Exercise 6.1 – Create DSE Search Index and Query

This purpose of this exercise is to teach you to create indexes using CQL as well as to familiarize you with the Solr UI.

In this exercise, you will:

- Create a Search core based on a Cassandra table using a CQL command.
- Familiarize yourself with the Solr Admin UI

Here's what you'll do:

Step 1: Create indexes
Step 2: Launch and Explore the Solr UI
Step 3: Execute five queries via the Solr UI

Step 1: Create indexes

In this step you will be creating an index and then altering it.

1. The first alter statement sets up a schema with a tokenizer and filtering. It has been broken out directly below to make it more readable.

```
ALTER SEARCH INDEX SCHEMA ON killrvideo_test.videos
ADD types.fieldtype[
                    @class='org.apache.solr.schema.TextField',
                    @name='TextField']
WITH'{"analyzer":
            {"tokenizer":{"class":"solr.StandardTokenizerFactory"},
             "filter":[{"class": "solr.StandardFilterFactory"},
                        {"class": "solr.LowerCaseFilterFactory"},
                        {"class": "solr.StopFilterFactory"}
                       ]
             }
         }';
```

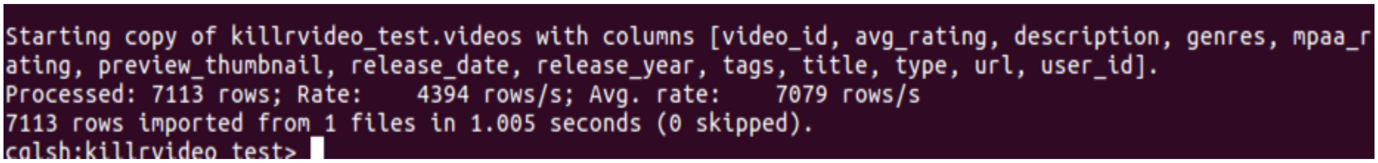
2. The remaining alter statements set up the data type on each field in the table. The last alter statement makes sure we can do *facet searching on genres* by setting `docValues = true`.

3. At the end of the alter statements, the `RELOAD` statement makes the new statement configuration active while `REBUILD` constructs the index data, or “core”.

4. Reload the videos table using the following statement:

```
copy videos from '/projects/session2/csv/videos.csv';
```

   You should see a message like the following when the file has finished loading:



```
Starting copy of killrvideo_test.videos with columns [video_id, avg_rating, description, genres, mpaa_r
ating, preview_thumbnail, release_date, release_year, tags, title, type, url, user_id].
Processed: 7113 rows; Rate:    4394 rows/s; Avg. rate:    7079 rows/s
7113 rows imported from 1 files in 1.005 seconds (0 skipped).
cqlsh:killrvideo_test>
```

5. Run a `SELECT` statement to ensure the table has data in it. If not, there is a pre-installed keyspace called *killrvideo_search* that you can use for the DSE search exercises.

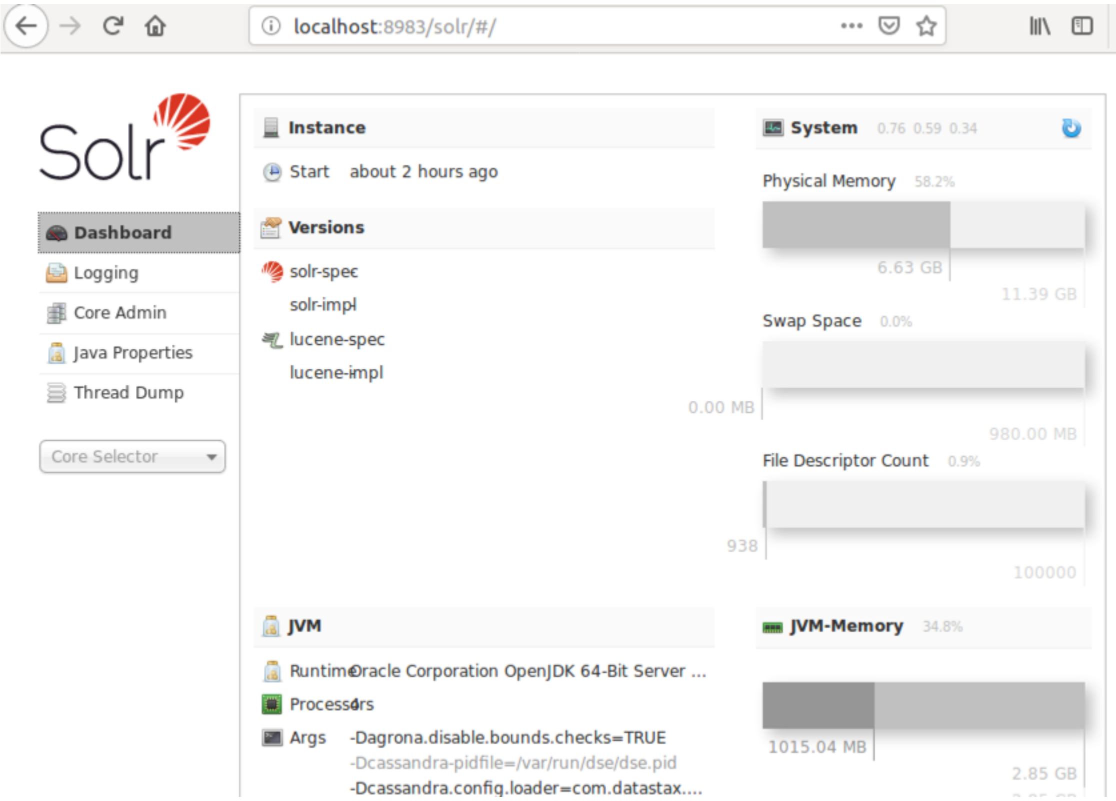
6. Let’s create an index. Type the following code into *cqlsh*:

```
CREATE SEARCH INDEX ON killrvideo_test.videos;
ALTER SEARCH INDEX SCHEMA ON killrvideo_test.videos ADD
types.fieldtype[@class='org.apache.solr.schema.TextField',
@name='TextField'] WITH
'{"analyzer":{"tokenizer":{"class":"solr.StandardTokenizerFactory"},
"filter":[{"class": "solr.StandardFilterFactory"}, {"class":
"solr.LowerCaseFilterFactory"},{"class": "solr.StopFilterFactory"}]}}';
ALTER SEARCH INDEX SCHEMA ON killrvideo_test.videos SET
field[@name='mpaa_rating']@type = 'TextField';
ALTER SEARCH INDEX SCHEMA ON killrvideo_test.videos SET
field[@name='type']@type = 'TextField';
ALTER SEARCH INDEX SCHEMA ON killrvideo_test.videos SET
field[@name='url']@type = 'TextField';
ALTER SEARCH INDEX SCHEMA ON killrvideo_test.videos SET
field[@name='title']@type = 'TextField';
ALTER SEARCH INDEX SCHEMA ON killrvideo_test.videos SET
field[@name='description']@type = 'TextField';
ALTER SEARCH INDEX SCHEMA ON killrvideo_test.videos SET
field[@name='tags']@type = 'TextField';
ALTER SEARCH INDEX SCHEMA ON killrvideo_test.videos SET
field[@name='genres']@docValues = 'true';
RELOAD SEARCH INDEX ON killrvideo_test.videos;
REBUILD SEARCH INDEX ON killrvideo_test.videos;
```

Step 2: Launch and Explore the Solr UI

The purpose of this step is to familiarize you with the Solr UI.

1. Open a web browser and use your personal URL: http://<NODE1-IP>:8983/solr/



Start about 2 hours ago

Runtime Oracle Corporation OpenJDK 64-Bit Server ...

Processors

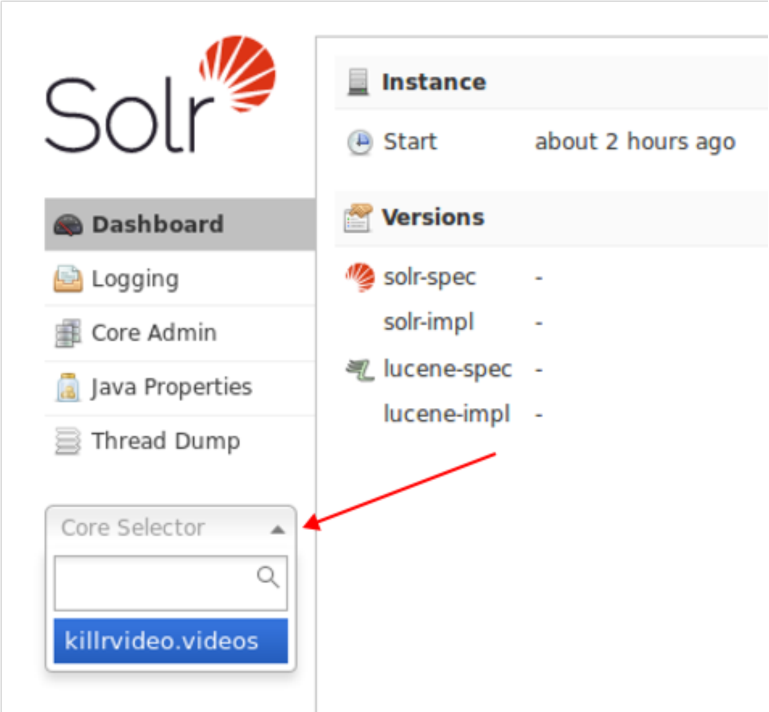
Args -Dagrona.disable.bounds.checks=TRUE
-Dcassandra-pidfile=/var/run/dse/dse.pid
-Dcassandra.config.loader=com.datastax....

The Solr UI will be displayed and the Dashboard page will be visible by default. The Dashboard shows information about your system, swap space and JVM memory.

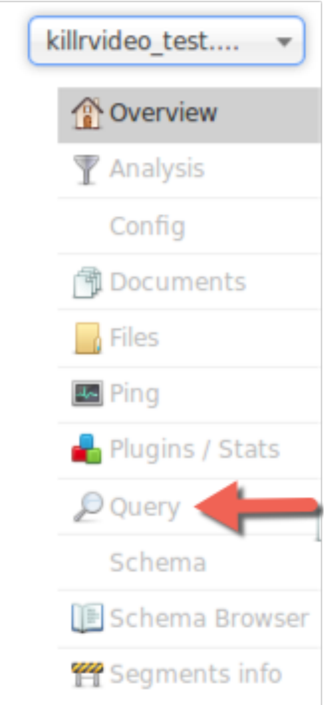
The left-hand navigation bar has options for viewing Solr logs, administering cores, and viewing Java properties and thread dumps.

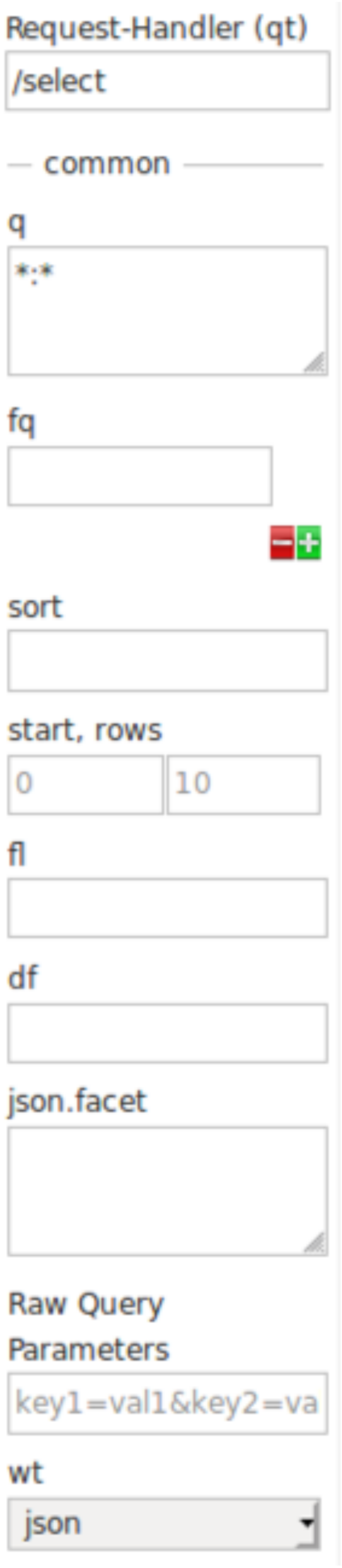
Just below the options is a drop-down box that says “Core Selector”. The purpose of this drop-down box is to select a core on which you would like to execute queries.

2. Go to *Core Selector* and select *killrvideo_test.videos* in the drop-down menu list.



3. A navigation bar associated with the text box will appear. Select Query so you can enter and execute a query.





Request-Handler (qt)

/select

— common —

q

:

fq

sort

start, rows

0 10

fl

df

json.facet

Raw Query Parameters

key1=val1&key2=va

wt

json

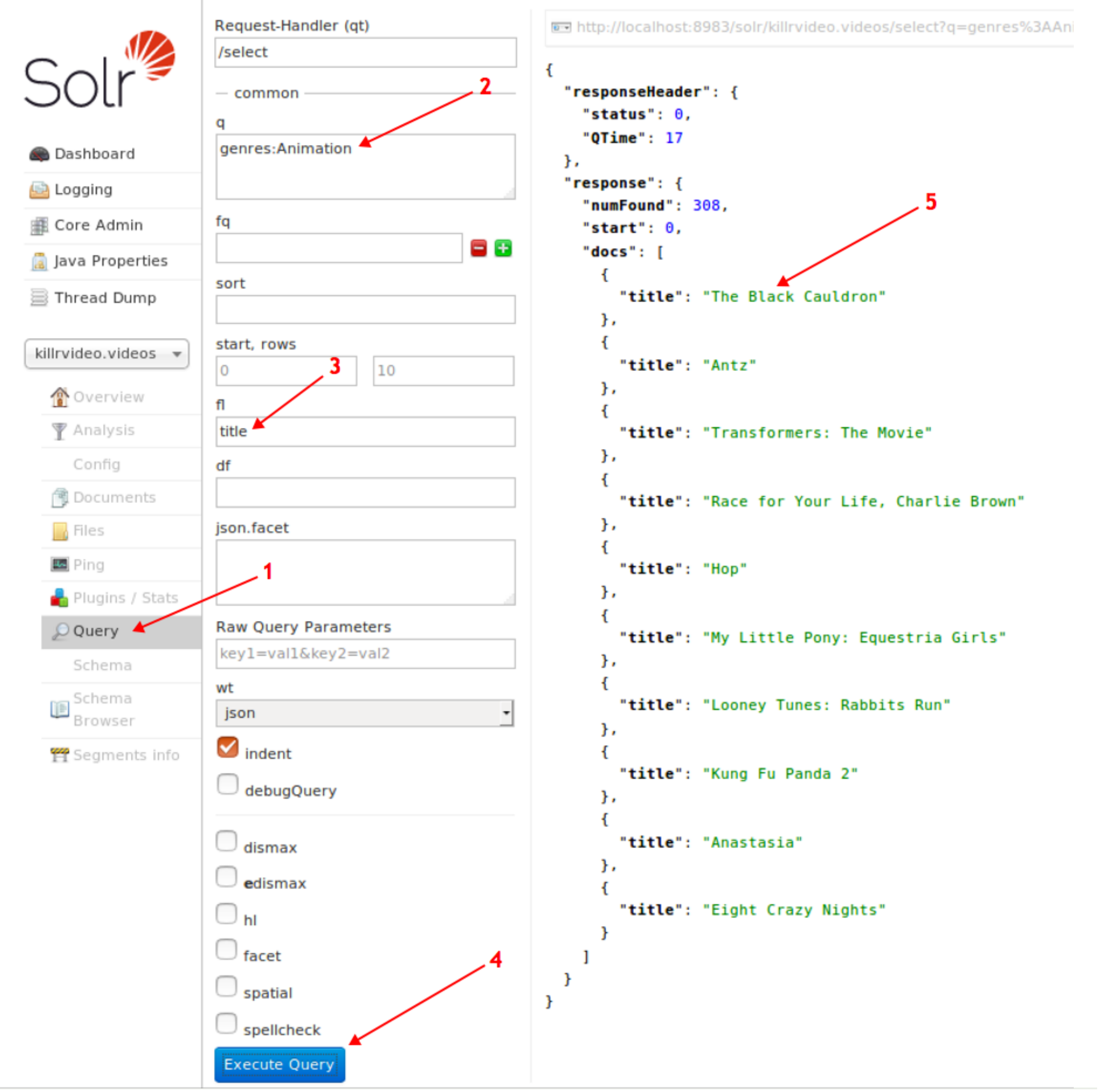
A query bar is displayed to the left. Note the meaning of each query text box below:

1. q = <Solr predicate> (required)
2. fq = <filter query (explained later)
3. sort = <sort fields> asc | desc
4. start = <which row to start (for paging)>
5. fl = <fields to return>
6. rows = <number of rows to return (default 10)>
7. wt = <writer type (xml, json, etc)>

Step 3: Execute five queries via the Solr UI

The purpose of this step is to give you some hands-on experience with executing queries in the Solr UI. First, we are going to create query to search for all films where the genre is animation. It will return the title field.

1. Enter in the textbox below *q: genres:Animation.*
2. Enter in the textbox below *fl: title.*
3. Select *Execute Query.*
4. The titles should be displayed in json format at shown below:



Request-Handler (qt): /select

— common —

q: genres:Animation

fq:

sort:

start, rows: 0 10

fl: title

df:

json.facet:

Raw Query Parameters: key1=val1&key2=val2

wt: json

☑ indent

□ debugQuery

□ dismax

□ edismax

□ hl

□ facet

□ spatial

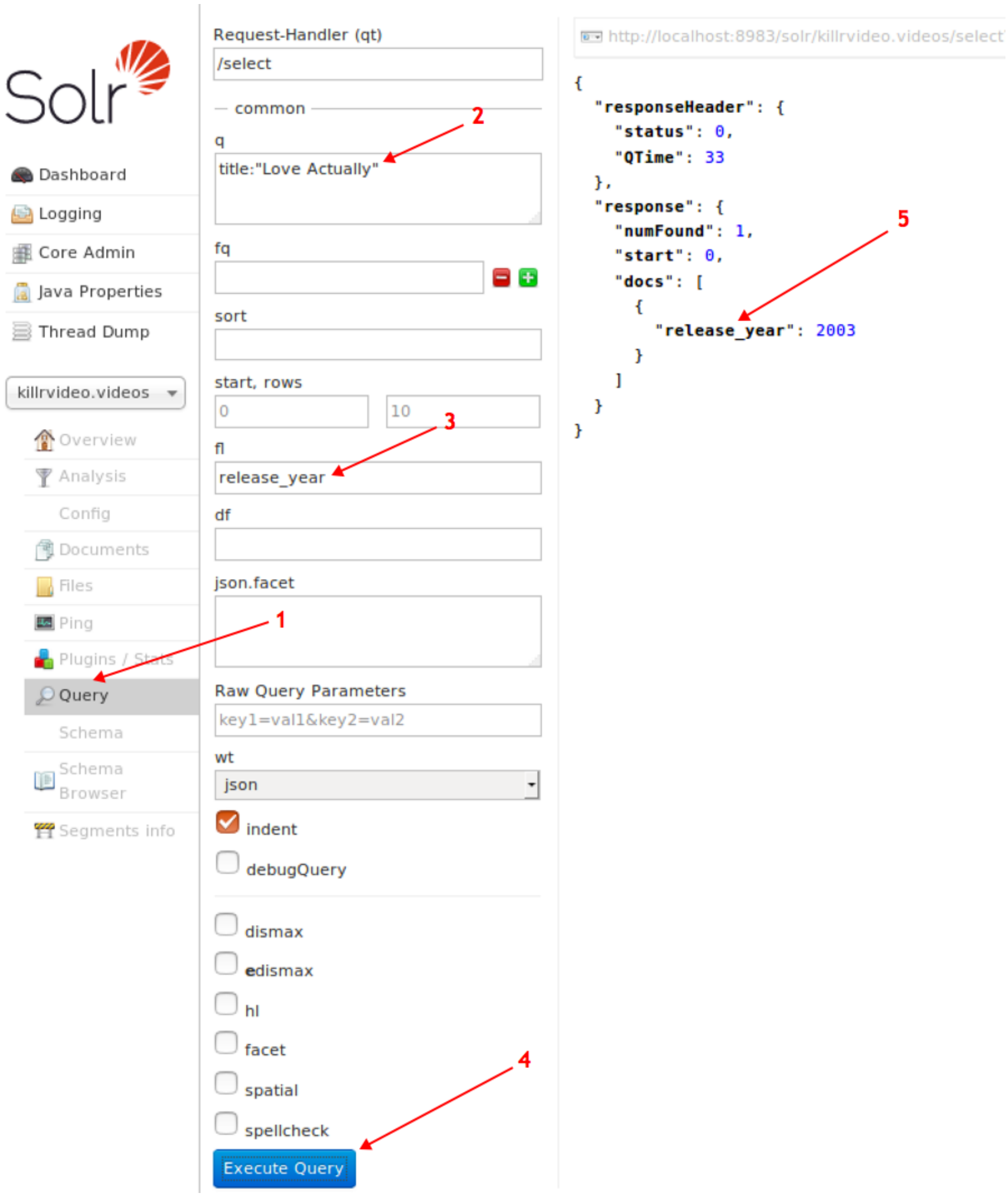
□ spellcheck

http://localhost:8983/solr/killrvideo.videos/select?q=genres%3AAn

```
{
  "responseHeader": {
    "status": 0,
    "QTime": 17
  },
  "response": {
    "numFound": 308,
    "start": 0,
    "docs": [
      {
        "title": "The Black Cauldron"
      },
      {
        "title": "Antz"
      },
      {
        "title": "Transformers: The Movie"
      },
      {
        "title": "Race for Your Life, Charlie Brown"
      },
      {
        "title": "Hop"
      },
      {
        "title": "My Little Pony: Equestria Girls"
      },
      {
        "title": "Looney Tunes: Rabbits Run"
      },
      {
        "title": "Kung Fu Panda 2"
      },
      {
        "title": "Anastasia"
      },
      {
        "title": "Eight Crazy Nights"
      }
    ]
  }
}
```

Execute a query that gives us the release year for the film *"Love Actually"*.

1. Select `Query` from the left-side menu list.
2. Enter in the textbox below *q: title: "Love Actually".*
3. Enter in the textbox below *fl: release_year.*
4. Select `Execute Query`.
5. This will show that the movie *"Love Actually"* was released in *2003.*



Request-Handler (qt): /select

q: title:"Love Actually"

fq:

sort:

start, rows: 0, 10

fl: release_year

df:

json.facet:

Raw Query Parameters: key1=val1&key2=val2

wt: json

☑ indent

☐ debugQuery

☐ dismax

☐ edismax

☐ hl

☐ facet

☐ spatial

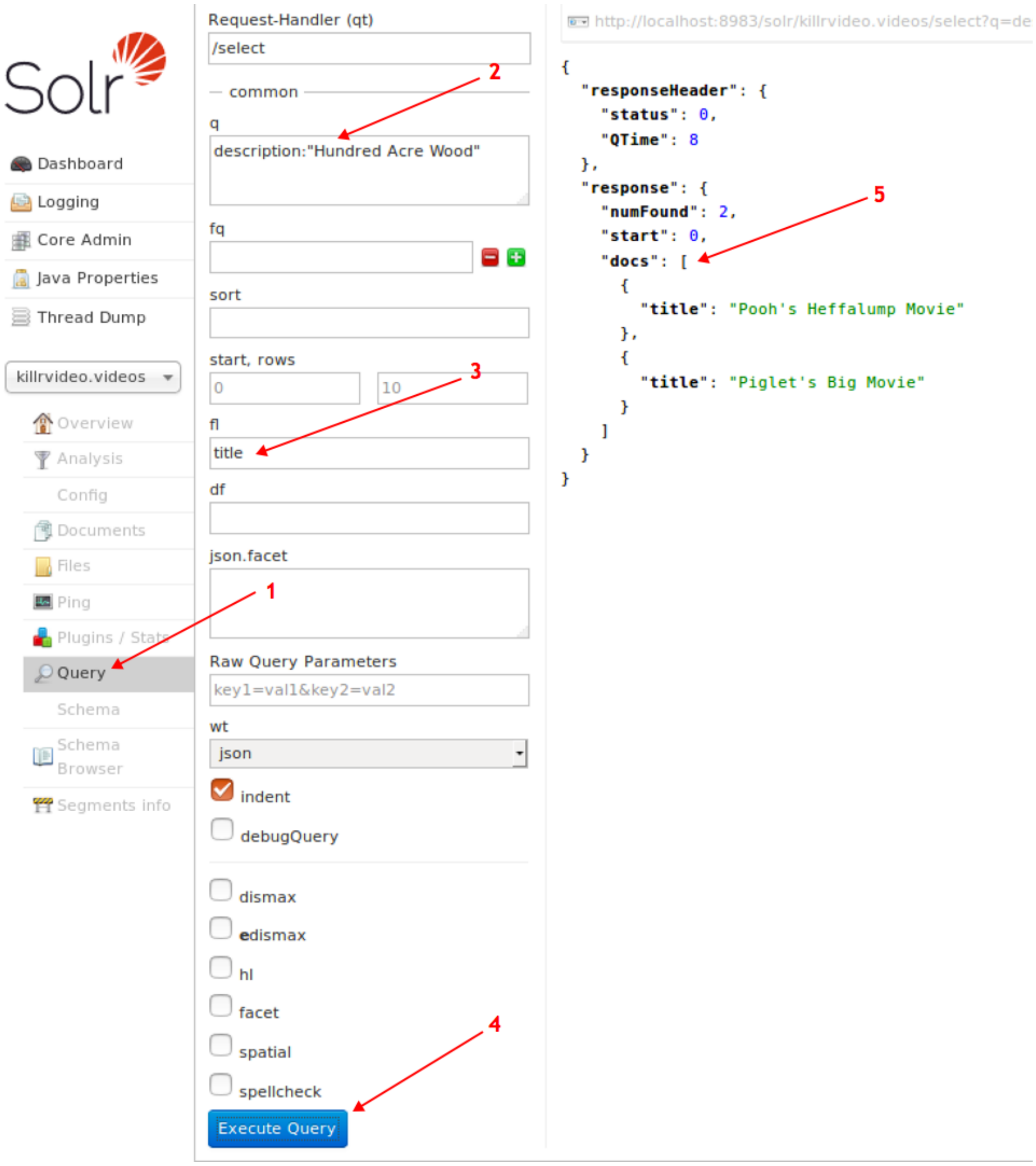
☐ spellcheck

http://localhost:8983/solr/killrvideo.videos/select

```
{
  "responseHeader": {
    "status": 0,
    "QTime": 33
  },
  "response": {
    "numFound": 1,
    "start": 0,
    "docs": [
      {
        "release_year": 2003
      }
    ]
  }
}
```

Next, execute a query that shows the title of a film with a specific description.

1. Select `Query` from the left-side menu list.
2. Enter in the textbox below *q: description: “Hundred Acre Wood”.*
3. Enter in the textbox below *fl: title.*
4. Select `Execute Query`.



Request-Handler (qt): /select

q: description:"Hundred Acre Wood"

fq:

sort:

start, rows: 0 10

fl: title

df:

json.facet:

Raw Query Parameters: key1=val1&key2=val2

wt: json

☑ indent

☐ debugQuery

☐ dismax ☐ edismax ☐ hl ☐ facet ☐ spatial ☐ spellcheck

http://localhost:8983/solr/killrvideo.videos/select?q=de

```
{
  "responseHeader": {
    "status": 0,
    "QTime": 8
  },
  "response": {
    "numFound": 2,
    "start": 0,
    "docs": [
      {
        "title": "Pooh's Heffalump Movie"
      },
      {
        "title": "Piglet's Big Movie"
      }
    ]
  }
}
```

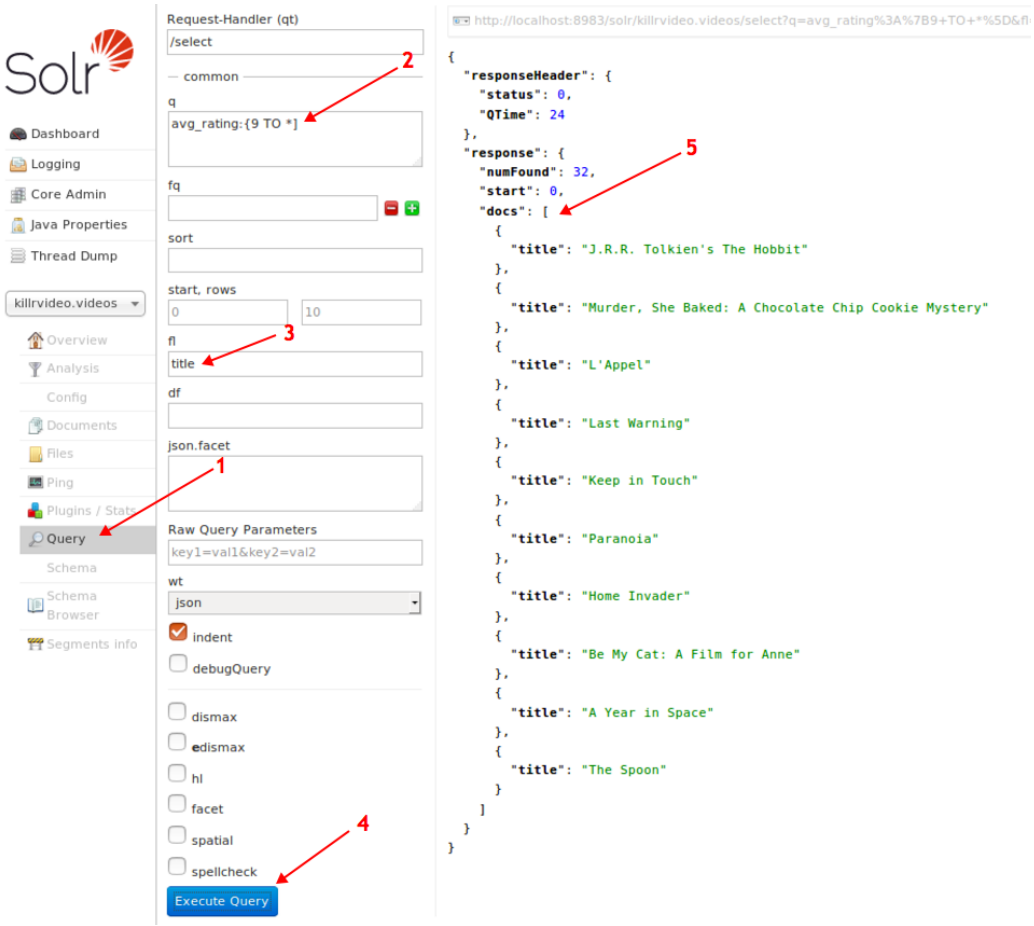
Execute a query that displays all movies with an average rating that is higher than 9. We'll use our inclusion and exclusion brackets for this query. Note the bullets and examples below before starting on step 1:

- Range searches follow the syntax ***Value* TO *Value***
  - Less than -- * **TO *Value***
  - Greater than -- ***Value* TO** *
- Square bracket for inclusive bound -- **[ ]**
- Curly brace for exclusive bound -- **{ }**

Example: Greater than or equal to 2000, less than 2015.

: Syntax = release_year:[2000 TO 2015}

1. Select `Query` from the left side menu list.
2. Enter in the textbox below *q: avg_rating: {9 TO *].*
3. Enter in the textbox below *fl: title.*
4. Select `Execute Query`.



Request-Handler (qt): /select

common

q: avg_rating:{9 TO *]

fq

sort

start, rows: 0 | 10

fl: title

df

json.facet

Raw Query Parameters: key1=val1&key2=val2

wt: json

☑ indent

☐ debugQuery

☐ dismax

☐ edismax

☐ hl

☐ facet

☐ spatial

☐ spellcheck

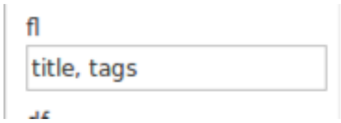
Execute Query

http://localhost:8983/solr/killrvideo.videos/select?q=avg_rating%3A%7B9+TO+*%5D&fl

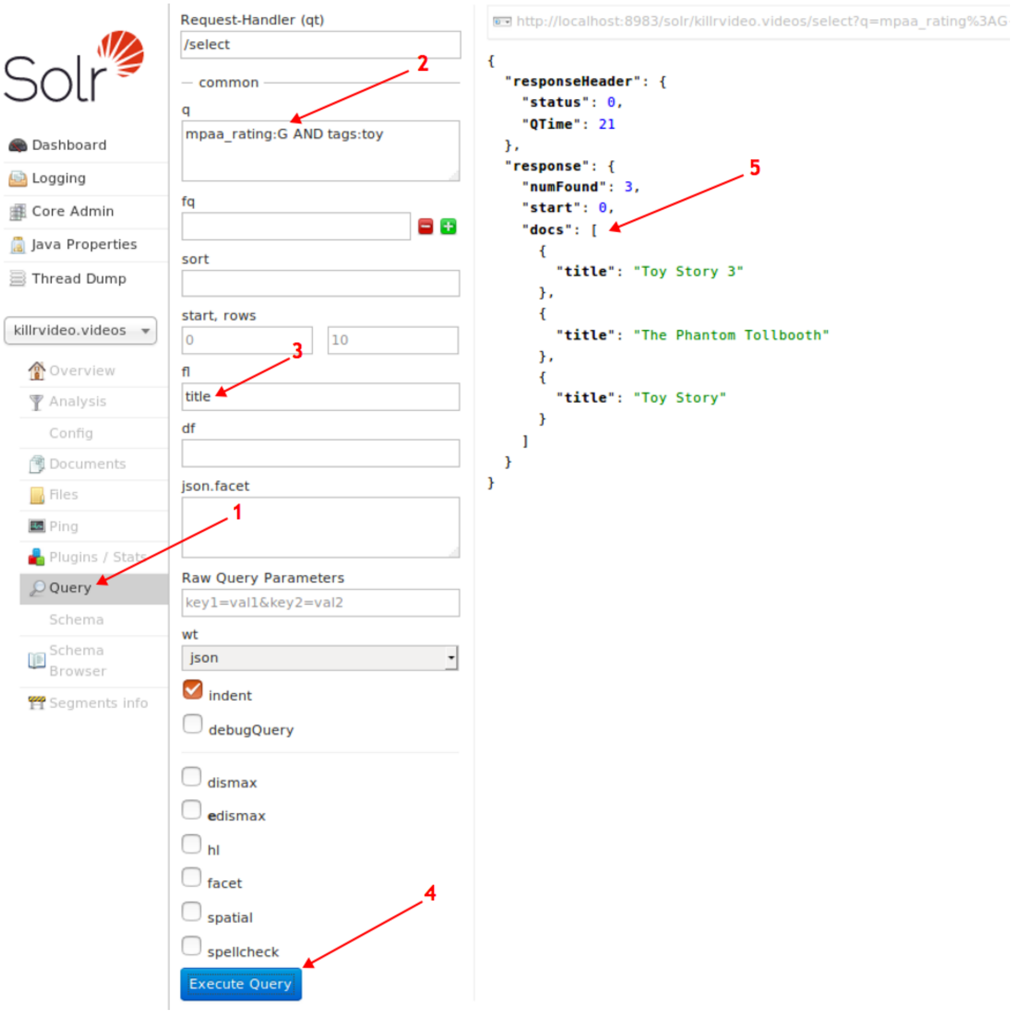
```
{
  "responseHeader": {
    "status": 0,
    "QTime": 24
  },
  "response": {
    "numFound": 32,
    "start": 0,
    "docs": [
      {
        "title": "J.R.R. Tolkien's The Hobbit"
      },
      {
        "title": "Murder, She Baked: A Chocolate Chip Cookie Mystery"
      },
      {
        "title": "L'Appel"
      },
      {
        "title": "Last Warning"
      },
      {
        "title": "Keep in Touch"
      },
      {
        "title": "Paranoia"
      },
      {
        "title": "Home Invader"
      },
      {
        "title": "Be My Cat: A Film for Anne"
      },
      {
        "title": "A Year in Space"
      },
      {
        "title": "The Spoon"
      }
    ]
  }
}
```

The last query we will execute will be a multiple filter query that will return the title for all film with a G rating and with a tag equal to "toy".

1. Select `Query` from the left-side menu list.
2. Enter in the textbox below *q: mpaa_rating:G AND tags:toy.*
3. Enter in the textbox below *fl: title.*
4. Select `Execute Query`.
5. This will show the rated G movies that have been tagged with the word *toy.*
6. To see the tags, enter “title, tags” in the fl text box and press Execute Query.



fl: title, tags



Request-Handler (qt): /select

q: mpaa_rating:G AND tags:toy

start, rows: 0, 10

fl: title

wt: json

☑ indent

http://localhost:8983/solr/killrvideo.videos/select?q=mpaa_rating%3AG

```
{
  "responseHeader": {
    "status": 0,
    "QTime": 21
  },
  "response": {
    "numFound": 3,
    "start": 0,
    "docs": [
      {
        "title": "Toy Story 3"
      },
      {
        "title": "The Phantom Tollbooth"
      },
      {
        "title": "Toy Story"
      }
    ]
  }
}
```