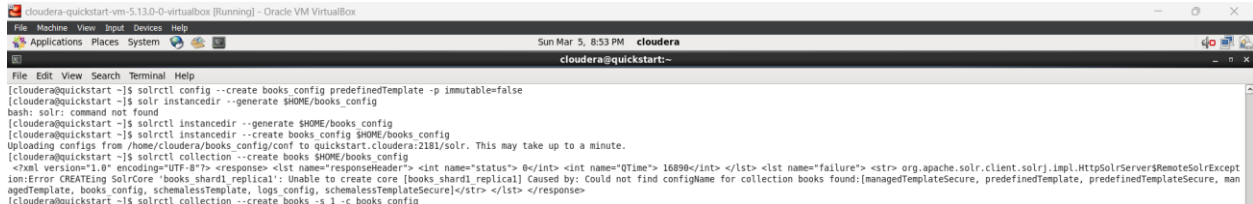


Assignment 5

Big Data and Data Science

Task 1: Dealing Book Data

Creating the configs and collection



```
cloudera-quickstart-vm:5.13.0-0-virtualbox [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Sun Mar 5, 8:53 PM cloudera
cloudera@quickstart:~
File Edit View Search Terminal Help
[cloudera@quickstart ~]$ solrctl config --create books.config predefinedTemplate -p immutable=false
[cloudera@quickstart ~]$ solrctl instanceDir --generate $HOME/books.config
bash: solr: command not found
[cloudera@quickstart ~]$ solrctl instanceDir --generate $HOME/books.config
[cloudera@quickstart ~]$ solrctl instanceDir --create books.config $HOME/books.config
Uploading configs from /home/cloudera/books.config/conf to quickstart.cloudera:2181/solr. This may take up to a minute.
[cloudera@quickstart ~]$ solrctl collection --create books $HOME/books.config
<?xml version="1.0" encoding="UTF-8"?><response><list name="responseHeader"><int name="status"> 0</int><int name="QTime"> 16890</int></list><list name="failure"><str> org.apache.solr.client.solrj.impl.HttpSolrServer$RemoteSolrExcept
ion:Error CREATEing SolrCore 'books shard1 replica1': Unable to create core [books shard1 replica1] Caused by: Could not find configName for collection books found:[managedTemplateSecure, predefinedTemplate, predefinedTemplateSecure, man
agedTemplate, books.config, schemalessTemplate, logs.config, schemalessTemplateSecure]</str></list></response>
[cloudera@quickstart ~]$ solrctl collection --create books -s 1 -c books.config
```

There are a series of steps involved before creating the collection and loading the data. First create a config file for books by the name of books_config and then create a path for the config file by using generate command followed by path. Now we are good to create a collection. Now create a collection with name books using create command as shown in the above figure. All the commands in the above figure needs to be executed in the same order.

Schema.xml for books data set

```
<!-- Common metadata fields, named specifically to match up with
SolrCell metadata when parsing rich documents such as Word, PDF.
Some fields are multiValued only because Tika currently may return
multiple values for them. Some metadata is parsed from the documents,
but there are some which come from the client context:
"content_type": From the HTTP headers of incoming stream
"resourceName": From SolrCell request param resource.name
-->
<field name="title" type="text_general" indexed="true" stored="true" multiValued="true"/>
<field name="subject" type="text_general" indexed="true" stored="true"/>
<field name="description" type="text_general" indexed="true" stored="true"/>
<field name="comments" type="text_general" indexed="true" stored="true"/>
<field name="author" type="text_general" indexed="true" stored="true"/>
<field name="keywords" type="text_general" indexed="true" stored="true"/>
<field name="category" type="text_general" indexed="true" stored="true"/>
<field name="resourceName" type="text_general" indexed="true" stored="true"/>
<field name="url" type="text_general" indexed="true" stored="true"/>
<field name="content_type" type="string" indexed="true" stored="true" multiValued="true"/>
<field name="last_modified" type="date" indexed="true" stored="true"/>
<field name="Links" type="string" indexed="true" stored="true" multiValued="true"/>

<!-- Main body of document extracted by SolrCell.
NOTE: This field is not indexed by default, since it is also copied to "text"
using copyField below. This is to save space. Use this field for returning and
highlighting document content. Use the "text" field to search the content. -->
<field name="content" type="text_general" indexed="false" stored="true" multiValued="true"/>

<!-- catchall field, containing all other searchable text fields (implemented
via copyField further on in this schema -->
<field name="text" type="text_general" indexed="true" stored="false" multiValued="true"/>

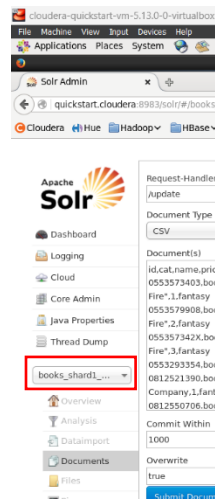
<!-- catchall text field that indexes tokens both normally and in reverse for efficient
leading wildcard queries. -->
<field name="text_rev" type="text_general_rev" indexed="true" stored="false" multiValued="true"/>

<!-- non-tokenized version of manufacturer to make it easier to sort or group
results by manufacturer. copied from "manu" via copyField -->
<field name="manu_exact" type="string" indexed="true" stored="false"/>

<field name="payloads" type="payloads" indexed="true" stored="true"/>

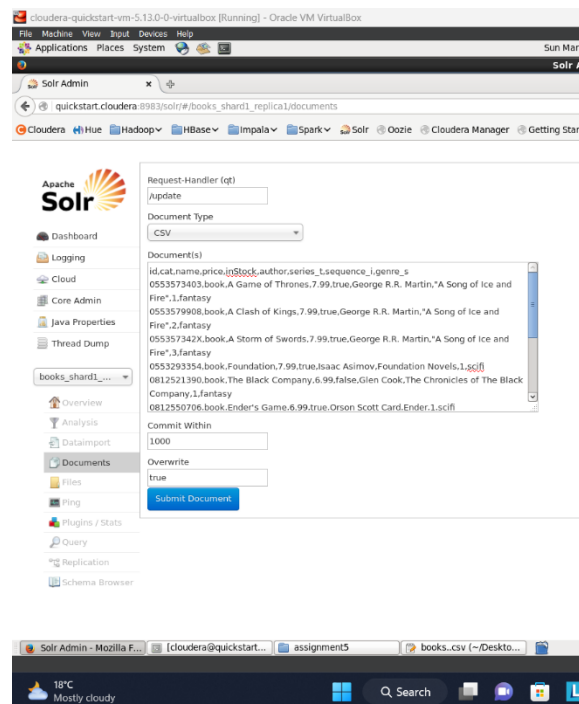
<field name="_version_" type="long" indexed="true" stored="true"/>
```

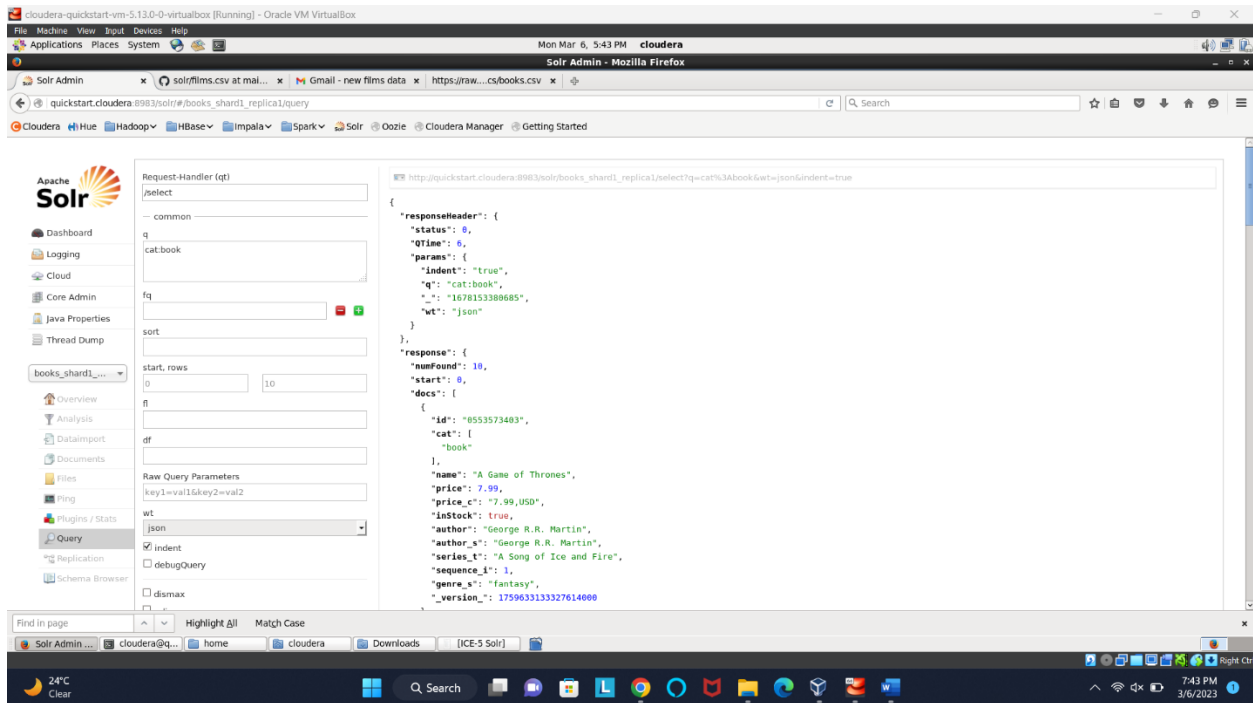
Then open solr in browser in cloudera. Solr will run on localhost:8983 by default. On successful execution of above commands a new core can be seen in the side navigation bar.



Click on document under the core and add the csv data that is obtained from the github link in the canvas assignment document.

Select the data type as csv and click on submit document and the will get added successfully.

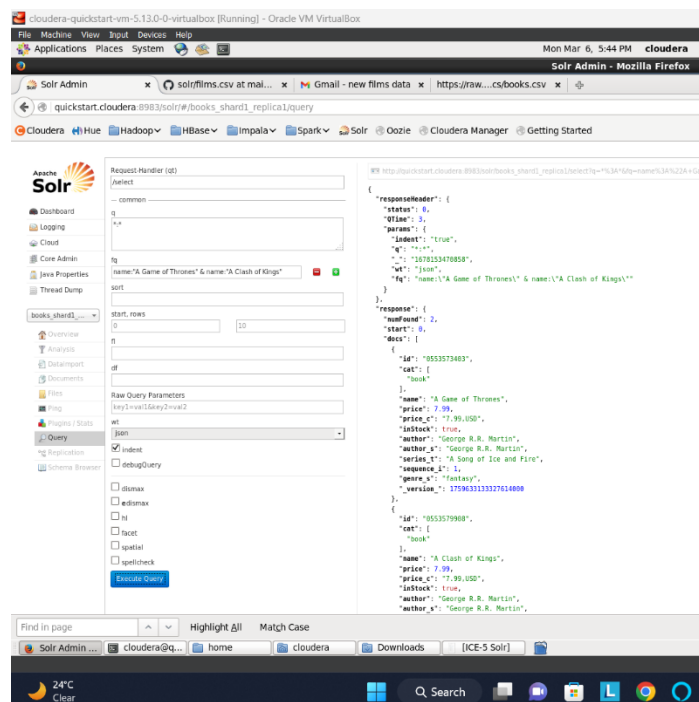




A: Display all the books with its attributes

With cat:book in q, on hit of the execute query button, the result of the query is available in json format on the right side of the screen.

B: To obtain the data whose book name is game of thrones and a clash of kings with their author's name.



Query with the name of the movie separated by and symbol and name of the other movie in fq text box. Then click on execute query. The result is the data which has the name of the movie specified in fq column.

C: Get all the books which has series_t as chronicle of prydain with their genre.

The screenshot shows the Solr Admin web interface in a browser. The left sidebar contains navigation links: Dashboard, Logging, Cloud, Core Admin, Java Properties, Thread Dump, books shard1, Overview, Analytics, Dataimport, Documents, Files, Ping, Plugins / Stats, Query (selected), and Schema Browser. The main content area is titled 'Request Handler (dt)' and shows a query execution for the 'books' core. The query is: `select * from books where series_t='The Chronicles of Prydain'`. The 'fq' (filter query) is set to `series_t:'The Chronicles of Prydain'`. The 'wt' (wrapper type) is set to 'json'. The 'indent' checkbox is checked. The 'debugQuery' checkbox is unchecked. The 'start' is 0 and the 'rows' is 10. The 'Raw Query Parameters' section shows `key3=val1&key2=val2`. The 'JSON' button is highlighted. The right pane displays the JSON response, which is a list of books with their names, series, and genres.

```
{
  "responseHeader": {
    "status": 0,
    "QTime": 1,
    "params": {
      "fq": "series_t:'The Chronicles of Prydain'",
      "wt": "json",
      "indent": "true",
      "q": "*",
      "start": 0,
      "rows": 10
    }
  },
  "response": {
    "numFound": 2,
    "start": 0,
    "docs": [
      {
        "name": "The Book of Three",
        "series_t": "The Chronicles of Prydain",
        "genre_s": "fantasy"
      },
      {
        "name": "The Black Cauldron",
        "series_t": "The Chronicles of Prydain",
        "genre_s": "fantasy"
      }
    ]
  }
}
```

Add the condition in fq text box with series_t:<name> and add the required attributes in the output to fl column. On click of the execute button the result in json will be available in the right side of the screen.

D: get all the records which has sequence greater than 1.

The screenshot shows the Solr Admin web interface in a browser. The left sidebar contains navigation links like Dashboard, Logging, Cloud, Core Admin, Java Properties, Thread Dump, and books_shard1. The main area is divided into two panels. The left panel, titled 'Request Handler (st)', shows the query 'sequence_i:[2 TO *]' and the 'Execute Query' button. The right panel, titled 'JSON', displays the response in JSON format. The response is a JSON object with a 'responseHeader' and a 'response' array. The 'response' array contains two book records.

```
{
  "responseHeader": {
    "status": 0,
    "time": 3,
    "params": {
      "indent": "true",
      "q": "*",
      "wt": "json",
      "fq": "sequence_i:[2 TO *]"
    }
  },
  "response": [
    {
      "id": "9553579908",
      "cat": [
        "book"
      ],
      "name": "A Clash of Kings",
      "price": 7.99,
      "price_d": "77.99 USD",
      "inStock": true,
      "author": "George R.R. Martin",
      "series_i": 1,
      "genre": "Fantasy",
      "version": "375963113387362808"
    },
    {
      "id": "955357342C",
      "cat": [
        "book"
      ],
      "name": "A Storm of Swords",
      "price": 7.99,
      "price_d": "77.99 USD",
      "inStock": true,
      "author": "George R.R. Martin",
      "series_i": 2
    }
  ]
}
```

To achieve this task, the condition for sequence greater than 1 can be written in fq as sequence:[2 TO *]. Here 2 is inclusive and * means the maximum that is available in the data. On click of the execute button the data whose sequence is greater than 1 is available on the screen.

E: finding books of fantasy genre.

This can be achieved by adding the condition genre=Fantasy in the fq column. The result can be limited to few attributes by adding the names of desired attributes in the fl field. On execution the query the result will be available on the screen with the books that are of genre fantasy.

The screenshot displays the Solr Admin web interface within a Mozilla Firefox browser. The browser's address bar shows the URL: `http://quickstart.cloudera:8983/solr/#/books_shard1_replica1/query`. The Solr Admin interface has a left sidebar with navigation links like Dashboard, Logging, Cloud, Core Admin, and Query. The main area is divided into two panels. The left panel contains the query editor with fields for 'Request/Handler (q)', 'fq' (set to 'genre_s:"fantasy"'), and 'fl' (set to 'name, genre_s'). The 'Execute Query' button is visible at the bottom of this panel. The right panel displays the JSON response of the query, which lists several books of the fantasy genre, including 'A Game of Thrones', 'A Clash of Kings', 'A Storm of Swords', 'The Black Company', 'The Hobbit', 'The Silmarillion', 'The Hobbit', and 'The Book of Three'.

Task 2

Creating config file and collection.

```
cloudera-quickstart-vm-5.13.0-0-virtualbox [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Applications Places System
cloudera@quickstart:~
File Edit View Search Terminal Help
[cloudera@quickstart ~]$ solrctl config --create books.config predefinedTemplate -p immutable=false
[cloudera@quickstart ~]$ solr instantiater --generate $HOME/books.config
bash: solr: command not found
[cloudera@quickstart ~]$ solrctl instantiater --generate $HOME/books.config
[cloudera@quickstart ~]$ solrctl instantiater --create books.config $HOME/books.config
Uploading configs from /home/cloudera/books.config/conf to quickstart.cloudera:2181/solr. This may take up to a minute.
[cloudera@quickstart ~]$ solrctl collection --create books $HOME/books.config
<?xml version="1.0" encoding="UTF-8"?><response><lst name="responseHeader"><int name="status">0</int><int name="QTime">16890</int></lst><lst name="failure"><str>org.apache.solr.client.solrj.impl.HttpSolrServer$RemoteSolrException: Error CREATING SolrCore 'books shard1 replica': Unable to create core [books shard1 replica] Caused by: Could not find configFile for collection books found:[managedTemplateSecure, predefinedTemplate, predefinedTemplateSecure, managedTemplate, books.config, schemasTemplate, logs.config, schemasTemplateSecure]</str></lst></response>
[cloudera@quickstart ~]$ solrctl collection --create books -s 1 -c books.config
[cloudera@quickstart ~]$ solrctl config --create films.config predefinedTemplate -p immutable=false
[cloudera@quickstart ~]$ solrctl instantiater --generate $HOME/films.config
[cloudera@quickstart ~]$ solrctl instantiater --create films.config $HOME/films.config
Uploading configs from /home/cloudera/films.config/conf to quickstart.cloudera:2181/solr. This may take up to a minute.
[cloudera@quickstart ~]$ solrctl collection --create films -s 1 -c films.config
[cloudera@quickstart ~]$
```

This is similar to the creation of books collection. Create the config and collections by executing the commands shown in the above figure.

Schema.xml for films dataset

```
<!-- Common metadata fields, named specifically to match up with
SolrCell metadata when parsing rich documents such as Word, PDF.
Some fields are multiValued only because Tika currently may return
multiple values for them. Some metadata is parsed from the documents,
but there are some which come from the client context:
"content_type": From the HTTP headers of incoming stream
"resourceName": From SolrCell request param resource.name
-->
<field name="title" type="text_general" indexed="true" stored="true" multiValued="true"/>
<field name="subject" type="text_general" indexed="true" stored="true"/>
<field name="description" type="text_general" indexed="true" stored="true"/>
<field name="comments" type="text_general" indexed="true" stored="true"/>
<field name="author" type="text_general" indexed="true" stored="true"/>
<field name="keywords" type="text_general" indexed="true" stored="true"/>
<field name="category" type="text_general" indexed="true" stored="true"/>
<field name="resourceName" type="text_general" indexed="true" stored="true"/>
<field name="url" type="text_general" indexed="true" stored="true"/>
<field name="content_type" type="string" indexed="true" stored="true" multiValued="true"/>
<field name="last_modified" type="date" indexed="true" stored="true"/>
<field name="links" type="string" indexed="true" stored="true" multiValued="true"/>

<!-- Main body of document extracted by SolrCell.
NOTE: This field is not indexed by default, since it is also copied to "text"
using copyField below. This is to save space. Use this field for returning and
highlighting document content. Use the "text" field to search the content. -->
<field name="content" type="text_general" indexed="false" stored="true" multiValued="true"/>

<!-- catchall field, containing all other searchable text fields (implemented
via copyField further on in this schema -->
<field name="text" type="text_general" indexed="true" stored="false" multiValued="true"/>

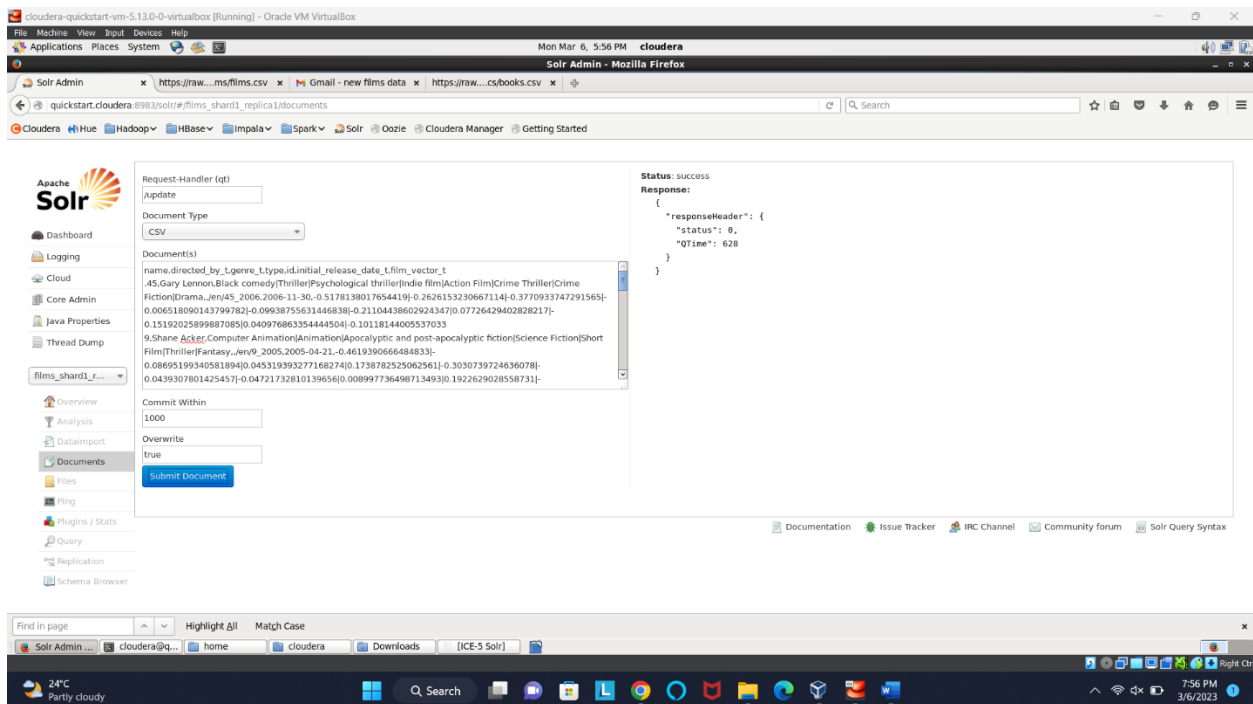
<!-- catchall text field that indexes tokens both normally and in reverse for efficient
leading wildcard queries. -->
<field name="text_rev" type="text_general_rev" indexed="true" stored="false" multiValued="true"/>

<!-- non-tokenized version of manufacturer to make it easier to sort or group
results by manufacturer. copied from "manu" via copyField -->
<field name="manu_exact" type="string" indexed="true" stored="false"/>

<field name="payloads" type="payloads" indexed="true" stored="true"/>

<field name="_version_" type="long" indexed="true" stored="true"/>
```

Then, open solr in the browser and the new collection will be available to use. Select the films core and click on documents and add the csv file in the document field. Increase the commit width to 10000 as the file is huge it will take some extra time.



A: Fetch all the movies with the attributes initial release date, genre and directed by.

Add the desired column names in the fl column click on the execute query button. The result in JSON format is available on the screen with the specified fields only.

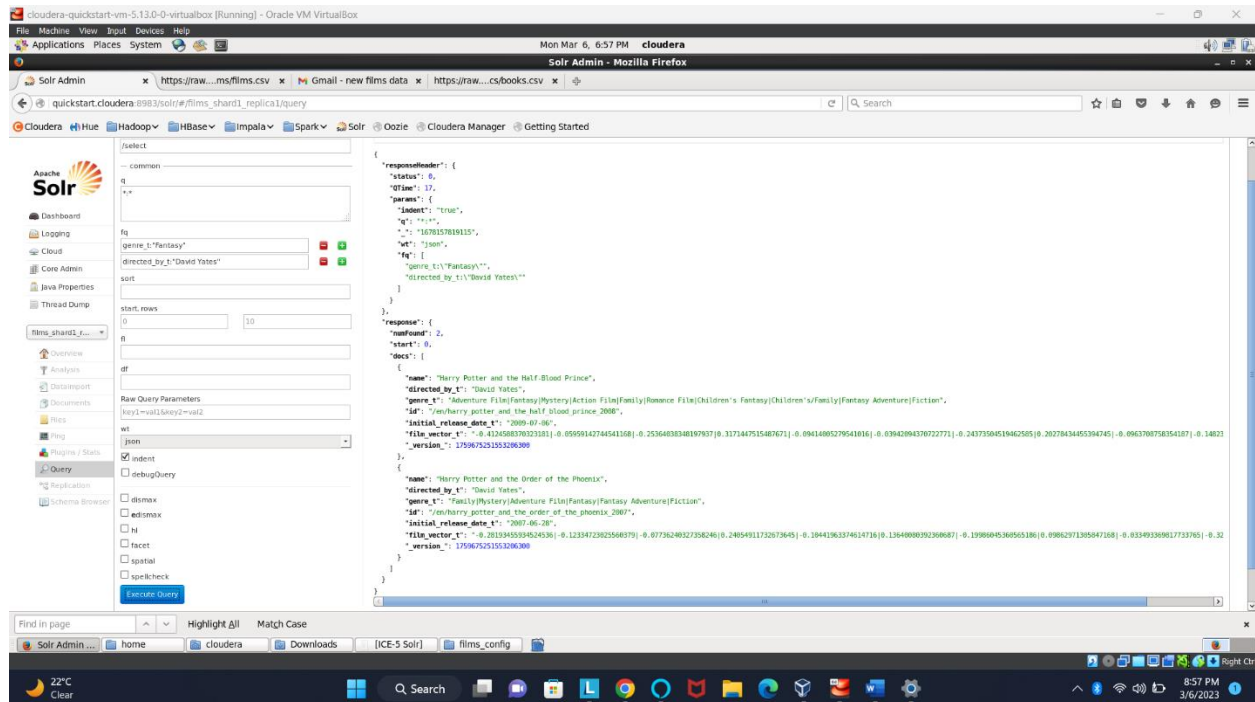
The screenshot shows the Solr Admin web interface in a browser. The left sidebar contains navigation links like Dashboard, Logging, Cloud, Core Admin, Java Properties, Thread Dump, Overview, Analysis, Dataimport, Documents, Files, Ping, Plugins / Stats, Query, Replication, and Schema Browser. The main area is divided into two panels. The left panel, titled 'Request-Handler (qt)', shows the 'select' handler with various parameters: 'q' is empty, 'fq' is 'directed_by_t:"Zack Snyder"', 'start' is 0, and 'rows' is 10. The right panel displays the JSON response from the query. The response includes a 'responseHeader' with status 0 and a 'response' object containing a list of documents. The first document is for the movie 'The Dark Knight' directed by 'Zack Snyder'.

B) Obtain all the films which are directed by Zack Snyder

This screenshot is similar to the one above, showing the Solr Admin interface. The 'fq' parameter in the 'Request-Handler (qt)' panel is set to 'directed_by_t:"Zack Snyder"'. The JSON response in the right panel shows a list of documents, including 'The Dark Knight' and 'The Dark Knight Rises', both directed by 'Zack Snyder'.

This can be obtained by adding the condition for directed_by_t column with the name of the director Zack Snyder.

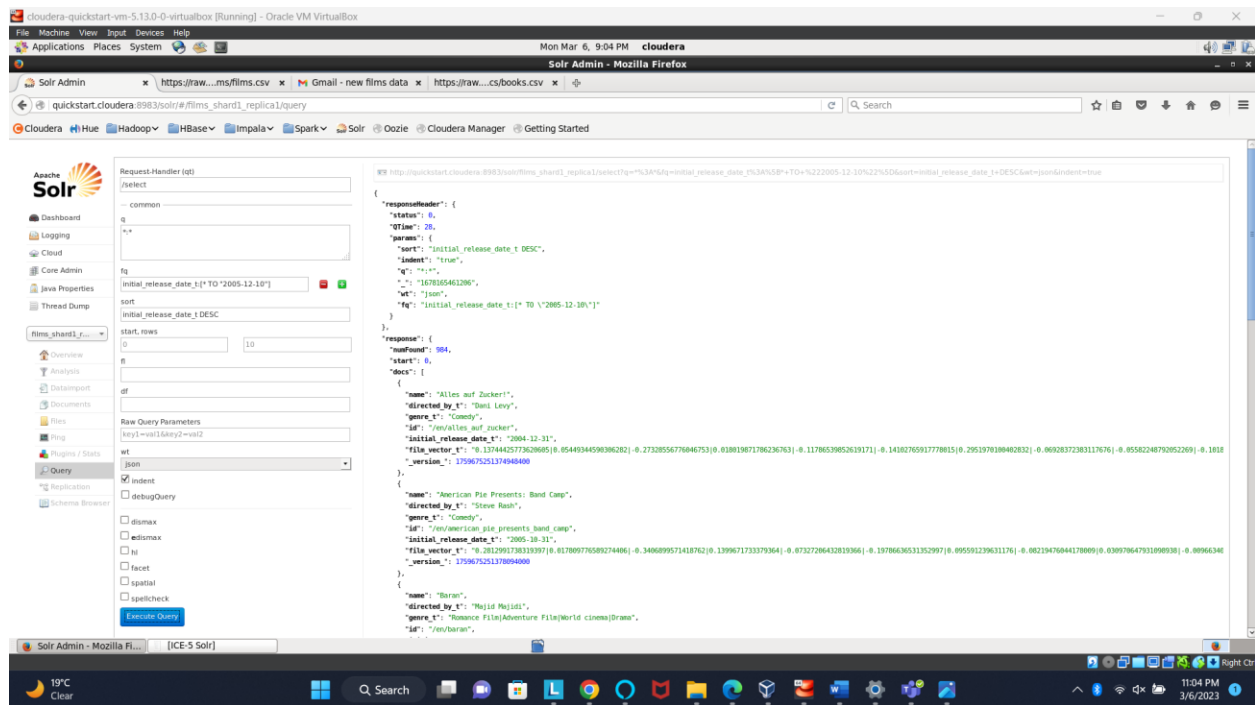
C) get all the films whose director is David Yates and the films are of genre fantasy.



Add the condition genre as fantasy in fq field and click on the plus icon to add a new condition. Add the new condition as directed by David yates in the newly added fq field. Execute the query and the result will be right on the screen.

D) Obtain all the films which have the release date before December 10th 2005 and sort the date of release in descending order.

Add the condition in the fq filed in the solr page and add the value initial_release_date_t desc in the sort field.



E) Give all the directors whose films were released in the year 2006.

Added the condition in fq as initial release data is 2006*. Which means all the data whose initial release data is beginning with 2006. Limited the field in the output by adding the necessary attribute values in the fl field.

cloudera-quickstart-vm-5.13.0-0-virtualbox [Running] - Oracle VM VirtualBox

File Machine View Input Devices Help

Applications Places System

Mon Mar 6, 6:59 PM cloudera

Solr Admin - Mozilla Firefox

Solr Admin x https://raw...ms/films.csv x Gmail - new films data x https://raw...cs/books.csv

quickstart.cloudera:8983/solr/#/films_shard1_replica1/query

Cloudera Hue Hadoop HBase Impala Spark Solr Oozie Cloudera Manager Getting Started

Apache Solr

Dashboard
Logging
Cloud
Core Admin
Java Properties
Thread Dump
films_shard1_replica1
Overview
Analysis
Dataimport
Documents
Files
Hing
Hue / Stats
Query
Hing Application
Schema Browser

Query

select

common

q

q

fq

initial_release_date:[2006*

sort

start_rows

0 10

n

directed_by:1,initial_release_date:1

df

Raw Query Parameters

key1=val1&key2=val2

wt

json

indent

debugQuery

dismax

edismax

fq

facet

spatial

spellcheck

Execute Query

```
{
  "responseHeader": {
    "status": 0,
    "QTime": 2,
    "params": {
      "q": "directed_by:1,initial_release_date:1",
      "indent": "true",
      "wt": "json",
      "fq": "initial_release_date:[2006*"
    }
  },
  "response": {
    "numFound": 173,
    "start": 0,
    "docs": [
      {
        "directed_by_1": "Gary Lennon",
        "initial_release_date_t": "2006-11-30"
      },
      {
        "directed_by_1": "Jack Snyder",
        "initial_release_date_t": "2006-12-09"
      },
      {
        "directed_by_1": "John LaFia",
        "initial_release_date_t": "2006-03-18"
      },
      {
        "directed_by_1": "Robert Florence",
        "initial_release_date_t": "2006-08-18"
      },
      {
        "directed_by_1": "Richard Donner",
        "initial_release_date_t": "2006-03-01"
      },
      {
        "directed_by_1": "Thom Fitzgerald",
        "initial_release_date_t": "2006-12-01"
      }
    ]
  }
}
```

Find in page Highlight All Match Case

Solr Admin - M... home cloudera Downloads [ICE-5 Solr]

22°C Clear

Search

8:59 PM 3/6/2023