

Download and index facebook's posts using nifi and solr

We are going to download posts from the API <https://graph.facebook.com/> using the nifi work flow process and put them to hdfs, run MapreduceIndexerTool over all files and finally display results as dashboards using banana

Download Posts

First let's open <https://developers.facebook.com/tools/explorer/> to get our token

If we try to download all posts from "European Commission" public page we just put "107898832590939/posts" in the field and press submit.

The result should look something like this



The screenshot shows the Facebook Graph API Explorer interface. At the top, it says "Graph API Explorer" and "Application: [?] Graph API Explorer". Below that, the "Access Token" field contains a long alphanumeric string. The "GET" method is selected, and the URL is set to "/v3.0 /107898832590939/posts". A "Submit" button is visible. The response is displayed in a JSON format, showing a list of posts with fields like "created_time", "message", and "id".

The schema of the result:

```
{
  "data": [
    {
      "created_time": ,
      "message": ,
      "id":
    }
  ],
  "paging": {
    "cursors": {
      "before": ,
      "after": },
    "next":
  }
}
```

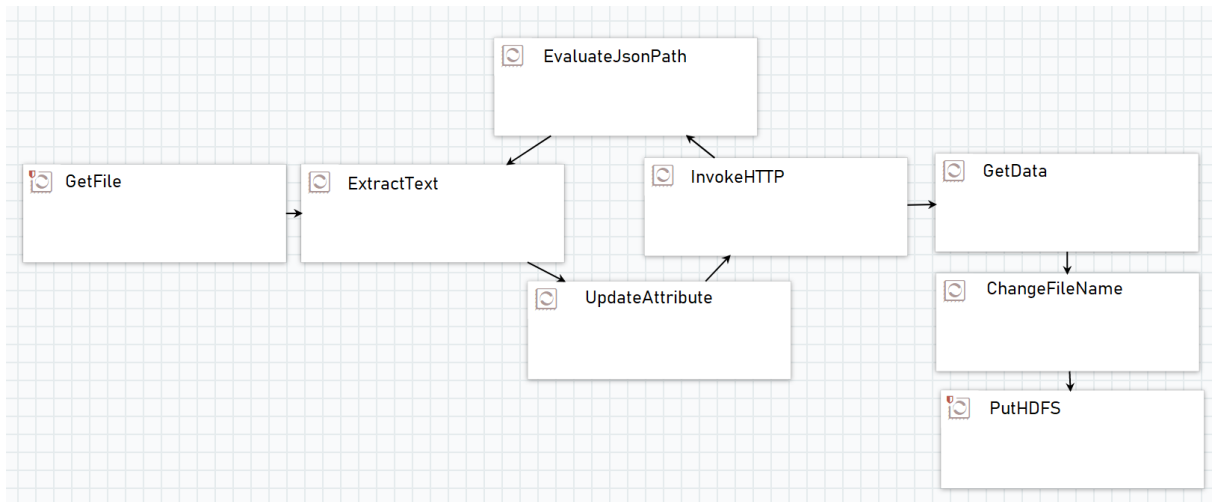
We have three fields for each post `created_time`, `message` and `id` we will improve the query to get more information (likes count, share count , comments ...)

We noticed there are a limitation of the number of posts per response, we have to follow the link in the next field to get the next portion of the response

Now we have an idea about how it's work and about our data structure, it will be easy to understand the way how nifi manage the paging between responses.

NIFI

The nifi flow



We will run through every processor to see the configurations of each. Before that let's see how the data pass across every processor

GetFile -> [ExtractText -> UpdateAttribute -> InvokeHttp -> EvaluateJsonPath] -> GetData -> ChangeFileName -> PutHDFS

Step 1

GetFile: the processor get a file from a specified directory and deleted, in this file we put the first URL

Type : Getfile

Configure Processor

SETTINGS

SCHEDULING

PROPERTIES

COMMENTS

Required field

+

Property	Value
Input Directory	/home/cloudera/Desktop/FBlink
File Filter	[^\\].*
Path Filter	No value set
Batch Size	10
Keep Source File	false
Recurse Subdirectories	true
Polling Interval	0 sec
Ignore Hidden Files	true
Minimum File Age	0 sec
Maximum File Age	No value set
Minimum File Size	0 B
Maximum File Size	No value set

CANCEL

APPLY

Step 2

ExtractText: extract text from the file flow

In the first run it get the URL from the get file processor, after it will get the next URL from the EvaluateJsonPath

Type : ExtractText

Configure Processor

SETTINGS | SCHEDULING | **PROPERTIES** | COMMENTS

Required field

Property	Value
Maximum Buffer Size	1 MB
Maximum Capture Group Length	1024
Enable Canonical Equivalence	true
Enable Case-insensitive Matching	false
Permit Whitespace and Comments in Pattern	false
Enable DOTALL Mode	false
Enable Literal Parsing of the Pattern	false
Enable Multiline Mode	false
Enable Unicode-aware Case Folding	true
Enable Unicode Predefined Character Classes	true
Enable Unix Lines Mode	false
Include Capture Group 0	true
Enable repeating capture group	false
next	(.*)

CANCEL | APPLY

Step 3

UpdateAttribute : assign the URL in the attribute "link"

Type : UpdateAttribute

Configure Processor

SETTINGS | SCHEDULING | **PROPERTIES** | COMMENTS

Required field

Property	Value
Delete Attributes Expression	No value set
Store State	Do not store state
Stateful Variables Initial Value	No value set
link	\${next}

CANCEL | APPLY

ADVANCED

Step 4

InvokeHttp: invoke the attribute "link"

We use urlDecode() to avoid encoding problem (ex : type%2Cmessage => type,message)

Type : InvokeHttp

Configure Processor

SETTINGS SCHEDULING **PROPERTIES** COMMENTS

Required field +

Property	Value
HTTP Method	GET
Remote URL	<code>\${link:urlDecode()}</code>
SSL Context Service	No value set
Connection Timeout	10 secs
Read Timeout	30 secs
Include Date Header	True
Follow Redirects	True
Attributes to Send	No value set
Basic Authentication Username	No value set
Basic Authentication Password	No value set
Proxy Host	No value set
Proxy Port	No value set
Proxy Type	http
Proxy Username	No value set

CANCEL APPLY

Step 5

EvaluateJsonPath : get only the attribute next from response

Type : EvaluateJsonPath

Configure Processor

SETTINGS SCHEDULING **PROPERTIES** COMMENTS

Required field +

Property	Value
Destination	flowfile-content
Return Type	auto-detect
Path Not Found Behavior	ignore
Null Value Representation	empty string
next	<code>\$.paging.next</code>

CANCEL APPLY

Step 6


GetData: get only the data from response

Type : EvaluateJsonPath

Configure Processor

SETTINGS | SCHEDULING | **PROPERTIES** | COMMENTS

Required field

Property		Value	
Destination	?	flowfile-content	
Return Type	?	auto-detect	
Path Not Found Behavior	?	ignore	
Null Value Representation	?	empty string	
Data	?	\$.data	

CANCEL | APPLY

Step 7


ChangeFileName: change the name of the file before saving it, here we set the name to a date time


Type : UpdateAttribute

Configure Processor

SETTINGS | SCHEDULING | **PROPERTIES** | COMMENTS

Required field

Property		Value	
Delete Attributes Expression	?	No value set	
Store State	?	Do not store state	
Stateful Variables Initial Value	?	No value set	
filename	?	\${now():format("yyyy-MM-dd-HH-mm-ss")}	

 ADVANCED

CANCEL | APPLY

Step 8

PutHDFS: Save files to hdfs

Type : PutHDFS

Configure Processor

SETTINGS

SCHEDULING

PROPERTIES

COMMENTS

Required field

Property	Value	
Hadoop Configuration Resources	/etc/hadoop/conf/core-site.xml	
Kerberos Credentials Service	No value set	
Kerberos Principal	No value set	
Kerberos Keytab	No value set	
Kerberos Relogin Period	4 hours	
Additional Classpath Resources	No value set	
Directory	/user/cloudera/facebook/	
Conflict Resolution Strategy	fail	
Block Size	No value set	
IO Buffer Size	No value set	
Replication	1	
Permissions umask	No value set	
Remote Owner	No value set	
Remote Group	No value set	

CANCEL

APPLY

Well, after all this configuration let's have a look of the first URL and how it's build.

In this example:

[https://graph.facebook.com/v3.0/107898832590939/posts?access_token=YourToken&fields=created_time,link,type,message,likes.limit\(1\).summary\(true\),description,from,name,shares,comments&limit=25](https://graph.facebook.com/v3.0/107898832590939/posts?access_token=YourToken&fields=created_time,link,type,message,likes.limit(1).summary(true),description,from,name,shares,comments&limit=25)

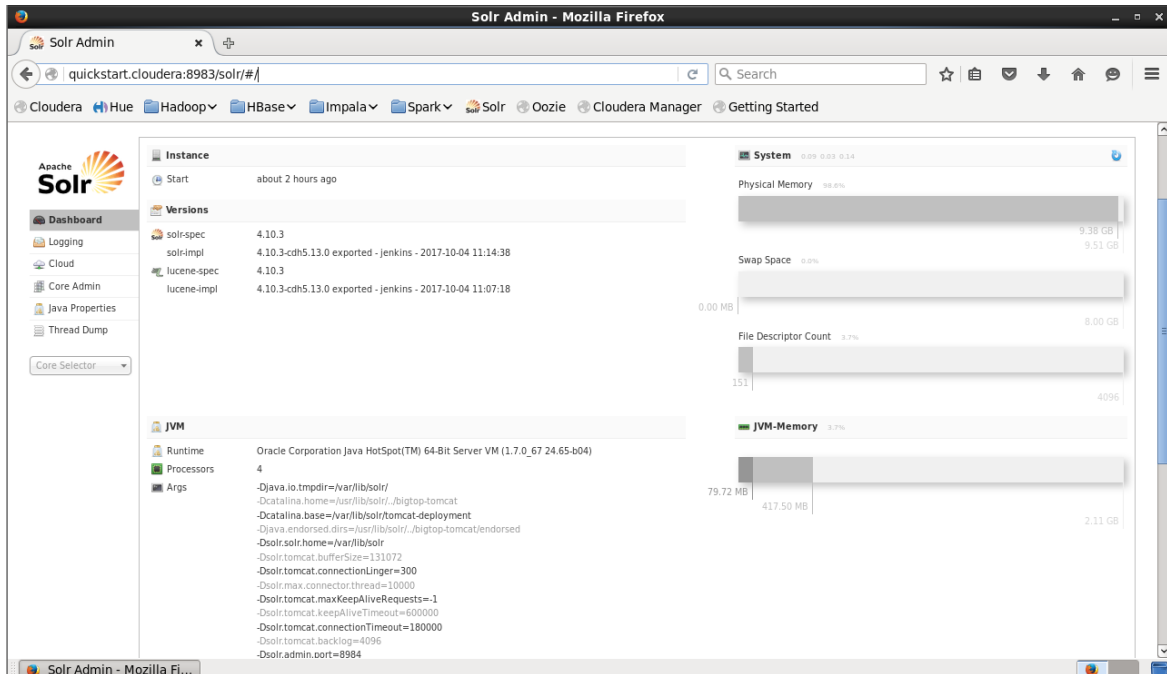
We get the following schema

```
{
  "created_time": ,
  "link": ,
  "type": ,
  "message": ,
  "likes": {
    "data": [
    ],
    "paging": {
      "cursors": {
        "before": ,
        "after":
      },
      "next":
    },
    "summary": {
      "total_count": ,
      "can_like": ,
      "has_liked":
    }
  },
  "from": {
    "name": ,
    "id":
  },
  "shares": {
    "count":
  },
  "comments": {
    "data": [{
      "created_time": ,
      "message": ,
      "id":
    }
  ],
  "paging": {
    "cursors": {
      "before": ,
      "after":
    }
  }
}
```

Solr

Now we have to configure a new solr collection and upload it to hdfs. Make sure that solr is running

go to <http://quickstart.cloudera:8983/solr/#/>



Well let's start

Open a new terminal window and go to solr directory

```
cd /usr/lib/solr/bin/
```

```
sudo su
```

Generate a new *instancedir* with *solrctl*

```
solrctl --zk quickstart.cloudera:2181/solr instancedir --generate /tmp/fb_collection/
```

Copy the schema.xml into /tmp/ fb_collection /conf/

```
cp -r /home/cloudera/Desktop/schema.xml /tmp/fb_collection/conf/
```

type yes for overwrite

```
root@quickstart:/usr/lib/solr/bin
File Edit View Search Terminal Help
[cloudera@quickstart ~]$ sudo su
[root@quickstart cloudera]# cd /usr/lib/solr/bin/
[root@quickstart bin]# solrctl --zk quickstart.cloudera:2181/solr instancedir --
generate /tmp/fb_collection/
[root@quickstart bin]# cp -r /home/cloudera/Desktop/schema.xml /tmp/fb_collectio
n/conf/
cp: overwrite `/tmp/fb_collection/conf/schema.xml'? y
[root@quickstart bin]#
```


Create an instancedir from this configuration

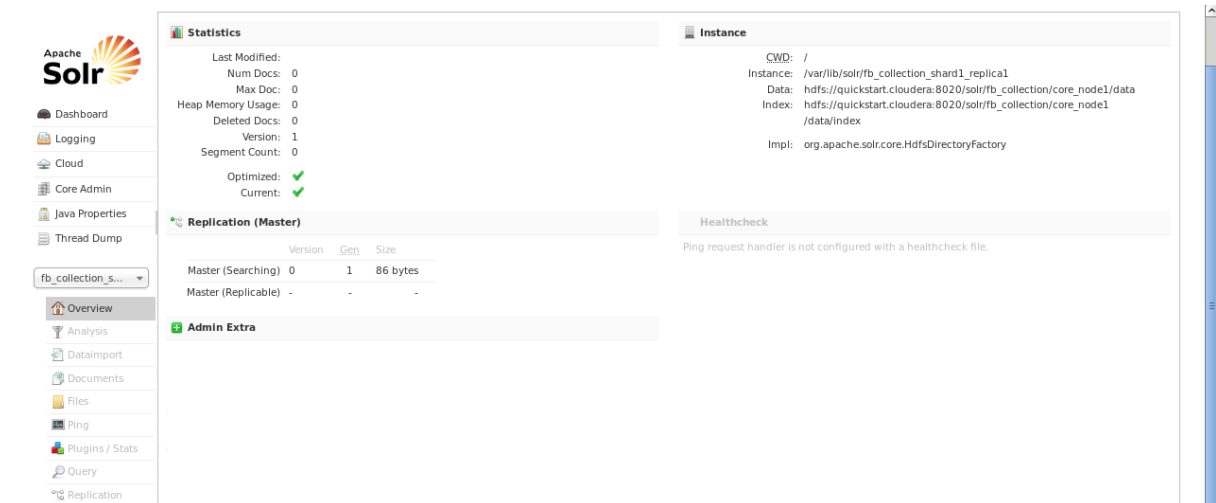
```
solrctl --zk quickstart.cloudera:2181/solr instancedir --create fb_collection /tmp/fb_collection/
```

Create the collection with the same name of the instancedir

```
solrctl --zk quickstart.cloudera:2181/solr collection --create fb_collection
```

```
root@quickstart:/usr/lib/solr/bin
File Edit View Search Terminal Help
[cloudera@quickstart ~]$ sudo su
[root@quickstart cloudera]# cd /usr/lib/solr/bin/
[root@quickstart bin]# solrctl --zk quickstart.cloudera:2181/solr instancedir --
generate /tmp/fb_collection/
[root@quickstart bin]# cp -r /home/cloudera/Desktop/schema.xml /tmp/fb_collectio
n/conf/
cp: overwrite `/tmp/fb_collection/conf/schema.xml'? y
[root@quickstart bin]# solrctl --zk quickstart.cloudera:2181/solr instancedir --
create fb_collection /tmp/fb_collection/
Uploading configs from /tmp/fb_collection/conf to quickstart.cloudera:2181/solr
. This may take up to a minute.
[root@quickstart bin]# solrctl --zk quickstart.cloudera:2181/solr collection --c
reate fb_collection
```

Go to core admin http://quickstart.cloudera:8983/solr/#/~cores/fb_collection_shard1_replica1 to see the new core running



Solr mapreduceindexertool

install mapreduceindexertool using the commands

```
sudo yum install solr-mapreduce
```

Just before, you have to change the hadoop ACLs for direct inject the result into solr,

To enable ACLs set the `dfs.namenode.acls.enabled` property to true in the NameNode's `hdfs-site.xml`.

https://www.cloudera.com/documentation/enterprise/5-8-x/topics/cdh_sg_hdfs_ext_acls.html#xd_583c10bfdbd326ba--6eed2fb8-14349d04bee--76a9

```
<property>
<name>dfs.namenode.acls.enabled</name>
<value>true</value>
</property>
```

Well Done.

Cloudera Morphlines

Cloudera Morphlines is a new open source framework that reduces the time and effort necessary to integrate, build, and change Hadoop processing applications that extract, transform, and load data into Apache Solr, Apache HBase, HDFS, enterprise data warehouses, or analytic online dashboards.

You can read more about Morphlines on the cloudera website

<https://blog.cloudera.com/blog/2013/07/morphlines-the-easy-way-to-build-and-integrate-etl-apps-for-apache-hadoop/>

FBmorphline.conf

```
# Specify server locations in a SOLR_LOCATOR variable; used later in variable substitutions:
SOLR_LOCATOR : {
  # Natschemae of solr collection
  collection : fb_collection

  # ZooKeeper ensemble
  zkHost : "127.0.0.1:2181/solr"

  # The maximum number of documents to send to Solr per network batch (throughput knob)
  # batchSize : 100
}

# Specify an array of one or more morphlines, each of which defines an ETL
# transformation chain. A morphline consists of one or more (potentially
# nested) commands. A morphline is a way to consume records (e.g. Flume events,
# HDFS files or blocks), turn them into a stream of records, and pipe the stream
# of records through a set of easily configurable transformations on it's way to
# Solr.
morphlines : [
  {
    # Name used to identify a morphline. E.g. used if there are multiple morphlines in a
    # morphline config file
    id : morphline1
    # Import all morphline commands in these java packages and their subpackages.
    # Other commands that may be present on the classpath are not visible to this morphline.
    importCommands : ["org.kitesdk.**", "org.apache.solr.**", "com.cloudera.**"]

    commands : [
      { readJson: {} }

      { extractJsonPaths {
        flatten : true # to transform arrays in real arrays (not a String representation)
        paths : {
          created_time : /created_time
          link : /link
          type : /type
          message : /message
          likes_count : /likes/summary/total count
          from_name : /from/name
          post_name : /name
          shares_count : /shares/count
          comments : /comments/data[]/message"
          id : /id
```

```

    }
  }
}

# Consume the output record of the previous command and pipe another
# record downstream.
#
# convert timestamp field to native Solr timestamp format
{
  convertTimestamp {
    field : created_time
    inputFormats : ["yyyy-MM-dd'T'HH:mm:ss'+SSSS"]
    inputTimezone : America/Los_Angeles
    outputFormat : "yyyy-MM-dd'T'HH:mm:ss.SSS'Z'"

    outputTimezone : UTC
  }
}

{
  sanitizeUnknownSolrFields {

    # Location from which to fetch Solr schema

    solrLocator : ${SOLR_LOCATOR}
    renameToPrefix : "ignored "
  }
}

# log the record at DEBUG level to SLF4J
{ logDebug { format : "output record: {}", args : ["@{}"] } }
# load the record into a Solr server or MapReduce Reducer.
{
  loadSolr {
    solrLocator : ${SOLR_LOCATOR}
  }
}
}
]

```

Put the FBmorphline.conf to /tmp/morphlines/

Run our indexer

Invoke the --help to get more details and examples

```

sudo -u hdfs hadoop jar /usr/lib/solr/contrib/mr/search-mr-*.job.jar
org.apache.solr.hadoop.MapReduceIndexerTool -help

```

Run the job

```

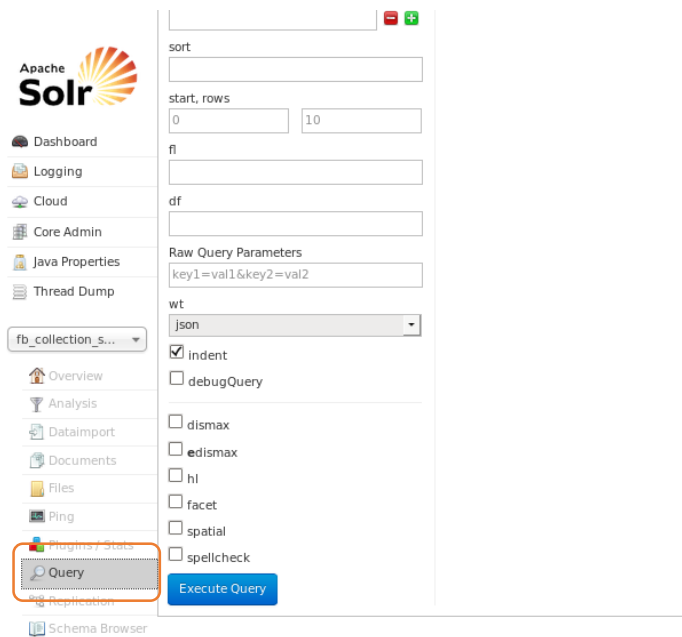
sudo -u hdfs hadoop jar /usr/lib/solr/contrib/mr/search-mr-*.job.jar
org.apache.solr.hadoop.MapReduceIndexerTool -D 'mapred.child.java.opts=-Xmx512m' --morphline-
file /tmp/morphlines/FBmorphline.conf --output-dir hdfs://quickstart.cloudera:8020/tmp/output/ --
verbose --go-live --zk-host quickstart.cloudera:2181/solr --collection fb_collection
\hdfs://quickstart.cloudera:8020/user/cloudera/facebook/

```

If the indexing was performed correctly you can select the collection and you can see the number of docs

The screenshot displays the Apache Solr Admin UI. On the left, the 'fb_collection_s...' dropdown is selected. The main panel shows the 'Statistics' tab for this collection, indicating 3375 documents and a size of 19.81 MB. Below this, the 'Replication (Master)' tab shows two Master nodes: 'Master (Searching)' and 'Master (Replicable)', both with a size of 19.81 MB. On the right, the 'Instance' tab shows the CWD as '/' and the Instance as '/var/lib/solr/fb_collection_shard1_replica1'.

We can try to see some data let's run an empty query in the query tab



The result will be like following

```
"response": {
  "numFound": 3375,
  "start": 0,
  "docs": [
    {
      "link": "https://www.facebook.com/EuropeanCommission/videos/1001306153250198/",
      "type": "video",
      "id": "107898832590939_1001306153250198",
      "from_name": "European Commission",
      "post_name": "Energy Union: Towards a smart, efficient and sustainable heati...",
      "likes_count": 85,
      "shares_count": 31,
      "message": [
        "Heating and cooling accounts for 50% of all the #energy we consume in the EU each year. Did you know that the heat being wast
      ],
      "created_time": "2016-02-18T22:47:57Z",
      "comments": [
        "Testimony of an honest lender Greeting has all Testimony of an honest lender on this Mister who returned to me happy. I am Fr
        \"What does the individual have to do\",
        \"PV ( photo voltaic ) barely does more than just recuperate the energy of manufacture, distribution, and installation: where S
        \"HEY, EU!!! YOU PRETEND YOU DON'T SEE THE VISIT OF HUNGARIAN PREMIER TO RUSSIA? NOTHING TO SAY? PLEASE, GIVE US AN ANSWER AS S
        \"We don't need fossil fuels , solar is cheap and very simple to install:\\n\\n EU should order millions of those small systems
        \"I see a worrying focus on reducing energy consumption at the expense of in house climate. At this rate we are replacing one e
        \"SUSCO Organization in Kurdistan-Iraq please support us\\nwww.susco-kurdistan.org\",
        \"\\\\\\\\\\\\\\\\ \\square\\square ... \\square\\square\",
        \"mah cosa ACCADE in europa ? ANCHE questo progetto di energia pulita  cadrà sulle spese dei milioni di poveri a salario e p
      ],
      "_version_": "1601640593035806592"
    }
  ]
}
```

Install Banana from Github <https://github.com/lucidworks/banana>

Decompress the folder in the desktop and rename it to just “banana”

Copy it in to the Solr directory

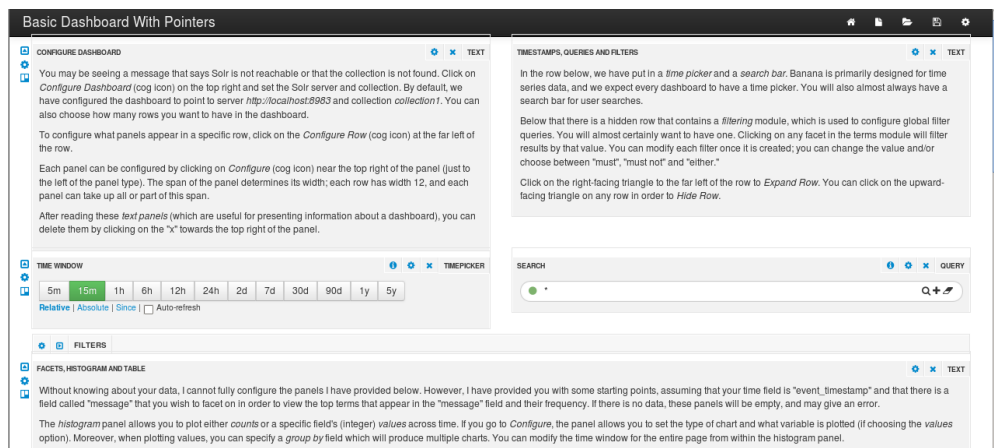
Sudo su

cp /home/cloudera/Desktop/banana/ /usr/lib/solr/webapps/

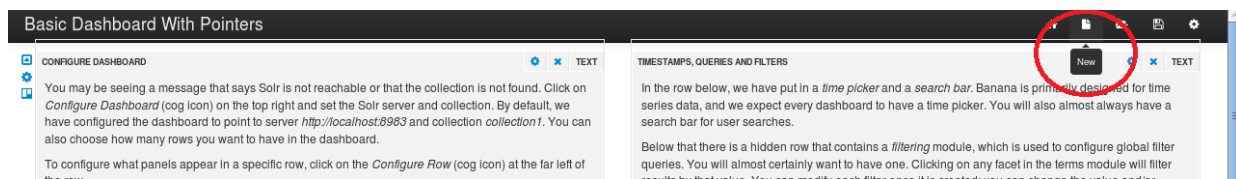
To deploy the new web app into the Solr server run the `tomcat-deployment.sh`
`/usr/lib/solr/tomcat-deployment.sh`

Go to the link to build your dashboard

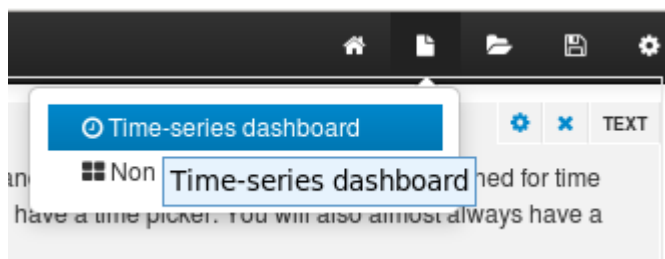
<http://quickstart.cloudera:8983/banana/src/index.html#/dashboard>



Click the button “New”



Choose time-series dashboard



In the configuration put the collection name and the time field

New Dashboard Settings

☐ Use Fusion?

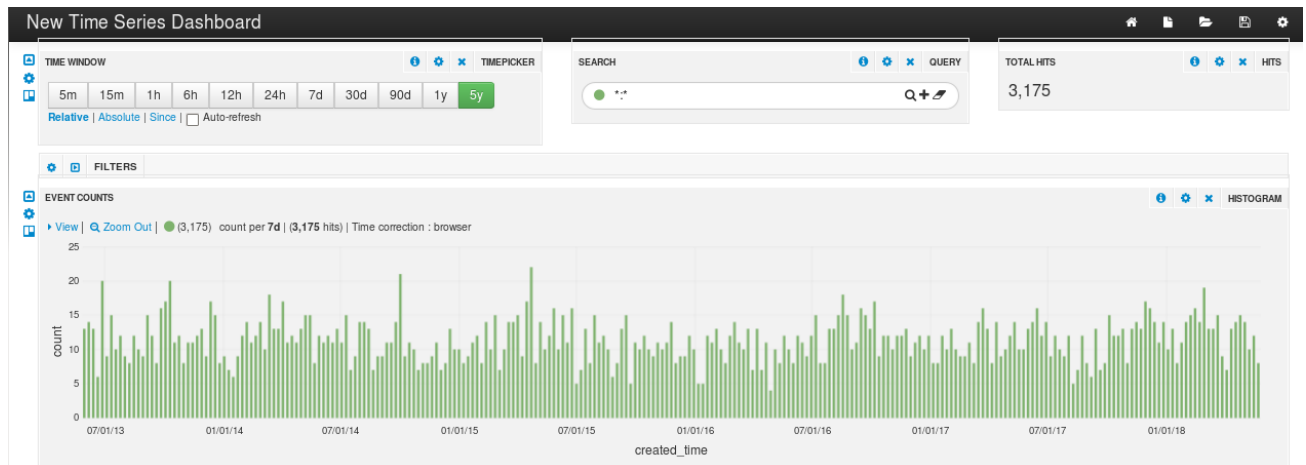
Solr Server / Fusion Query Pipeline

Collection Name

Time Field

Create Close

Now start customizing your dashboard, change the time picker



At the bottom, add a new row, name it, and set the Height to 250px

The screenshot shows the 'Dashboard Settings' dialog box. It has tabs for 'General', 'Rows', 'Controls', and 'Solr'. The 'Rows' tab is selected. It displays a table with columns for 'Title', 'Delete', and 'Move'. The table lists four rows: 'Query and Time Window', 'Filters', 'Graph', and 'Table'. Below the table, there are fields for 'Title', 'Height', and 'Editable'. The 'Title' field is set to 'Bars', the 'Height' field is set to '250px', and the 'Editable' checkbox is checked. At the bottom right, there are 'Create Row' and 'Close' buttons.

Title	Delete	Move
Query and Time Window	x	↓
Filters	x	↑ ↓
Graph	x	↑ ↓
Table	x	↑

Title: Bars Height: 250px Editable: ☒

Create Row Close

Add a new panel



Go to add panel and select bar in the Select Panel Type

Set the field to type (Facebook Post type)

Press Add panel when you finish

GeneralPanelsAdd Panel

Row Settings

Select Panel Type

bar

Experimental // Display the D3 Bar Chart with Tooltip.

Title

Span

4

Editable

☒

Inspect

☒

Field

type

Number of Bars

10

Queries

☒ Display query when Inspect

Panel Query

Add Panel

Close

Here you are

