Cloudera Runtime 7.2.18

Cloudera Search Tutorial

Date published: 2019-11-19
Date modified:



https://docs.cloudera.com/

Legal Notice

© Cloudera Inc. 2026. All rights reserved.

The documentation is and contains Cloudera proprietary information protected by copyright and other intellectual property rights. No license under copyright or any other intellectual property right is granted herein.

Unless otherwise noted, scripts and sample code are licensed under the Apache License, Version 2.0.

Copyright information for Cloudera software may be found within the documentation accompanying each component in a particular release.

Cloudera software includes software from various open source or other third party projects, and may be released under the Apache Software License 2.0 (“ASLv2”), the Affero General Public License version 3 (AGPLv3), or other license terms. Other software included may be released under the terms of alternative open source licenses. Please review the license and notice files accompanying the software for additional licensing information.

Please visit the Cloudera software product page for more information on Cloudera software. For more information on Cloudera support services, please visit either the Support or Sales page. Feel free to contact us directly to discuss your specific needs.

Cloudera reserves the right to change any products at any time, and without notice. Cloudera assumes no responsibility nor liability arising from the use of products, except as expressly agreed to in writing by Cloudera.

Cloudera, Cloudera Altus, HUE, Impala, Cloudera Impala, and other Cloudera marks are registered or unregistered trademarks in the United States and other countries. All other trademarks are the property of their respective owners.

Disclaimer: EXCEPT AS EXPRESSLY PROVIDED IN A WRITTEN AGREEMENT WITH CLOUDERA, CLOUDERA DOES NOT MAKE NOR GIVE ANY REPRESENTATION, WARRANTY, NOR COVENANT OF ANY KIND, WHETHER EXPRESS OR IMPLIED, IN CONNECTION WITH CLOUDERA TECHNOLOGY OR RELATED SUPPORT PROVIDED IN CONNECTION THEREWITH. CLOUDERA DOES NOT WARRANT THAT CLOUDERA PRODUCTS NOR SOFTWARE WILL OPERATE UNINTERRUPTED NOR THAT IT WILL BE FREE FROM DEFECTS NOR ERRORS, THAT IT WILL PROTECT YOUR DATA FROM LOSS, CORRUPTION NOR UNAVAILABILITY, NOR THAT IT WILL MEET ALL OF CUSTOMER’S BUSINESS REQUIREMENTS. WITHOUT LIMITING THE FOREGOING, AND TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, CLOUDERA EXPRESSLY DISCLAIMS ANY AND ALL IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, QUALITY, NON-INFRINGEMENT, TITLE, AND FITNESS FOR A PARTICULAR PURPOSE AND ANY REPRESENTATION, WARRANTY, OR COVENANT BASED ON COURSE OF DEALING OR USAGE IN TRADE.

Contents

Cloudera Search tutorial..... 4

Validating the Cloudera Search deployment.....4

Create a test collection..... 4
Index sample data..... 5
Query sample data..... 5

Indexing sample tweets with Cloudera Search.....6

Create a collection for tweets.....6
Copy sample tweets to HDFS..... 7
Using MapReduce batch indexing to index sample Tweets.....8
Batch indexing into online Solr servers using GoLive.....9
Batch indexing into offline Solr shards..... 11

Cloudera Search tutorial

This tutorial introduces you to Cloudera Search and demonstrates some of its basic capabilities to help you become familiar with the concepts and components involved in Search. It demonstrates creating a simple test collection to validate your Cloudera Search installation, and then continues on to more advanced use cases of batch indexing.

The topics in this tutorial assume you have deployed Cloudera Search. The examples in this tutorial use two shards, so make sure that your deployment includes at least two Solr servers.

This tutorial uses modified schema.xml and solrconfig.xml configuration files. In the versions of these files included with the tutorial, unused fields have been removed for simplicity. Original versions of these files include many additional options. For information on all available options, see the Apache Solr wiki:

- SchemaXml
- SolrConfigXml

Validating the Cloudera Search deployment

After installing and deploying Cloudera Search, validate the deployment by indexing and querying sample documents.



Important: This tutorial does not account for Apache Ranger authorization. If you are using Ranger authorization, you must create policies to allow the users and actions described in the tutorial.

You can think of this as a type of "Hello, World!" for Cloudera Search to make sure that everything is installed and working properly.

Before beginning this process, make sure you have access to the Apache Solr admin web console. If your cluster is Kerberos-enabled, make sure you have access to the solr@EXAMPLE.COM Kerberos principal (where EXAMPLE.COM is your Kerberos realm name).

Create a test collection

Generate configuration files and upload the generated configuration to ZooKeeper, so that you can create a collection where you can index sample data.

Procedure

1. Make sure that the SOLR_ZK_ENSEMBLE environment variable is set in /etc/solr/conf/solr-env.sh.
   For example:

   ```
   cat /etc/solr/conf/solr-env.sh
   ```

   ```
   export SOLR_ZK_ENSEMBLE=zk01.example.com:2181,zk02.example.com:2181,zk03
   .example.com:2181/solr
   ```

   This is automatically set on hosts with a Solr Server or Gateway role in Cloudera Manager.
2. If you are using Kerberos, kinit as the user that has privileges to create the collection.
   For example:

   ```
   kinit SOLR@EXAMPLE.COM
   ```

   Replace *SOLR@EXAMPLE.COM* with your user name and Kerberos realm name respectively.

3. Generate configuration files for the collection:

```
solrctl instancedir --generate $HOME/test_collection_config
```

4. Upload the configuration to ZooKeeper:

```
solrctl config --upload test_collection_config $HOME/test_collection_con
fig
```

5. Create a new collection with two shards (specified by the -s parameter) using the named configuration (specified by the -c parameter).

```
solrctl collection --create test_collection -s 2 -c test_collection_config
```

Index sample data

Cloudera Search includes sample data for testing and validation. Run the relevant command to index this data for searching.

About this task

Replace *SEARCH01.EXAMPLE.COM* in the example below with the name of any host running the Solr Server process.

Procedure

1. SSH to a Solr host.
2. Run the following command:

   TLS enabled:

```
cd /opt/cloudera/parcels/CDH/share/doc/solr-doc*/example/exampledocs
                    find *.xml -exec curl -i -k --negotiate -u: https:
//SEARCH01.EXAMPLE.COM:8985/solr/test_collection/update -H "Content-Type:
text/xml" --data-binary @{} \;
```

   TLS disabled:

```
cd /opt/cloudera/parcels/CDH/share/doc/solr-doc*/example/exampledocs
               java -Durl=http://SEARCH01.EXAMPLE.COM:8983/solr/test_coll
ection/update -jar post.jar *.xml
```

Query sample data

Run a query to verify that the sample data is successfully indexed and that you are able to search it.

About this task

Procedure

1. SSH into a cluster node.
2. Open the Solr admin web interface in a browser.
   - TLS enabled: https://*SEARCH01.EXAMPLE.COM*:8985/solr
   - TLS disabled: http://*SEARCH01.EXAMPLE.COM*:8983/solr

Replace *SEARCH01.EXAMPLE.COM* with the name of any host running the Solr Server process. If you have security enabled on your cluster, enter the credentials for the Kerberos principal when prompted.

3. Select Cloud from the left panel.
4. From the Collection Selector drop-down menu in the left panel, select the test_collection collection.
5. Select Query from the left panel and click Execute Query. If you see results such as the following, indexing was successful:

```
"response":{"numFound":32,"start":0,"maxScore":1.0,"docs":[
      {
        "id":"SP2514N",
        "name":["Samsung SpinPoint P120 SP2514N - hard drive - 250 GB -
ATA-133"],
        "manu":["Samsung Electronics Co. Ltd."],
        "manu_id_s":"samsung",
        "cat":["electronics",
          "hard drive"],
        "features":["7200RPM, 8MB cache, IDE Ultra ATA-133",
          "NoiseGuard, SilentSeek technology, Fluid Dynamic Bearing (FDB)
 motor"],
        "price":[92.0],
        "popularity":[6],
        "inStock":[true],
        "manufacturedate_dt":"2006-02-13T15:26:37Z",
        "store":["35.0752,-97.032"],
        "manu_str":["Samsung Electronics Co. Ltd."],
        "_version_":1650678864820568064,
        "cat_str":["electronics",
          "hard drive"],
        "name_str":["Samsung SpinPoint P120 SP2514N - hard drive - 250 GB
 - ATA-133"],
        "features_str":["7200RPM, 8MB cache, IDE Ultra ATA-133",
          "NoiseGuard, SilentSeek technology, Fluid Dynamic Bearing (FDB)
 motor"],
        "store_str":["35.0752,-97.032"]},
```

Indexing sample tweets with Cloudera Search

After you have verified that Cloudera Search is installed and running properly, you can experiment with other methods of ingesting and indexing data. This tutorial uses tweets to demonstrate batch indexing.

Related Information

To learn more about Solr, see the Apache Solr Tutorial

Create a collection for tweets

In this part of the Cloudera Search tutorial, you create a collection for tweets.

About this task

The remaining examples in the tutorial use the same collection, so make sure that you follow these instructions carefully.

Procedure

1. On a host with Solr Server installed, make sure that the SOLR_ZK_ENSEMBLE environment variable is set in /etc/solr/conf/solr-env.sh.

For example:

```
cat /etc/solr/conf/solr-env.sh
```

```
export SOLR_ZK_ENSEMBLE=zk01.example.com:2181,zk02.example.com:2181,zk03
.example.com:2181/solr
```

This is automatically set on hosts with a Solr Server or Gateway role in Cloudera Manager

2. If you are using Kerberos, kinit as the user that has privileges to create the collection.

   For example:

   ```
   kinit SOLR@EXAMPLE.COM
   ```

   Replace *SOLR@EXAMPLE.COM* with your user name and Kerberos realm name respectively.
3. Generate the configuration files for the collection, including the tweet-specific managed-schema:

   ```
   solrctl instancedir --generate $HOME/cloudera_tutorial_tweets_config
                     cp /opt/cloudera/parcels/CDH/share/doc/search*/search-
   crunch/solr/collection1/conf/schema.xml $HOME/cloudera_tutorial_tweets_c
   onfig/conf/managed_schema
   ```

   To the overwrite confirmation prompt:

   ```
   cp: overwrite 'cloudera_tutorial_tweets_config/conf/managed-schema'?
   ```

   type 'y'.
4. Upload the configuration to ZooKeeper:

   ```
   solrctl config --upload cloudera_tutorial_tweets_config $HOME/cloudera_t
   utorial_tweets_config
   ```
5. Create a new collection with two shards (specified by the -s parameter) using the named configuration (specified by the -c parameter):

   ```
   solrctl collection --create cloudera_tutorial_tweets -s 2 -c cloudera_tu
   torial_tweets_config
   ```
6. Verify that the collection is live. Open the Solr admin web interface in a browser by accessing the relevant URL:

   - TLS enabled: https://*SEARCH01.EXAMPLE.COM*:8985/solr/#/~cloud
   - TLS disabled: http://*SEARCH01.EXAMPLE.COM*:8983/solr/#/~cloud

   If you have Kerberos authentication enabled on your cluster, enter the credentials for the solr@EXAMPLE.COM principal when prompted. Replace *SEARCH01.EXAMPLE.COM* with the name of any host running the Solr Server process. Look for the cloudera_tutorial_tweets collection to verify that it exists.
7. Prepare the configuration for use with MapReduce:

   ```
   cp -r $HOME/cloudera_tutorial_tweets_config $HOME/cloudera_tutorial_twee
   ts_mr_config
   ```

Copy sample tweets to HDFS

Copy the provided sample tweets to HDFS. These tweets are used to demonstrate the batch indexing capabilities of Cloudera Search.

Procedure

1. Copy the provided sample tweets to HDFS:

   Security Enabled:

   a. `kinit [***HDFS@EXAMPLE.COM***]`

   b. `hdfs dfs -mkdir -p /user/[***USER***]`

   c. `hdfs dfs -chown [***USER***]:[***GROUP***] /user/[***USER***]`

   d. `kinit [***USER@EXAMPLE.COM***]`

   e. `hdfs dfs -mkdir -p /user/[***USER***]/indir`

   f. `hdfs dfs -put /opt/cloudera/parcels/CDH/share/doc/search*/examples/test-documents/sample-statuses-*.avro /user/[***USER***]/indir/`

   g. `hdfs dfs -ls /user/[***USER***]/indir`

   Security Disabled: Run the following commands as *[***USER***]*:

   ```
   sudo -u hdfs hdfs dfs -mkdir -p /user/[***USER***]
   ```

   ```
   sudo -u hdfs hdfs dfs -chown [***USER***]:[***GROUP***] /user/[***USER***]
   ```

   ```
   hdfs dfs -mkdir -p /user/[***USER***]/indir
   ```

   ```
   hdfs dfs -put /opt/cloudera/parcels/CDH/share/doc/search*/examples/test-documents/sample-statuses-*.avro /user/[***USER***]/indir/
   ```

   ```
   hdfs dfs -ls /user/[***USER***]/indir
   ```

2. Ensure that outdir is empty and exists in HDFS:

   ```
   hdfs dfs -rm -r -skipTrash /user/[***USER***]/outdir
   ```

   ```
   hdfs dfs -mkdir /user/[***USER***]/outdir
   ```

   ```
   hdfs dfs -ls /user/[***USER***]/outdir
   ```

What to do next

The sample tweets are now in HDFS and ready to be indexed. Continue to the next section to index the sample tweets.

Using MapReduce batch indexing to index sample Tweets

Batch indexing is useful for periodically indexing large amounts of data, or for indexing a dataset for the first time.

Before continuing, make sure that you have completed the procedures earlier in the tutorial.

Batch indexing into online Solr servers using GoLive

MapReduceIndexerTool is a MapReduce batch job driver that creates a set of Solr index shards from a set of input files and writes the indexes into HDFS in a flexible, scalable, and fault-tolerant manner. Using GoLive, MapReduceIndexerTool also supports merging the output shards into a set of live customer-facing Solr servers.

About this task

Warning: Batch indexing into offline Solr shards is not supported in environments in which batch indexing into online Solr servers using GoLive occurs.

Procedure

1. If you are working with a secured cluster, configure your client JAAS file ($HOME/jaas.conf) as follows:

```
Client {
 com.sun.security.auth.module.Krb5LoginModule required
 useKeyTab=false
 useTicketCache=true
 principal="solr@EXAMPLE.COM";
 };
```

2. If you are using Kerberos, kinit as a user that has privileges to update the collection:

```
kinit solr@EXAMPLE.COM
```

   Replace EXAMPLE.COM with your Kerberos realm name.

3. Delete any existing documents in the cloudera_tutorial_tweets collection. If your cluster does not have security enabled, run the following commands as the solr user by adding sudo -u solr before the command:

```
solrctl collection --deletedocs cloudera_tutorial_tweets
```

4. Run the MapReduce job with the --go-live parameter. Replace *NN01.EXAMPLE.COM* and *ZK01.EXAMPLE.COM* with your NameNode and ZooKeeper hostnames, respectively.

   - Security Enabled:

```
YARN_OPTS="-Djava.security.auth.login.config=/PATH/TO/jaas.conf" yarn
 jar /opt/cloudera/parcels/CDH/lib/solr/contrib/mr/search-mr-job.jar org
.apache.solr.hadoop.MapReduceIndexerTool -D 'mapred.child.java.opts=-
Xmx500m' -D 'mapreduce.job.user.classpath.first=true' --log4j /opt/clou
dera/parcels/CDH/share/doc/search*/examples/solr-nrt/log4j.properties
 --morphline-file /opt/cloudera/parcels/CDH/share/doc/search*/examples/
solr-nrt/test-morphlines/tutorialReadAvroContainer.conf --output-dir hdf
s://NN01.EXAMPLE.COM:8020/user/JDOE/outdir --verbose --go-live --zk-hos
t ZK01.EXAMPLE.COM:2181/solr --collection cloudera_tutorial_tweets hdf
s://NN01.EXAMPLE.COM:8020/user/JDOE/indir
```

   - Security Disabled:

```
yarn jar /opt/cloudera/parcels/CDH/lib/solr/contrib/mr/search-mr-job.jar
 org.apache.solr.hadoop.MapReduceIndexerTool -D 'mapred.child.java.opts=
-Xmx500m' -D 'mapreduce.job.user.classpath.first=true' --log4j /opt/clou
dera/parcels/CDH/share/doc/search*/examples/solr-nrt/log4j.properties
 --morphline-file /opt/cloudera/parcels/CDH/share/doc/search*/examples/
solr-nrt/test-morphlines/tutorialReadAvroContainer.conf --output-dir hdf
s://NN01.EXAMPLE.COM:8020/user/JDOE/outdir --verbose --go-live --zk-hos
t ZK01.EXAMPLE.COM:2181/solr --collection cloudera_tutorial_tweets hdf
s://NN01.EXAMPLE.COM:8020/user/JDOE/indir
```

This command requires a morphline file, which must include a SOLR_LOCATOR directive. Any CLI parameters for --zkhost and --collection override the parameters of the solrLocator. The SOLR_LOCATOR directive might appear as follows:

```
SOLR_LOCATOR : {
  # Name of solr collection
  collection : collection_name
  # ZooKeeper ensemble
  zkHost : "zk01.example.com:2181,zk02.example.com:2181,zk03.example.com
:2181/solr"
}

morphlines : [
  {
    id : morphline1
    importCommands : ["org.kitesdk.**", "org.apache.solr.**"]
    commands : [
      { generateUUID { field : id } }

      { # Remove record fields that are unknown to Solr schema.xml.
        # Recall that Solr throws an exception on any attempt to load a
document that
        # contains a field that isn't specified in schema.xml.
        sanitizeUnknownSolrFields {
          solrLocator : ${SOLR_LOCATOR} # Location from which to fetch
Solr schema
        }
      }

      { logDebug { format : "output record: {}", args : ["@{}"] } }

      {
        loadSolr {
          solrLocator : ${SOLR_LOCATOR}
        }
      }
    ]
  }
]
```

For help on how to run the MapReduce job, run the following command:

```
yarn jar /opt/cloudera/parcels/CDH/lib/solr/contrib/mr/search-mr-*-job.jar
 org.apache.solr.hadoop.MapReduceIndexerTool --help
```

For development purposes, you can use the --dry-run option to run in local mode and print documents to stdout instead of loading them into Solr. Using this option causes the morphline to run locally without submitting a job to MapReduce. Running locally provides faster turnaround during early trial and debug sessions.

To print diagnostic information, such as the content of records as they pass through the morphline commands, enable TRACE log level diagnostics by adding the following entry to your log4j.properties file:

```
log4j.logger.org.kitesdk.morphline=TRACE
```

The log4j.properties file location can be specified by using the MapReduceIndexerTool --log4j /path/to/log4j.properties command-line option.

5. Check the job status at:

```
http://RM01.EXAMPLE.COM:8088/ui2/#/yarn-apps/apps
```

For secure clusters, replace http with https and port 8088 with 8090.

6. When the job is complete, run a Solr query. For example, for a Solr server running on search01.example.com, go to one of the following URLs in a browser, depending on whether you have enabled security on your cluster:
   - Security Enabled: https://search01.example.com:8985/solr/cloudera_tutorial_tweets/select?q=*:*
   - Security Disabled: http://search01.example.com:8983/solr/cloudera_tutorial_tweets/select?q=*:*

   If indexing was successful, this page displays the first 10 query results.

Batch indexing into offline Solr shards

Batch indexing into offline Solr shards is mainly intended for offline use-cases by advanced users. Use cases requiring read-only indexes for searching can be handled by using batch indexing without the --go-live option. By not using GoLive, you can avoid copying datasets between segments, thereby reducing resource utilization.

About this task

Running the MapReduce job without GoLive causes the job to create a set of Solr index shards from a set of input files and write the indexes to HDFS. You can then explicitly point each Solr server to one of the HDFS output shard directories.

Procedure

1. If you are working with a secured cluster, configure your client JAAS file ($HOME/jaas.conf) as follows:

```
Client {
 com.sun.security.auth.module.Krb5LoginModule required
 useKeyTab=false
 useTicketCache=true
 principal="solr@EXAMPLE.COM";
 };
```

2. If you are using Kerberos, kinit as the user that has privileges to update the collection:

```
kinit jdoe@EXAMPLE.COM
```

   Replace EXAMPLE.COM with your Kerberos realm name.

3. Delete any existing documents in the cloudera_tutorial_tweets collection. If your cluster does not have security enabled, run the following commands as the solr user by adding sudo -u solr before the command:

```
solrctl collection --deletedocs cloudera_tutorial_tweets
```

4. Delete the contents of the outdir directory:

```
hdfs dfs -rm -r -skipTrash /user/JDOE/outdir/*
```

5. Run the MapReduce job as follows, replacing *NN01.EXAMPLE.COM* in the command with your NameNode hostname.
   - Security enabled:

```
YARN_OPTS="-Djava.security.auth.login.config=/PATH/TO/jaas.conf" yarn
 jar /opt/cloudera/parcels/CDH/lib/solr/contrib/mr/search-mr-*-job.jar
 org.apache.solr.hadoop.MapReduceIndexerTool -D 'mapred.child.java.opt
s=-Xmx500m' -D 'mapreduce.job.user.classpath.first=true' --log4j /opt/
cloudera/parcels/CDH/share/doc/search*/examples/solr-nrt/log4j.propert
ies --morphline-file /opt/cloudera/parcels/CDH/share/doc/search*/exa
mples/solr-nrt/test-morphlines/tutorialReadAvroContainer.conf --outp
ut-dir hdfs://NN01.EXAMPLE.COM:8020/user/JDOE/outdir --verbose --zk-h
```

```
ost ZK01.EXAMPLE.COM:2181/solr --collection cloudera_tutorial_tweets --s
hards 2 hdfs://NN01.EXAMPLE.COM:8020/user/JDOE/indir
```

- Security disabled:

```
yarn jar /opt/cloudera/parcels/CDH/lib/solr/contrib/mr/search-mr-*-job.j
ar org.apache.solr.hadoop.MapReduceIndexerTool -D 'mapred.child.java
.opts=-Xmx500m' -D 'mapreduce.job.user.classpath.first=true' --log4j /
opt/cloudera/parcels/CDH/share/doc/search*/examples/solr-nrt/log4j.pro
perties --morphline-file /opt/cloudera/parcels/CDH/share/doc/search*/
examples/solr-nrt/test-morphlines/tutorialReadAvroContainer.conf --outp
ut-dir hdfs://NN01.EXAMPLE.COM:8020/user/JDOE/outdir --verbose --zk-h
ost ZK01.EXAMPLE.COM:2181/solr --collection cloudera_tutorial_tweets --s
hards 2 hdfs://NN01.EXAMPLE.COM:8020/user/JDOE/indir
```

6. Check the job status at:

```
http://RM01.EXAMPLE.COM:8088/ui2/#/yarn-apps/apps
```

For secure clusters, replace http with https and port 8088 with 8090.

7. After the job is completed, check the generated index files. Individual shards are written to the results directory with names of the form part-00000, part-00001, part-00002, and so on. This example has two shards:

```
hdfs dfs -ls /user/JDOE/outdir/results
```

```
hdfs dfs -ls /user/JDOE/outdir/results/part-00000/data/index
```

8. In the Cloudera Manager web console for the cluster, stop the Solr service (Solr service Actions Stop).
9. Identify the paths to each Solr core:

```
hdfs dfs -ls /solr/cloudera_tutorial_tweets
```

```
Found 2 items
drwxr-xr-x   - solr solr          0 2017-03-13 06:20 /solr/cloudera_tutori
al_tweets/core_node1
drwxr-xr-x   - solr solr          0 2017-03-13 06:20 /solr/cloudera_tut
orial_tweets/core_node2
```

10. Move the index shards into place.
    a) (Kerberos only) Switch to the solr user:

```
kinit solr@EXAMPLE.COM
```

    b) Remove outdated files. If your cluster does not have security enabled, run the following commands as the solr user by adding sudo -u solr before the command:

```
hdfs dfs -rm -r -skipTrash /solr/cloudera_tutorial_tweets/core_node1/dat
a/index
hdfs dfs -rm -r -skipTrash /solr/cloudera_tutorial_tweets/core_node1/dat
a/tlog
hdfs dfs -rm -r -skipTrash /solr/cloudera_tutorial_tweets/core_node2/
data/index
hdfs dfs -rm -r -skipTrash /solr/cloudera_tutorial_tweets/core_node2/
data/tlog
```

c) Change ownership of the results directory to solr. If your cluster has security enabled, kinit as the HDFS superuser (hdfs by default) before running the following command. If your cluster does not have security enabled, run the command as the HDFS superuser by adding sudo -u hdfs before the command:

```
hdfs dfs -chown -R solr /user/JDOE/outdir/results
```

d) (Kerberos only) Switch to the solr user:

```
kinit solr@EXAMPLE.COM
```

e) Move the two index shards into place. If your cluster does not have security enabled, run the following commands as the solr user by adding sudo -u solr before the command:

```
hdfs dfs -mv /user/JDOE/outdir/results/part-00000/data/index /solr/clou
dera_tutorial_tweets/core_node1/data
```

```
hdfs dfs -mv /user/JDOE/outdir/results/part-00001/data/index /solr/clou
dera_tutorial_tweets/core_node2/data
```

11. In the Cloudera Manager web console for the cluster, start the Solr service (Solr service Actions Start).

12. Run some Solr queries. For example, for a Solr server running on search01.example.com, go to one of the following URLs in a browser, depending on whether you have enabled security on your cluster:

- Security enabled: https://search01.example.com:8985/solr/cloudera_tutorial_tweets/select?q=*:*
- Security disabled: http://search01.example.com:8983/solr/cloudera_tutorial_tweets/select?q=*:*

If indexing was successful, this page displays the first 10 query results.