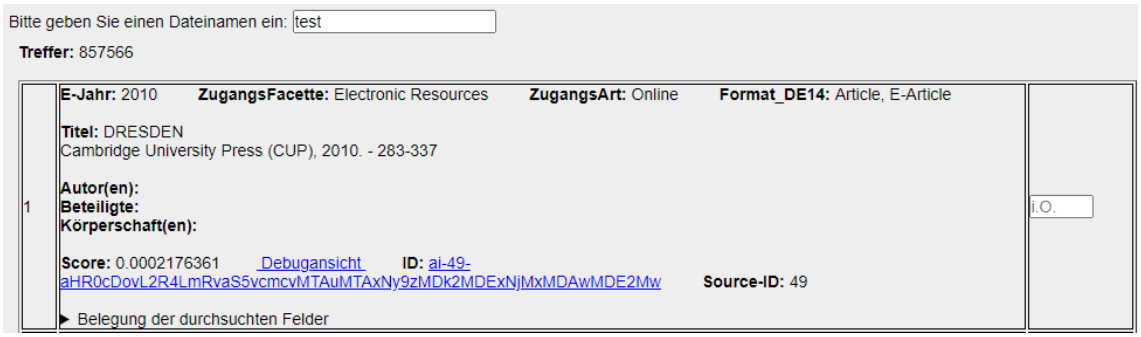


Abbildung 42: Trefferanzeige in RankingTest_Version2

7 Fazit

Ziel dieser Masterarbeit war die Entwicklung eines Konzepts zur Verbesserung des Relevanzrankings im SLUB-Katalog. Dieses Ziel wurde nur teilweise erreicht. Es konnten aber Grundlagen erarbeitet werden, auf welchen im Anschluss an die Masterarbeit gemeinsam mit dem Team „Katalog“ ein Testkonzept erarbeitet werden kann. Die Arbeit bildet die Diskussionsgrundlage für eine Überarbeitung der Suche im SLUB-Katalog.

Während der Bearbeitungszeit der Masterarbeit wurde erkannt, dass Kenntnisse über die Arbeitsweise von Discovery-Systemen, über Möglichkeiten der Parametrisierung der Suche auf Solr-Indizes und das finc-Solr-Schema bisher nur bei einem eng begrenzten Personenkreis in der SLUB vorhanden sind. Auch die aktuellen Sucheinstellungen sind den Mitarbeitenden eher unbekannt. Das betrifft auch das Team „Katalog“. Im Intranet-Bereich des Teams „Katalog“ befinden sich z.Z. verschiedene Seiten zur Beschreibung der Sucheinstellungen. Die Angaben sind teilweise widersprüchlich und veraltet. Die gültige Konfiguration kann nur in TYPO3 eingesehen werden. Die Zugriffsrechte dafür sind begrenzt.

Die Übertragung der Ergebnisse der Masterarbeit in das Intranet soll nun ein erster Schritt sein, um Mitarbeitenden ein besseres Verständnis des Relevanzrankings im Allgemeinen und im SLUB-Katalog im Besonderen zu vermitteln.

Zuerst wurden die aktuellen Suchkonfigurationen und die Eignung verfügbarer Felder des finc-Solr-Schemas als Suchfeld analysiert. Dazu wurden insbesondere die in der schema.xml festgelegten Feldtypen untersucht. Das Verständnis der unterschiedlichen Verarbeitung der Feldinhalte bei der Indexierung bzw. der Suchanfrageverarbeitung ist die Basis, um die Sucheinstellungen zu konfigurieren. Unter Hinzuziehung der Belegungshäufigkeit konnten bereits Vorschläge zu Änderungen in der Standardsuche „Alles“ und in vier spezialisierten Sucheinstiegen entwickelt werden.

Die Webapplikation RankingTest kann das Team „Katalog“ in doppelter Hinsicht unterstützen. Zum Testen mussten die Boost-Werte bisher in den TYPO3-find-Konfigurationen eines Test-Frontends durch den IT-Mitarbeiter mit TYPO3-find-

Kenntnissen geändert werden. Die Webapplikation RankingTest ermöglicht es SLUB-Mitarbeitenden verschiedene Boost-Konfigurationen zur Suchanfragezeit auszuprobieren. Da die Webapplikation in mehreren Browserfenstern geöffnet sein kann, können unterschiedliche Konfigurationen der Werte direkt verglichen werden. Gleichzeitig bietet sie die Möglichkeit einer einfachen, aber gleichmäßigen und nachhaltigen Dokumentation durch downloadbare CSV-Dateien, welche die verwendeten Werte sowie wichtige Angaben zu Benutzeranfrage und Treffern enthalten.

Entscheidend bei Änderungen von Boost-Werten und anderen Parametern ist ein systematisches Vorgehen, um zu verhindern, dass Änderungen zugunsten einer Anforderung sich negativ auf andere Anfragen auswirken. Deshalb müssen durch das Team „Katalog“ Testanfragen für alle Sucheinstiege definiert werden, die bei Änderungen der Suchkonfiguration regelmäßig mittels der Webapplikation getestet werden. Die Dokumentation wird durch die Webapplikation unterstützt.

Die Webapplikation bietet bisher keine Möglichkeit, die Bewertungen der Konfigurationen zu einem Gesamtergebnis zu erfassen.

Nach der Vorstellung der Ergebnisse der Masterarbeit und der Demonstration der Webapplikation im Team-Katalog wird die Überführung in die TYPO3-find-Umgebung angestrebt. Dadurch entsteht die Möglichkeit, die Parameter aller Sucheinstiege des SLUB-Katalogs zur Suchanfragezeit konfigurieren zu können.

Der Möglichkeit, das Relevanzranking mit überschaubarem technischen Aufwand durch die Änderung von Parametern zu beeinflussen, sollten sich Überlegungen im Zusammenklang der Abteilungen Benutzung, IT, Team „Datenmanagement“ und Team „Katalog“ anschließen, ob das Boosting auch für individuelle Sucheinstellungen genutzt werden kann.

Literaturverzeichnis

Bade, David (2007): Relevance ranking is not relevance ranking or, when the user is not the user, the search results are not search results. In: *Online Information Review* 31 (6), S. 831–844. Online verfügbar unter https://www.emerald.com/insight/content/doi/10.1108/14684520710841793/full/pdf, zuletzt geprüft am 21.07.2021

Baeza-Yates, Ricardo; Moffat, Alistair; Navarro, Gonzalo (2002): Searching Large Text Collections. In: James Abello, Panos M. Pardalos und Mauricio G. C. Resende (Hg.): Handbook of Massive Data Sets. Boston, MA: Springer US, S. 195–243

Behnert, Christiane; Borst, Timo (2015): Neue Formen der Relevanz-Sortierung in bibliothekarischen Informationssystemen: Das DFG-Projekt LibRank. In: *Bibliothek Forschung und Praxis* 39 (3), S. 384–393. DOI: 10.1515/bfp-2015-0052

Breeding, Marshall (2012): Library Web-Scale. In: *Computers in libraries* 32 (1), S. 19–21

Christensen, A.; Kessler, K.; Maas, J.; Schrader, J.; Seng, H.; Strötgen, R. (2018): Elektronische Schatzsuche. In: *Deutsche Universitätszeitung* 7 (7.9.2018), S. 22–27. Online verfügbar unter https://www.leuphana.de/fileadmin/user_upload/kooperationen/DUZ-Artikel/duz_WiMa_07_2018_Elektronische_Schatzsuche.pdf, zuletzt geprüft am 26.05.2021

Christensen, Anne (2014): Discovery-Systeme. Zwischenbilanz für eine bibliothekarische Lösung in einer post-bibliothekarischen Welt. Jena (Discovery-Summit der GeSIG an der ThULB Jena). Online verfügbar unter https://de.slideshare.net/xenzen/discoverysysteme-zwischenbilanz-fr-eine-bibliothekarische-lsung-in-einer-postbibliothekarischen-welt, zuletzt geprüft am 07.06.2021

Croft, W. Bruce; Metzler, Donald; Strohman, Trevor (2010): Search engines : information retrieval in practice. Boston, Mass.: Pearson. Online verfügbar unter https://ciir.cs.umass.edu/downloads/SEIRiP.pdf

Grainger, Trey; Potter, Timothy (2014): Solr in action. Shelter Island, NY: Manning

Klose, Markus (2014): Einführung in Apache Solr. Unter Mitarbeit von Daniel Wrigley. Beijing: O'Reilly

Lazarus, Jens (2016): Das machen wir selbst. Der Aufbau eines eigenen Artikelindex als Alternative zu proprietären Angeboten. Leipzig (105. Deutscher Bibliothekartag). Online verfügbar unter https://opus4.kobv.de/opus4-bib-info/files/2359/Lazarus+AI+BKongress+2016+final.pdf, zuletzt geprüft am 28.01.2022

Lewandowski, Dirk (2010): Der OPAC als Suchmaschine. In: Julia Bergmann und Patrick Danowski (Hg.): Handbuch Bibliothek 2.0: De Gruyter Saur, S. 87–107. Online verfügbar unter http://dx.doi.org/10.1515/9783110232103, zuletzt geprüft am 28.04.2021

Lewandowski, Dirk (2014): Die Macht der Suchmaschinen und ihr Einfluss auf unsere Entscheidungen. In: *Information. Wissenschaft & Praxis* 65 (4-5), S. 231–238. Online verfügbar unter https://www.degruyter.com/document/doi/10.1515/iwp-2014-0050/html, zuletzt geprüft am 12.01.2022

Manning, Christopher D.; Raghavan, Prabhakar; Schütze, Hinrich (2008): Introduction to information retrieval. Online verfügbar unter https://nlp.stanford.edu/IR-book/information-retrieval-book.html

Obenland, Oliver (2017): Konzeption und Implementation des Aufbaus eines Suchmaschinenindexes für Solr im bibliothekarischen Kontext. Masterarbeit. Eberhard Karls Universität, Tübingen. Wilhelm-Schickard-Institut für Informatik. Online verfügbar unter http://nbn-resolving.de/urn:nbn:de:bsz:21-dspace-783513, zuletzt geprüft am 02.06.2021

Robertson, Stephen (2004): Understanding Inverse Document Frequency: On Theoretical Arguments for IDF. In: *Journal of Documentation - J DOC* 60, S. 503–520. DOI: 10.1108/00220410410560582

Sächsische Landesbibliothek - Staats- und Universitätsbibliothek Dresden (2019): SLUB 2025 Wissen teilen - Menschen verbinden. Unter Mitarbeit von Bonte, Achim (Hrsg.) und Antonie [Hrsg.] Muschalek. Dresden: SLUB

Salton, Gerard; McGill, Michael J. (1983): Introduction to modern information retrieval. New York u.a.: McGraw-Hill (McGraw-Hill Computer Science Series)

Schuldt, Tino (2015): Aufbau und Vergleich webbasierter Suchmaschinen. Bachelorarbeit. Hochschule Neubrandenburg, Neubrandenburg. Studiengang Geoinformatik. Online verfügbar unter http://digibib.hs-nb.de/resolve?id=dbhsnb_thesis_0000001369, zuletzt geprüft am 29.11.2021

Stock, Wolfgang G. (2007): Information retrieval. Informationen suchen und finden. München: Oldenbourg (Einführung in die Informationswissenschaft, 1)

Tramullas, Jesus; Garrido, Piedad (2013): Library automation and OPAC 2.0. Information access and services in the 2.0 landscape. Hershey PA: Information Science Reference. Online verfügbar unter http://kxp.k10plus.de/DB=2.1/PPNSET?PPN=688501915

Wilhelm-Stein, Thomas (2016): Information Retrieval in der Lehre. Unterstützung des Erwerbs von Praxiswissen zu Information Retrieval Komponenten mittels realer Experimente und Spielemechaniken. Chemnitz: Universitätsverlag Chemnitz (Wissenschaftliche Schriftenreihe Dissertationen der Medieninformatik, Band 5). Online verfügbar unter http://nbn-resolving.de/urn:nbn:de:bsz:ch1-qucosa-199778, zuletzt geprüft am 26.06.2021

Abkürzungen und Begriffserläuterungen

ansigeln ..... Kennzeichnung in gemeinsamen Nachweissystemen, dass eine Bibliothek ein Exemplar des Titels besitzt bzw. einen bestimmten Datensatz verwenden möchte

Barcode ..... zur Identifizierung von Exemplaren z.B. bei der Ausleihverbuchung

Boost-Wert ..... auch Boost, Boost-Wert oder Boost-Faktor: Werte, die unabhängig von der vom Ranking-Algorithmus zur Berechnung verwendeten Formel Einfluss auf den berechneten Relevanzwert nehmen

BSZ ..... Bibliotheksservice-Zentrum Baden-Württemberg

CBS ..... Zentrales Bibliothekssystem, gemeinsame Datenbank von Bibliotheken, in der eine kooperative Katalogisierung stattfindet

CMS ..... Content Managemen System, Software zum Erstellen, Bearbeiten,und Darstellen digitaler Inhalte, z.B. Webseiten

Crossref ..... Non-Profit-Organisation, die unter Verwendung von Digital Object Identifiern (DOIs) die Verlinkung von Millionen von Artikeln aus Zeitschriften, Büchern, Konferenzberichten etc. aus verschiedensten Wissenschaftsbereichen ermöglicht

DOAJ ..... Directory of Open Access Journals, Verzeichnis elektronischer, frei im Internet zugänglicher Zeitschriften

DOI ..... Digital Object Identifier

DOM ..... Document Object Model

EFRE ..... Europäischer Fonds für regionale Entwicklung

ERM ..... Electronic Ressource Management

Exemplarsatz ..... vgl. Lokalsatz

finc-ai ..... Solr Index, der Artikelindex der finc-Nutzergemeinschaft, der vorrangig unselbständige Werke, d.h. Artikel aus Zeitschriften und Büchern verzeichnet

finc-main ..... Solr Index der finc-Nutzergemeinschaft, der vorrangig bibliographisch selbständige Medieneinheiten, z.B. Daten aus K10plus, verzeichnet

GBV ..... Gemeinsamen Bibliotheksverbund

GND ..... Gemeinsamen Normdatei: Datenbank zur Verzeichnung von Normdaten, die Personen, Körperschaften, Konferenzen, Geografika, Sachbegriffe und Werke eindeutig beschreiben

ISBN ..... International Standard Book Number

ISMN ..... International Standard Music Number

ISSN ..... International Standard Serial Number

JSTOR ..... Journal Storage: digitale Bibliothek, kostenpflichtiges Online-Archiv ausgewählter Fachzeitschriften und wissenschaftlicher Büchern hauptsächlich aus dem Bereich der Geistes- und Sozialwissenschaften

K10plus..... vom BSZ und der VZG gemeinsam betriebene Datenbank, entstand durch die Vereinigung der Datenbanken der beiden Verbundsysteme SWB und GBV

KXP ..... andere Abkürzung für K10plus

LBS..... Lokales Bibliothekssystem

Libero ..... ist das Lokalsystem (lokale Bibliothekssystem) der SLUB

Lokalsatz..... auch Lokaldatensatz, Exemplarsatz, Bestandsnachweis einer Bibliothek im CBS

Lokalsystem..... auch Lokales Bibliothekssystem, LBS: verzeichnet die lokalen Daten der Bibliothek zu ihren Beständen, wie Inventarisierungs-, Erwerbungs- und Ausleihdaten

MARC ..... Machine-Readable Cataloging: international verwendetes Datenformat zur Erfassung, Speicherung und Austausch bibliografischer Daten

MARCXML ..... XML-Kodierung für MARC21-Daten

Normdaten ..... normierte Begriffe aus kontrollierten Vokabularen, z.B. GND oder ORCID

OPAC ..... Online Public Access Catalogue, verzeichnet den lokalen Bestand einer Bibliothek

ORCID..... ID zur Vernetzung zwischen Publizierenden und ihren Aufsätzen und Forschungsdaten

PICA ..... bibliografisches Datenformat,
es gibt zwei Ausprägungen: Pica3 und PICA+

PPN..... Pica-Produktionsnummer, ID in Pica-Datenbanken

RSN ..... Record Serial Number, interner Identifier im Lokalsystem Libero

Sigel..... auch Bibliothekssigel: eindeutige Identifikatoren für Bibliotheken

SLUB ..... Sächsische Landesbibliothek - Staats- und Universitätsbibliothek Dresden

slub-kxp ..... Solr Index der SLUB mit eigenprozessierten Daten

SWB ..... Südwestdeutscher Bibliotheksverbund (Baden-Württemberg, Saarland, Sachsen)

TYPO3..... Software, CMS (Content-Management-System)

TYPO3-find ..... TYPO3-Erweiterung (Extension), Frontend-Lösung für Solr-Indizes

UBL ..... Universitätsbibliothek Leipzig

VD16 ..... Verzeichnis der im deutschen Sprachbereich erschienenen Drucke des 16. Jahrhunderts

VD17 ..... Verzeichnis der im deutschen Sprachraum erschienenen Drucke des 17. Jahrhunderts

VD18 ..... Verzeichnis der im deutschen Sprachraum erschienenen Drucke des 18. Jahrhunderts

VZG..... Verbundzentrale des Gemeinsamen Bibliotheksverbund

ZDB ..... Zeitschriftendatenbank

Abbildungsverzeichnis

Tabellenverzeichnis

Verzeichnis der Anhänge

Die Anhänge 2-10 liegen in elektronischer Form auf der beiliegenden CD vor.

Anhang 1: Anleitungsseite der Webapplikation im SLUB-Intranet

WebApplikation RankingTest als Unterstützung bei der Verbesserung des Rankings im SLUB-Katalog

Die Qualität eines Bibliothekskatalogs wird u.a. daran gemessen, ob er Medientitel, die inhaltlich zur Suchanfrage passen, in ausreichendem Maß und einer für den Anfragenden sinnvollen Sortierung anzeigt. Discovery-Systeme arbeiten, ähnlich wie Internetsuchmaschinen, nach dem Best-Match-Verfahren. Dabei erfolgt die Erstellung der Treffermenge und deren Sortierung durch mathematische Algorithmen.

Die Bildung der Treffermenge ist der erste Schritt zur Anzeige des Suchergebnisses. Sie wird u.a. beeinflusst durch die Auswahl der durchsuchten Solr-Felder und die Angaben, mit welcher Genauigkeit die Suchanfrage zu beantworten ist. Es kann z.B. festgelegt werden, wie viele Begriffe (Terme) der Suchanfrage im Datensatz enthalten sein müssen, um in die Treffermenge aufgenommen zu werden. Wird dieser Wert hoch angesetzt, steigt die Genauigkeit (Precision), aber die Menge der ausgegeben Treffer (Recall) sinkt und es besteht die Gefahr, relevante Treffer auszuschließen.

Die Treffermenge muss anschließend in ein Ranking gebracht werden, welches die Treffer mit der höchsten Bedeutung für diese Suchanfrage an die vordersten Plätze der Trefferliste rückt.

Beim Ranking der Treffer spielt das Boosting der verwendeten Felder eine entscheidende Rolle. Durch Boostfaktoren wird der berechneten Bedeutung eines Terms mehr Gewicht gegeben, wenn er in einem bestimmten Feld gefunden wird. Der Treffer rutscht dann in der Trefferliste nach oben.

Bisher konnten die Boostfaktoren nur in der Konfiguration der TYPO3-find-Settings geändert werden, was Programmieraufwand bedeutet. Die WebApplikation RankingTest soll es SLUB-MitarbeiterInnen ermöglichen, die Boostfaktoren für verschiedene Felder

- parallel, ohne Programmieraufwand auszuprobieren,
- deren Einfluss auf die Rangfolge der Trefferliste zu beobachten und
- die Beurteilungen in vergleichbarer Weise dokumentieren zu können (CSV-Download).

Es geht nicht darum, zu prüfen, ob die WebApplikation mit ihren Voreinstellung das richtige Ranking ausgibt. Vielmehr sollen für ein und dieselbe Anfrage verschiedene Boost-Faktoren in den Feldern ausprobiert und die Änderungen im Ranking beurteilt werden.

Diese WebApplikation ist ein erster Prototyp und noch erweiterbar. Denkbar wären folgende Aspekte

- gleiches Design wie der SLUB-Katalog (Trefferanzeige, Facetten ...)
- mehr Felder mit Boostfaktoren-Wahl
- Highlighting der Terme im Suchfeld

Ihr Feedback zu RankingTest würde es ermöglichen, diese WebApplikation weiter zu entwickeln. Dazu sind Sie herzlich eingeladen.

Vielen Dank für Ihre Zeit und Ideen!

Ulrike Schünemann

Team Datenmanagement
Abteilung 3 – Bestandsaufbau und Metadaten
Ref. 3.4 – Datenmanagement
Tel.: 0351 – 4677 338

ulrike.schuenemann@slub-dresden.de

Nutzungsanleitung

Adresse der WebApp: http://sdvdmgtest02.slub-dresden.de/RankingTest.html

Startseite der Applikation

Die aktuelle Konfiguration des SLUB-Katalogs verwendet bei einer Suche, die mit einem Wort beginnt weniger Suchfelder als bei der Suche, die mit einer Zahlenfolge beginnt. Diese Suche wird durch den Prototyp der WebApplikation nachgebildet. Voreingestellt sind die aktuell verwendeten Boostfaktoren. Der Prototyp läuft (noch) außerhalb der gewohnten SLUB-Katalog-Umgebung.

Die Liste der Felder, für welche die Boostfaktoren geändert werden können, ist in der Testphase auf die für die Suche verwendeten Titel-, Autoren- und einige formale Felder begrenzt.

⚠ Diese erste Variante der Applikation ist nur zur Suche nach Wörtern bzw. Wort/Zahlen-Kombinationen mit Wort am Anfang und NICHT für die Phrasensuche geeignet.

Eingabe Sucheterme / Boosts

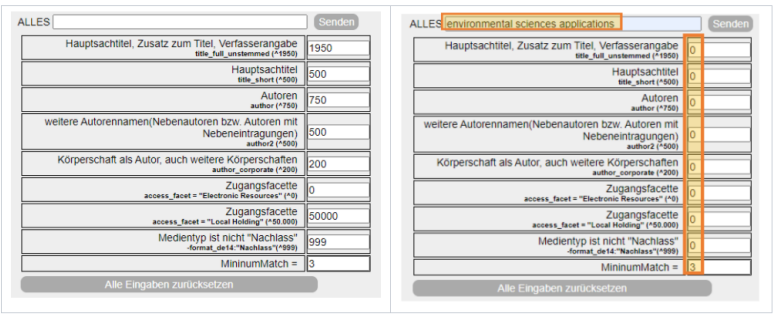
Im Suchschlitz hinter ALLES werden die **Suchterme** eingegeben

Es folgen die Angaben mittels denen dem Vorkommen der Terme in den Suchfeldern mehr Gewicht gegeben werden soll **(Boosts)**. Die bisherigen Sucheinstellungen werden als Default übergeben, sollten aber natürlich für das Testen verschiedener Varianten überschrieben werden.

Der Klick auf "**Senden** " fragt die gleichen SolrShards wie der SLUB-Katalog ab. Bei gefunden Treffern werden die Anzahl der Treffer und die ersten 10 Treffer angezeigt. (vgl. nächste Bilder)

Das Überschreiben der Suchworte und Absenden löst eine neue Suche mit den gleichen Boost-Einstellungen aus.

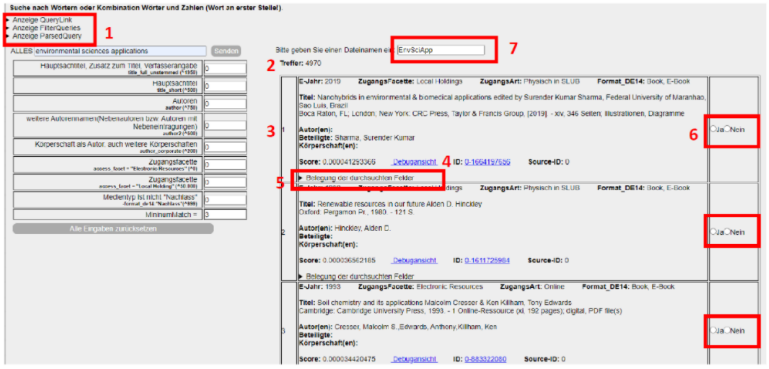
⚠"Alle Eingaben zurücksetzen" versetzt sowohl Suchfeld als auch Boosts in den Anfangszustand.



Anhang 1: Anleitungsseite der Webapplikation im SLUB-Intranet

Anzeige der Treffer und Beurteilung

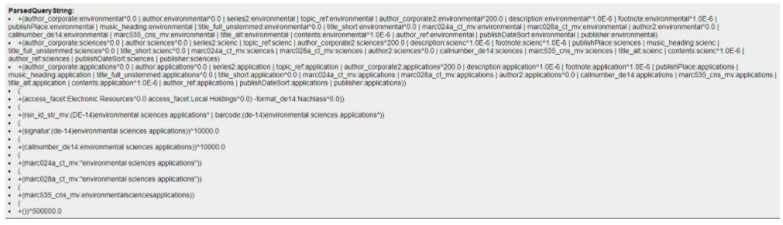
1. erweiterbare Anzeige von
   a. dem gebildeten QueryLink (die URL)
   b. den verwendeten FilterQueries
   c. des ParsedQueryStrings
   → öffnen sich durch Klick
2. Anzahl der erzielten Treffer
3. Anzeige der Treffer 1 bis 10 (wenn Treffermenge >= 10)
   (Zur besseren Übersicht sind die Angaben zu den Titeln vorerst gering gehalten.)
4. Links hinter **ID** bzw. **Debugansicht** als Zugang zum SLUB-Katalog
5. erweiterbare Anzeige "Belegung der durchsuchten Felder"
   → öffnet sich durch Klick
6. Möglichkeit zur Bewertungsabgabe:
   Relevant? Ja/Nein-Auswahl
7. Eingab für Dateinamen für Download-Datei (.csv)
   (Voreinstellung: test für test.csv)



Beispiel Anzeige QueryLink (URL)

Beispiel Anzeige FilterQueries

Beispiel Anzeige ParsedQueryString

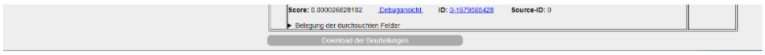


Beispiel Anzeige Belegung der durchsuchten Felder

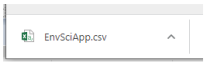
Download der Beurteilungen

Durch Betätigung des Downlaod-Button am Ende der Tabelle wird die Datei heruntergeladen.

⚠ Es werden nur Treffer mit einer Beurteilung in die Datei übernommen

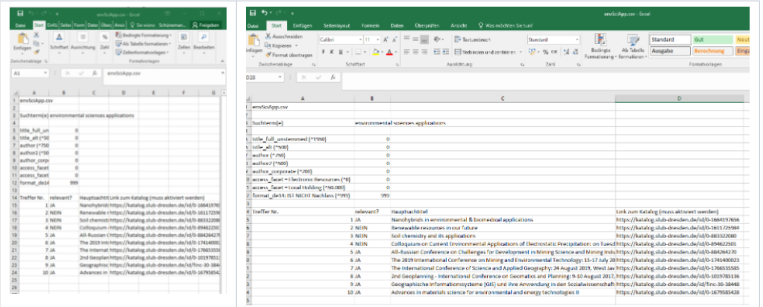


Durch Klick auf die Dateianzeige wird sie je nach eigenen Einstellungen/Angaben lokal abgespeichert oder das Öffnen der Datei angeboten.



Sie sehen in der Datei

- den Dateinamen
- die Suchterme
- die verwendeten Boost (hier nur unter Angabe der SolrBezeichnungen)
- die Trefferliste mit
  - lfd. Nummer
  - Angabe zur Relevanz
  - Titel
  - Link zum Katalog.



Selbständigkeitserklärung

Ich erkläre hiermit, dass ich die vorliegende Abschlussarbeit selbstständig verfasst habe, ohne Benutzung anderer als der angegebenen Hilfsmittel. Aus fremden Werken wörtlich oder sinngemäß übernommenen Gedanken habe ich unter Angabe der Quellen gekennzeichnet.

Dresden, 11.02.2022, Ulrike Schünemann