DSpace, Solr and Postman

Art Lowel
Access Solr using Postman: SSH Tunnel

Solr is only available from localhost

To use postman to access a remote Solr, first open a tunnel:

```sh
ssh ${server} -L ${local-port}:localhost:${remote-port}
```

e.g.

```sh
ssh art@server.com -L 9999:localhost:8080
```
Environments in Postman allow you to specify variables.

I use them for different ports on localhost:

- 8080 → a local instance
- 9090 → remote instance 1
- 9999 → remote instance 2

They all set the `{{base}}` variable.
Postman basics: Globals

I also use 2 global variables:
- `{{context}}` → to set the context-path
- `{{core}}` → the Solr core I’m working with

They vary by project

I usually set these when I start working with postman for a session

You can also set them by selecting text and right clicking
Postman basics: Collections

Collections allow you to group and store requests.

You can use the global and environment variables.

You can specify variables on a collection level as well.
Common queries
Common Solr queries: Get the number of records

Find the number of records in a Solr core:

```
{{base}}/{{context}}/{{core}}/select?q=:*:*&rows=0&wt=json
```

- `q=:*:*` → search for everything
- `rows=0` → don’t return regular results
- `wt=json` → in json format

The number of matches is in the `response.numFound` field
Common Solr queries: Commit

Solr doesn’t update its index when a new doc is added, but after a certain time, or after a certain number of new docs have been added (15 minutes or 10,000 docs by default)

This command will force it to update:

```
{{base}}/{{context}}/{{core}}/update?stream.body=<commit></commit>
```

- use the `/update` endpoint
- `stream.body=<commit></commit>`
Common Solr queries:
Delete

```
{{base}}/{{context}}/{{core}}/update?
stream.body=<delete><query>id:11975 AND type: 2</query></delete>&commit=true
```

- use the `/update` endpoint
- `stream.body=<delete><query>...</query></delete>` → fill in anything that you can put in the regular `q` parameter
- `commit=true` → commit when you’re done
Search core queries
Search: Standard params

A combination of parameters that is a useful starting point for most search queries:

```
{{base}}/{{context}}/search/select?
q=*&rows=10&wt=json&fq=not-
withdrawn:true&fq=not-
discoverable:false&fq=search.resourcetype:2&fq=read:(g0)&fl=handle,title,author
```
Search: Standard params

- `q=*:*`
- `rows=10` → for search queries I usually want to see results
- `wt=json`
- `fq=-withdrawn:true` → exclude withdrawn items
- `fq=-discoverable:false` → exclude items that aren’t discoverable
- `fq=search.resourcetype:2` → only return items
- `fq=read:(g0)` → only return things anonymous users can access
- `fl=handle,title,author` → only include the handle, title and author
Search: Specific Handle

Show what’s indexed about a specific DSpace object:

```
{{base}}/{{context}}/search/select?
q=::*&rows=1&wt=json&fq=handle:1234567/1234
```

- `q=::*`
- `rows=1` → we want to see the result here
- `wt=json`
- `fq=handle:1234567/1234` → only return the object with this handle
Search: Facet by collection

Count the number of Items in each collection:

```
{{base}}/{{context}}/search/select?
q=*&rows=0&wt=json&fq=-withdrawn:true&fq=-
discoverable:false&fq=search.resourcetype:2&fq=read:
(g0)&facet=true&facet.field=location.coll
```

Standard params +

- **rows=0** → not interested in regular results
- **facet=true** → enable facets
- **facet.field=location.coll** → facet by collection
Search: Facet by type

Get a breakdown of the repository by resource type:

```
{{base}}/{{context}}/search/select?
q=*:*&rows=0&wt=json&facet=true&facet.field=search.resourcetype&facet.mincount=1
```

- `q=*:*`
- `rows=0`
- `wt=json`
- `facet=true`
- `facet.field=search.resourcetype` → facet by resourcetype
- `facet.mincount=1` → only include facets with at least than 1 result
Statistics core queries
Statistics: Standard params

A combination of parameters that is a useful starting point for most statistics queries:

```
{{base}}/{{context}}/statistics/select?
q=*&rows=0&wt=json&fq=-isBot:true&fq=-(statistics_type:* AND -statistics_type:view)&fq=-(bundleName:* AND -bundleName:ORIGINAL)
```
Statistics: Standard params

- q=::*
- rows=0
- wt=json
- fq=-isBot:true → exclude bots
- fq=-(statistics_type:* AND -statistics_type:view) → exclude everything that has a statistics_type other than ‘view’.
  - Reason: includes old dspace 1.x stats that didn’t have a statistics type
- fq=-(bundleName:* AND -bundleName:ORIGINAL) → exclude everything that has a bundleName other than ‘ORIGINAL’
  - Reason: includes views as well, they won’t have a bundleName
Statistics: Most Active IPs

List the most active IPs on your repository. Useful for detecting bots that aren’t flagged yet.

```
{{base}}/{{context}}/statistics/select?q=*&rows=0&wt=json&fq=!isInternal:true&fq=!isBot:true&fq=!(statistics_type:* AND -statistics_type:view)&fq=!(bundleName:* AND -bundleName:ORIGINAL)&facet=true&facet.limit=20&facet.mincount=1&facet.field=ip
```
Statistics: Most Active IPs

Standard params +

- `facet=true` → enable facets
- `facet.limit=20` → show the top 20
- `facet.mincount=1` → only include IPs with at least 1 hit
- `facet.field=ip` → facet by IP
Statistics: IP By Day

If you have a suspicious IP, but you’re not sure it’s a bot, it can be helpful to check its activity grouped by day.

Bots often either:
• download a huge amount in a few consecutive days, nothing before or after
• download a similar small amount each day:
  • e.g. exactly 150 docs each day

{{base}}/{{context}}/statistics/select?q=*&rows=0&wt=json&fq=-isBot:true&fq=-(statistics_type:* AND -statistics_type:view)&fq=-(bundleName:* AND -bundleName:ORIGINAL)&fq=ip:192.168.95.74&facet=true&facet.date=time&facet.date.gap=%2B1DAY&facet.date.start=2016-01-01T00:00:00.00Z/DAY&facet.date.end=2017-01-01T00:00:00.00Z/DAY
Statistics: IP By Day

Standard params +

- **facet=true**
- **facet.date=time**
- **facet.date.gap=%2B1DAY → is +1DAY URL encoded.**
  - Note that you can URL en/decode from the right-click menu when selecting a string in postman
  - **WEEK, MONTH** and **YEAR** are also valid
- **facet.date.start=2016-01-01T00:00:00.00Z/DAY →**
  - start at 2016-01-01 rounded to the start of the day
- **facet.date.end=2017-01-01T00:00:00.00Z/DAY →**
  - end at 2016-01-01 rounded to the start of the day
Statistics: Most Active Countries

Using pivot queries you can facet multiple times in the same query. In this case we want to know both the number views and downloads per country:

```
{{base}}/{{context}}/statistics/select?
q=*&rows=0&wt=json&fq=-isBot:true&fq=-
(statistics_type:* AND -statistics_type:view)&fq=-
(bundleName:* AND -
bundleName:ORIGINAL)&facet=true&facet.limit=50&facet.pivot=countryCode,type&facet.pivot.mincount=1
```
Statistics: Most Active Countries

Standard params +

- facet=true
- facet.limit=50
- facet.pivot=countryCode,type → facet first by country, then by type
- facet.pivot.mincount=1 → same as facet.mincount, but for pivot queries
Thanks for listening!

Questions?