Search - using PostgreSQL (all releases up to Juniper)

- Search and filters
  - Search using Query language (Instance, Holdings, Item segment)
    - Examples on CQL syntax (WIP)
    - Boolean search within records and across instance/holdings/item:
    - Search queries that currently do not seem to work
    - Item search
    - Sort
    - Retrieve UUIDs
  - More documentation on search indexes (from a developer's perspective)

Search and filters

FOLIO’s current technical approach for search is using PostgreSQL, JSONB columns, CQL and RAML Module Builder. Using Elastic Search as a long term solution is being explored Autumn 2020 - Spring 2021 - see more.

In inventory, we have implemented segmented control search where it's possible to toggle between searching on an Instance, Holdings or individual Item level. Search in the Instance segment supports search on bibliographic data. Search in the Holdings and Item segments are a combination of key data from the Instance record, combined with holdings and item specific data elements. The filters reflect each of the individual segments and include search on effective location.

This is work in development, but as of end of January the following search and filter options are present in FOLIO Snapshot:
Search and filter options can be combined. When searching in a filter, the search typed will be highlighted. E.g.:

Language

- English
- English, Old (ca. 450-1100)
- English
- English, Middle (1100-1500)

Material Type

- re
- electronic resource
- sound recording
- video recording

If more than 100 hits are retrieved, the first 100 will load, and then you will see a "Load more" button at the bottom of the page. When clicked, it will display the next 100:
From the FOLIO version Juniper onwards the concept of loading more records has been redesigned. You can navigate through the records in steps of 100 by clicking the “next/previous” buttons at the bottom of the page.

There is also a Query search function, which can search across records on all data elements. See below.
Search using Query language (Instance, Holdings, Item segment)

See list of property names in Query Search - Inventory Metadata Elements, or in the API Documentation in the /instance-storage/instances POST, /holding s-storage/holdings POST and /item-storage/items POST.

See more CQL tips written up by Julian Ladisch's Explain CQL string matching. For experts: RAML Module Builder CQL documentation.

See: Tips: Use Query Search in Inventory for export MARC records from FOLIO

See list of defined supporting database indexes: db_scripts/schema.json – Skip to the **tableName**: "instance", "tableName": "holdings_reco rd" or "tableName": "item" section and look into the "uniqueIndex", "index", and "fullTextIndex" subsections. The id field is automatically indexed and not listed in schema.json.

See misc. test of search performance in Bug fest Goldenrod (https://bugfest-goldenrod.folio.ebsco.com) in Document Inventory Search Response time (Q3 Timebox)

**Tips:** When searching by an ID, in the Query search it's better to use "==" as it uses the B-tree index to search, whereas a CQL query using "=" will result in a full text search, which is not efficient for searching when there is already an index on the ID.

Examples on CQL syntax (WIP)

<table>
<thead>
<tr>
<th>Query search</th>
<th>Search result</th>
</tr>
</thead>
<tbody>
<tr>
<td>id == &quot;f31a36de-fcfd-44f9-87ef-a55d06ad21ae&quot;</td>
<td>give me the instance record where UUID = f31a36de-fcfd-44f9-87ef-a55d06ad21ae</td>
</tr>
<tr>
<td>hrid == &quot;in000000009&quot;</td>
<td>give me the instance record where the HRID = in000000009 (this number will later be an 11 digit sequence)</td>
</tr>
<tr>
<td>source == &quot;folio&quot;</td>
<td>give me all instance records where metadata source = FOLIO</td>
</tr>
<tr>
<td>source == &quot;marc&quot;</td>
<td>give me all instance records where metadata source = MARC, which means that the instance record has an underlying MARC record in SRS</td>
</tr>
<tr>
<td>contributorsNames all &quot;Sosa Keita&quot;</td>
<td>give me all instance records where the contributor names contain both Sosa and Keita (any position, any order)</td>
</tr>
<tr>
<td>contributorsNames adj &quot;Sosa Omar&quot;</td>
<td>give me all instance records where the word Sosa is immediately followed by the word Omar in the contributor names ignoring whitespace and punctuation between words (phrase search)</td>
</tr>
<tr>
<td>identifiers =/@value/@identifierTypeId=&quot;01ca9cda-7027-4d64-abed-9e3c4943da2f&quot; (&quot;OCoLC&quot;)670473988</td>
<td>give me all instance records where OCLC number = (OCoLC)670473988 - result in BugFest Iris environment</td>
</tr>
<tr>
<td>identifiers =/@value/@identifierTypeId=&quot;01ca9cda-7027-4d64-abed-9e3c4943da2f&quot; (&quot;806014467&quot; OR &quot;560591246&quot;)</td>
<td>give me the two instance records with OCLC number = (OCoLC)806014467, (OCoLC)560591246 by using the OR boolean operator - result in BugFest Goldenrod environment</td>
</tr>
<tr>
<td>statisticalCodeIds =&quot;c6de6928-bb6e-4458-97f4-9145b27abb24&quot;</td>
<td>give me all instance records where the statistical code = books (University of Chicago) - result in BugFest Iris environment</td>
</tr>
<tr>
<td>title == dave *</td>
<td>give me all instance records where the resource title exactly matches against the resource title data element as: dave *</td>
</tr>
<tr>
<td>alternativeTitles=ATLA Proceedings</td>
<td>give me all instance records where the alternative title = ATLA Proceedings</td>
</tr>
<tr>
<td>series = &quot;cooperative information systems&quot;</td>
<td>give me all instance records where the series title = Cooperative information systems</td>
</tr>
<tr>
<td>publication = &quot;MIT Press&quot;</td>
<td>give me all instance records where the publisher = MIT Press</td>
</tr>
</tbody>
</table>
Boolean search within records and across instance/holdings/item:
Query search | Search result
--- | ---
publication = "MIT Press" and publication = "c2004" | give me all instance records where publisher = MIT Press AND publication year is c2004
publication = "MIT Press" and holdingsRecords.callNumber = "TK5105.88815 . A58 2004 FT MEADE" | give me all instance records where publisher = MIT Press AND the associated holdings records has holdings level call number = TK5105.88815 . A58 2004 FT MEADE
subjects = "history" or identifiers = "OCoLC" not publication = "2017" | give me all instance records where publisher = MIT Press AND the resource identifier is starting with OCoLC and does NOT have publication year = 2017
languages="eng" AND item.itemLevelCallNumber="TK5105" | give me all instance records where language = English AND the associated item records has item level call number starts with TK5105. The query includes wildcard search with an asterisk
identifiers =/@value/@identifierTypeld == "216b156b-215e-4839-a53e-ade35cb5702a" 10419/192087 | give me all instance records where resource identifier type = handle AND resource identifier = 10419/192087
source = "*" not source = "folio" Alternative CQL: id=* not source = "folio" cq.allRecords = "1" not source = "folio" | give me all instance records where metadata source is NOT Folio (but: MARC, and other source formats)
item.status="available" AND statusId = "26f5208e-110a-4394-be29-1569a8c84a65" | give me all records where item status = available AND instance status term = Uncataloged
item.barcode="10101" | give me the instance record the item with barcode 10101 belongs to

Search queries that currently do not seem to work

<table>
<thead>
<tr>
<th>Query search</th>
<th>Expected search result</th>
<th>Actual search result</th>
<th>Tested by / on environment</th>
<th>Comments</th>
</tr>
</thead>
</table>
discoverySuppress="false" AND holdingsRecords.discoverySuppress="false" AND item.discoverySuppress="false" AND item.barcode="null" | Instance, Holdings, Item are not suppressed (for discovery?) AND item barcode = blank example from 2020-01-30 Metadata Management Meeting notes | | The instance record JSON includes a boolean "discoverySuppress" which is set to false in my instance example record. Holding and item records don't have this elements in their response. Use Chrome's dev tools to verify. I'm not sure how to search for NULL values. If an item has no barcode assigned, the element is not included in the JSON response either. | cc Charlotte Whitt |

Item search

<table>
<thead>
<tr>
<th>Query search</th>
<th>Search result</th>
</tr>
</thead>
<tbody>
<tr>
<td>id!= NOT item.barcode=** NOT discoverySuppress=&quot;true&quot; NOT holdingsRecords.discoverySuppress=&quot;true&quot; NOT instance.discoverySuppress=&quot;true&quot;</td>
<td>give me all item records that have no barcode and are not suppressed for discovery on item, holdings and instance level. Note: Using NOT also matches if the field is not defined (null). Note: This query must run against the item API. Running a similar query against the instance API doesn't always work because item.barcode and item.discoverySuppress can match different items of the same instance, and these items may belong to a different holding than the holding used for matching holdingsRecords.discoverySuppress.</td>
</tr>
</tbody>
</table>

Sort

Change the sort field in the URL.

Example:
We need to use publicationPeriod.start because there can be multiple dateOfPublication entries in the publication array.

**Retrieve UUIDs**

For some CQL queries you need the UUID of the value, because you cannot search for the actual term. To do this, you can use the developer tools that are built into your browser. In Chrome, press F12 to open the developer tools and switch to the "network" tab. Clear out any previous data by pressing "clear" and refresh the Folio page. Please note, that you'll first have to open a detail instance record in the third pane. After the page has loaded, you should look in the network data for the record UUID. Select it to see the record data in JSON format.

<table>
<thead>
<tr>
<th>RefData</th>
<th>Value</th>
<th>UUID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Instance status types</td>
<td>Batch Loaded</td>
<td>52a2ff34-2a12-420d-8539-21aa8d3cf5d8</td>
</tr>
<tr>
<td></td>
<td>Cataloged</td>
<td>9634a5ab-9228-4703-baf2-4d12ebc77d56</td>
</tr>
<tr>
<td></td>
<td>Not yet assigned</td>
<td>f5cc2ab6-bb92-4cab-b83f-5a3d09261a41</td>
</tr>
<tr>
<td></td>
<td>Other</td>
<td>2a340d34-6b70-443a-bb1b-1b8d1c65d862</td>
</tr>
<tr>
<td></td>
<td>Temporary</td>
<td>daf2681c-25af-4202-a3fa-e58fd806183</td>
</tr>
<tr>
<td></td>
<td>Uncataloged</td>
<td>26f5208e-110a-4394-be29-1569a8d84a65</td>
</tr>
</tbody>
</table>

**More documentation on search indexes (from a developer's perspective)**

Following list of resources is gathered by Julian Ladisch:

Existing documentation:

- [https://dev.folio.org/faqs/explain-cql/](https://dev.folio.org/faqs/explain-cql/)
- [https://dev.folio.org/faqs/explain-database-schema/](https://dev.folio.org/faqs/explain-database-schema/)
- [https://github.com/folio-org/raml-module-builder#the-post-tenant-api](https://github.com/folio-org/raml-module-builder#the-post-tenant-api) about likeIndex, ginIndex, uniqueIndex, index, fullTextIndex
- [https://github.com/folio-org/raml-module-builder#cql-relations](https://github.com/folio-org/raml-module-builder#cql-relations) with several sections about CQL
- [https://github.com/folio-org/raml-module-builder#cql2pgjson-multi-field-index](https://github.com/folio-org/raml-module-builder#cql2pgjson-multi-field-index) with several sections about CQL2PgJSON indexes

All fields can be indexed, simply add an index entry to schema.json on GitHub. The indexing is the same for all institutions and a change requires a new release of the module.

Currently institutions cannot customize indexing at runtime and cannot have different indexing. GBV libraries have extensive indexing customization in the inventory of their current system (OCLC LBS) and will likely request this for FOLIO. Index customization use cases for inventory should be discussed in the metadata management sig.