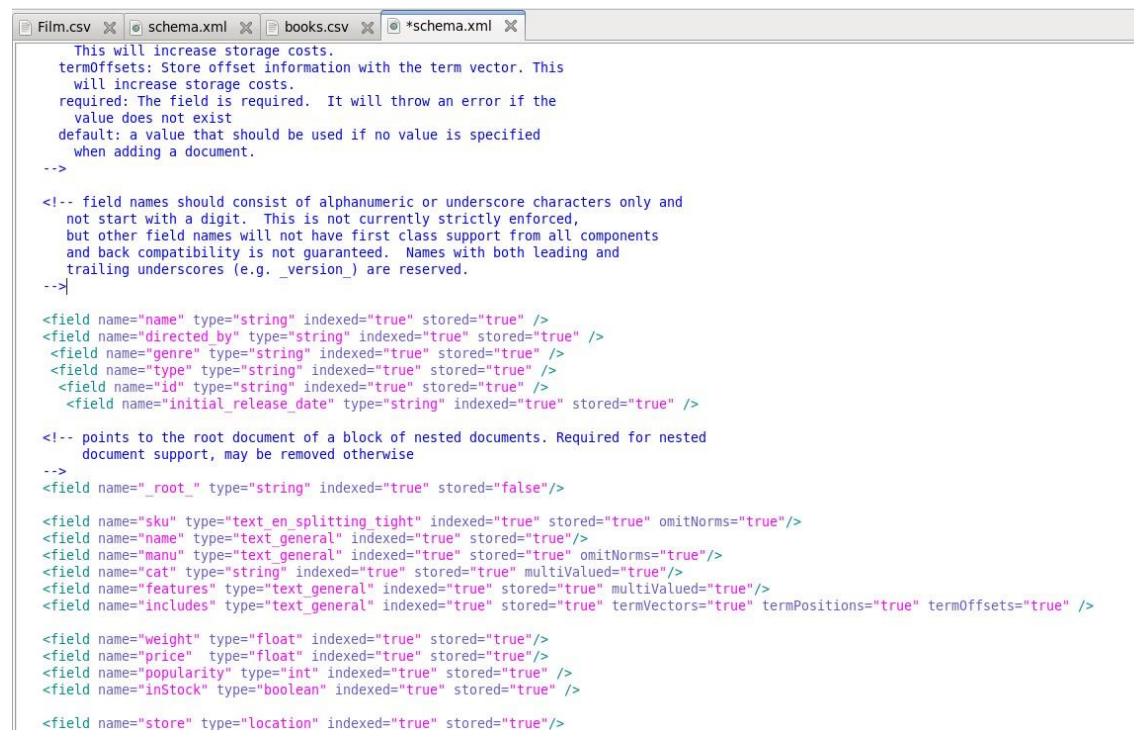**Films Dataset**

Creating collection

```
[cloudera@quickstart ~]$ solrctl collection --list
logNew2 (2)
books (2)
Books (2)
filmss (2)
[cloudera@quickstart ~]$ solrctl config --create log_config predefinedTemplate -p immutable=false
[cloudera@quickstart ~]$ solrctl instancedir --generate $HOME/log_config
[cloudera@quickstart ~]$ solrctl collection --create lognew3 -c log_config
[cloudera@quickstart ~]$ solrctl collection --list
logNew2 (2)
books (2)
Books (2)
lognew3 (2)
filmss (2)
[cloudera@quickstart ~]$ solrctl instancedir --generate /tmp/Films
[cloudera@quickstart ~]$ ls /tmp/Films/conf/
admin-extra.html        clustering    lang                    protwords.txt               _schema_analysis_synonyms_english.json solrconfig.xml        stopwords.txt   velocity
admin-extra.menu-bottom.html  currency.xml  mapping-FoldToASCII.txt  _rest_managed.json          schema.xml                  solrconfig.xml.secure synonyms.txt   xslt
admin-extra.menu-top.html    elevate.xml   mapping-ISOLatin1Accent.txt _schema_analysis_stopwords_english.json scripts.conf                spellings.txt         update-script.js
[cloudera@quickstart ~]$ gedit /tmp/Films/conf/schema.xml
[cloudera@quickstart ~]$ ▮
```

- Firstly we list the collections using –list command.
- Then we create temporary config named log_config based on existing instance directory predefinedTemplate
- Then we created collection lognew3 to store films data based on config log_config.
- Then we create schema configuration using instancedir –generate command.
- Then we list the files in Films configuration using ls command.
- Then we edit the schema config using gedit command

Editing Schema

```
Film.csv    schema.xml    books.csv    *schema.xml
      This will increase storage costs.
      termOffsets: Store offset information with the term vector. This
        will increase storage costs.
      required: The field is required.  It will throw an error if the
        value does not exist
      default: a value that should be used if no value is specified
        when adding a document.
    -->

    <!-- field names should consist of alphanumeric or underscore characters only and
       not start with a digit.  This is not currently strictly enforced,
       but other field names will not have first class support from all components
       and back compatibility is not guaranteed.  Names with both leading and
       trailing underscores (e.g. _version_) are reserved.
    -->

    <field name="name" type="string" indexed="true" stored="true" />
    <field name="directed_by" type="string" indexed="true" stored="true" />
    <field name="genre" type="string" indexed="true" stored="true" />
    <field name="type" type="string" indexed="true" stored="true" />
    <field name="id" type="string" indexed="true" stored="true" />
    <field name="initial_release_date" type="string" indexed="true" stored="true" />

    <!-- points to the root document of a block of nested documents. Required for nested
         document support, may be removed otherwise
    -->
    <field name="_root_" type="string" indexed="true" stored="false"/>

    <field name="sku" type="text_en_splitting_tight" indexed="true" stored="true" omitNorms="true"/>
    <field name="name" type="text_general" indexed="true" stored="true"/>
    <field name="manu" type="text_general" indexed="true" stored="true" omitNorms="true"/>
    <field name="cat" type="string" indexed="true" stored="true" multiValued="true"/>
    <field name="features" type="text_general" indexed="true" stored="true" multiValued="true"/>
    <field name="includes" type="text_general" indexed="true" stored="true" termVectors="true" termPositions="true" termOffsets="true" />

    <field name="weight" type="float" indexed="true" stored="true"/>
    <field name="price"  type="float" indexed="true" stored="true"/>
    <field name="popularity" type="int" indexed="true" stored="true" />
    <field name="inStock" type="boolean" indexed="true" stored="true" />

    <field name="store" type="location" indexed="true" stored="true"/>
```

- Here we create new fields according to the column names in Film.csv file.

- We create fields like name, directed_by, genre, type, id, and initial_release_date and specify their types.

Loading Data

- Here we load the csv file data into solr in the documents section and we get the status as success.

Queries

1) List all rows where films are directed by lee sang il.

- Here we used select command and specified condition as films directed by lee sang il and displayed the resulting rows.

2) List all rows where films are released in the year 2008.

- Here we used select command and specified condition as films released in 2008 and displayed the resulting rows.

3) List all rows where the genre of film is disaster.

- Here we used select command and specified condition as movie genre as disaster and displayed the resulting rows.

4) List all rows where the films are released in the year 2010.

- Here we used select command and specified condition as films released in year 2008 and displayed the resulting rows.

5) List all rows where the movies are directed by director Danny Boyle.

- Here we used select command and specified condition as director as Danny Boyle and displayed the resulting rows.

Books Dataset

Creating collection

```
[cloudera@quickstart ~]$ solrctl collection --create BooKs -c log_config
[cloudera@quickstart ~]$ solrctl instancedir --generate /tmp/BooKs
[cloudera@quickstart ~]$ ls /tmp/BooKs/conf
admin-extra.html          clustering   lang                      protwords.txt            _schema_analysis_synonyms_english.json solrconfig.xml        stopwords.txt     velocity
admin-extra.menu-bottom.html currency.xml mapping-FoldToASCII.txt  _rest_managed.json      schema.xml                             solrconfig.xml.secure synonyms.txt      xslt
admin-extra.menu-top.html   elevate.xml  mapping-ISOLatin1Accent.txt _schema_analysis_stopwords_english.json scripts.conf        spellings.txt         update-script.js
[cloudera@quickstart ~]$ gedit /tmp/BooKs/conf/schema.xml
[cloudera@quickstart ~]$ █
```

- Here we create collection named Books
- Then we create schema configuration using instancedir –generate command.
- Then we list the files in Books configuration using ls command.
- Then we edit the schema config using gedit command

Editing Schema

```
Film.csv    schema.xml    books.csv

    default: a value that should be used if no value is specified
      when adding a document.
-->

<!-- field names should consist of alphanumeric or underscore characters only and
    not start with a digit.  This is not currently strictly enforced,
    but other field names will not have first class support from all components
    and back compatibility is not guaranteed.  Names with both leading and
    trailing underscores (e.g. _version_) are reserved.
-->

<field name="id" type="string" indexed="true" stored="true" required="true" multiValued="false" />
 <field name="cat" type="string" indexed="true" stored="true" required="true" multiValued="false" />
<field name="name" type="string" indexed="true" stored="true" required="true" multiValued="false" />
<field name="price" type="float" indexed="true" stored="true" required="true" multiValued="false" />
<field name="inStock" type="boolean" indexed="true" stored="true" required="true" multiValued="false" />
<field name="author" type="string" indexed="true" stored="true" required="true" multiValued="false" />
<field name="series_t" type="string" indexed="true" stored="true" required="true" multiValued="false" />
<field name="sequence_i" type="int" indexed="true" stored="true" required="true" multiValued="false" />
<field name="genre_s" type="string" indexed="true" stored="true" required="true" multiValued="false" />

<!-- points to the root document of a block of nested documents. Required for nested
     document support, may be removed otherwise
-->
<field name="_root_" type="string" indexed="true" stored="false"/>

<field name="sku" type="text_en_splitting_tight" indexed="true" stored="true" omitNorms="true"/>
<field name="name" type="text_general" indexed="true" stored="true"/>
<field name="manu" type="text_general" indexed="true" stored="true" omitNorms="true"/>
<field name="cat" type="string" indexed="true" stored="true" multiValued="true"/>
<field name="features" type="text_general" indexed="true" stored="true" multiValued="true"/>
<field name="includes" type="text_general" indexed="true" stored="true" termVectors="true" termPositions="true" termOffsets="true" />

<field name="weight" type="float" indexed="true" stored="true"/>
<field name="price"  type="float" indexed="true" stored="true"/>
<field name="popularity" type="int" indexed="true" stored="true" />
<field name="inStock" type="boolean" indexed="true" stored="true" />

<field name="store" type="location" indexed="true" stored="true"/>

<!-- Common metadata fields, named specifically to match up with
    SolrCell metadata when parsing rich documents such as Word, PDF.
```

- Here we create new fields according to the column names in books.csv file.
- We create fields like id, cat, name, price, inStock, author, series_t, sequence_i and genre_s and specify their types.

Loading The Data

- Here we load the csv file data into solr in the documents section and we get the status as success.

Queries

1. List all rows with genre as "fantasy"

- Here we have used select command and specified genre = fantasy and displayed all the rows with fantasy as genre.

2. List all the rows where instock value is true.

- Here we used select command and specified condition as inStock = true and displayed all rows where instock is true.

3. List all the rows with genre as fantasy, instock as true and price between 7 to 8

- Here we used select command and specified condition like genre as fantasy, inStock as true and price between 7 to 8. We used AND command within the conditions.

4. List all the rows where price is between 4 and 8.

- Here we used select command and specified condition as price between 4 to 8 and displayed all the resulting rows.

5. List all the rows where author is George R.R

- Here we used select command and specified condition as author = George R.R and displayed all the resulting rows.

6. List all the rows where genre is scifi.

- Here we used select command and specified genre as scifi and displayed all the resulting rows.