Assignment 18.2

Associated Data Files

Link: https://drive.google.com/file/d/0Bxr27gVaXO5sU3g5ZUFhVDYxczQ/view?usp=sharing

Problem Statement

1 Perform the earlier operation using Java program.

2 The program should be able to check for the presence of 'customer' table.

3 In case the table exists, it must be dropped and recreated.

Note: Prefer using Batch Load for this purpose

we will be discussing the steps to perform data bulk loading file contents from HDFS path into an HBase table using Java MapReduce API

Bulk Loading:

HBase gives us random, real-time, read/write access to Big Data, generally we try to load data to HBase table via the client APIs or by using a MapReduce job with TableOutputFormat, but those approaches are problematic, Instead, the HBase bulk loading feature is much easier to use and can insert the same amount of data more quickly

When To Use Bulk Loading:

If you have any of these symptoms, bulk loading is probably a good choice for you:

- You needed to tweak MemStores to use most of the memory.
- You needed to either use bigger WALs or bypass them entirely.
- Your compaction and flush queues are in the hundreds.
- Your GC is out of control because your inserts range in the MBs.
- Your latency goes out of your SLA when you import data.

In general speak bulk loading is the process of preparing and loading HFiles directly into the RegionServers, thus bypassing write path and obviating issues related to them.

The Bulk Loading Process Looks Like:

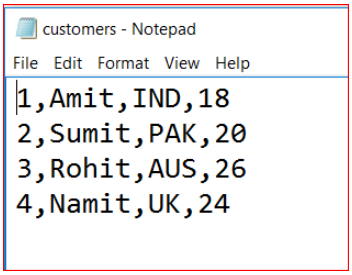
1.Extract data from source(in our case from Text File).

2.Transform data into HFiles.

3.Loading the files into HBase by telling RegionServers where to find them.

So, after simple overview of bulk loading let us see a simple example by loading a comma separated text file into HBase using Bulkloading feature, To do this please follow below steps.

Our Sample Data File User.Txt Looks Like:



It Is Stored As Below Format:

RowKey(id), Name, Location, Age

Steps to perform the bulk load into HBase Table

Step 1: start hadoop and hbase daemons.

start-all.sh

start-hbase.sh

jps

```
[acadgild@localhost dataset]$ start-all.sh
This script is Deprecated. Instead use start-dfs.sh and start-yarn.sh
17/08/19 13:04:48 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Starting namenodes on [localhost]
localhost: starting namenode, logging to /usr/local/hadoop-2.6.0/logs/hadoop-acadgild-namenode-localhost.localdomain.out
localhost: starting datanode, logging to /usr/local/hadoop-2.6.0/logs/hadoop-acadgild-datanode-localhost.localdomain.out
Starting secondary namenodes [0.0.0.0]
0.0.0.0: starting secondarynamenode, logging to /usr/local/hadoop-2.6.0/logs/hadoop-acadgild-secondarynamenode-localhost.localdomain.out
17/08/19 13:05:06 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
starting yarn daemons
starting resourcemanager, logging to /usr/local/hadoop-2.6.0/logs/yarn-acadgild-resourcemanager-localhost.localdomain.out
localhost: starting nodemanager, logging to /usr/local/hadoop-2.6.0/logs/yarn-acadgild-nodemanager-localhost.localdomain.out
[acadgild@localhost dataset]$ start-hbase.sh
starting master, logging to /usr/local/hbase/logs/hbase-acadgild-master-localhost.localdomain.out
[acadgild@localhost dataset]$ jps
2501 DataNode
3333 HMaster
3430 Jps
2375 NameNode
2807 ResourceManager
2664 SecondaryNameNode
2908 NodeManager
[acadgild@localhost dataset]$
```

Step 2: move customers.dat file into hdfs.

hadoop fs -put customers.dat /user/acadgild/hadoop/dataset

hadoop fs -ls /user/acadgild/hadoop/dataset

```
[acadgild@localhost dataset]$ hadoop fs -put customers.dat /user/acadgild/hadoop/dataset
17/08/19 13:09:40 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
[acadgild@localhost dataset]$ hadoop fs -ls /user/acadgild/hadoop/dataset
17/08/19 13:09:47 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Found 1 items
-rw-r--r--   1 acadgild supergroup         60 2017-08-19 13:09 /user/acadgild/hadoop/dataset/customers.dat
[acadgild@localhost dataset]$
[acadgild@localhost dataset]$
```

Step 3: Transform the data into HFiles via MapReduce job (map only)

Here we write a MapReduce job which will process our data and create HFile. There will be only Mapper class and will be no Reducer class. In our code, we configure HFileOutputFormat.configureIncrementalLoad() doing which HBase creates its own Reducer class.

HBaseBulkLoad.java

The above step finishes the MapReduce programming part.

Now, we need to create a new table in Hbase to import table contents from hdfs input directory. So, follow the below steps to import the contents from hdfs path to Hbase table.

Step 4: Program that checks the existence of the 'customer' table in HBase, if exists then drop the table and recreate the new table

RecreateTable.java

Step 5: Enter HBase shell and check existence of the table.

hbase shell

list

describe 'customer'

scan 'customer'

```
hbase(main):004:0> list
TABLE
clicks
customer
2 row(s) in 0.0110 seconds

=> ["clicks", "customer"]
hbase(main):005:0> describe 'customer'
Table customer is ENABLED
customer
COLUMN FAMILIES DESCRIPTION
{NAME => 'details', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY => 'false', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCODING => 'NONE', TTL
=> 'FOREVER', COMPRESSION => 'NONE', MIN_VERSIONS => '0', BLOCKCACHE => 'true', BLOCKSIZE => '65536', REPLICATION_SCOPE => '0'}
1 row(s) in 0.0260 seconds

hbase(main):006:0>
```

```
hbase(main):006:0> scan 'customer'
ROW                                   COLUMN+CELL
 1                                    column=details:age, timestamp=1502989458189, value=18
 1                                    column=details:location, timestamp=1502989458189, value=IND
 1                                    column=details:name, timestamp=1502989458189, value=Amit
 2                                    column=details:age, timestamp=1502989458200, value=20
 2                                    column=details:location, timestamp=1502989458200, value=PAK
 2                                    column=details:name, timestamp=1502989458200, value=Sumit
 3                                    column=details:age, timestamp=1502989458201, value=26
 3                                    column=details:location, timestamp=1502989458201, value=AUS
 3                                    column=details:name, timestamp=1502989458201, value=Rohit
 4                                    column=details:age, timestamp=1502989458201, value=24
 4                                    column=details:location, timestamp=1502989458201, value=UK
 4                                    column=details:name, timestamp=1502989458201, value=Namit
4 row(s) in 0.1430 seconds
```

Step 6: Export hadoop_classpath:

In the next step, we need to load the HBase library files into the Hadoop classpath this enables the Hadoop client to connect to HBase and get the number of splits.

export HADOOP_CLASSPATH=$HBASE_HOME/lib/*

echo $HADOOP_CLASSPATH

```
[acadgild@localhost dataset]$ export HADOOP_CLASSPATH=$HBASE_HOME/lib/*
[acadgild@localhost dataset]$ echo $HADOOP_CLASSPATH
/usr/local/hbase/lib/activation-1.1.jar /usr/local/hbase/lib/aopalliance-1.0.jar /usr/local/hbase/lib/asm-3.1.jar /usr/local/hbase/lib/avro-1.7.4.j
ar /usr/local/hbase/lib/commons-beanutils-1.7.0.jar /usr/local/hbase/lib/commons-beanutils-core-1.8.0.jar /usr/local/hbase/lib/commons-cli-1.2.jar
/usr/local/hbase/lib/commons-codec-1.7.jar /usr/local/hbase/lib/commons-collections-3.2.1.jar /usr/local/hbase/lib/commons-compress-1.4.1.jar /usr/
local/hbase/lib/commons-configuration-1.6.jar /usr/local/hbase/lib/commons-daemon-1.0.13.jar /usr/local/hbase/lib/commons-digester-1.8.jar /usr/loc
al/hbase/lib/commons-el-1.0.jar /usr/local/hbase/lib/commons-httpclient-3.1.jar /usr/local/hbase/lib/commons-io-2.4.jar /usr/local/hbase/lib/common
s-lang-2.6.jar /usr/local/hbase/lib/commons-logging-1.1.1.jar /usr/local/hbase/lib/commons-math-2.1.jar /usr/local/hbase/lib/commons-net-3.1.jar /u
sr/local/hbase/lib/findbugs-annotations-1.3.9-1.jar /usr/local/hbase/lib/gmbal-api-only-3.0.0-b023.jar /usr/local/hbase/lib/grizzly-framework-2.1.2
.jar /usr/local/hbase/lib/grizzly-http-2.1.2.jar /usr/local/hbase/lib/grizzly-http-server-2.1.2.jar /usr/local/hbase/lib/grizzly-http-servlet-2.1.2
.jar /usr/local/hbase/lib/grizzly-rcm-2.1.2.jar /usr/local/hbase/lib/guava-12.0.1.jar /usr/local/hbase/lib/guice-3.0.jar /usr/local/hbase/lib/guice
-servlet-3.0.jar /usr/local/hbase/lib/hadoop-annotations-2.2.0.jar /usr/local/hbase/lib/hadoop-auth-2.2.0.jar /usr/local/hbase/lib/hadoop-client-2.
2.0.jar /usr/local/hbase/lib/hadoop-common-2.2.0.jar /usr/local/hbase/lib/hadoop-hdfs-2.2.0.jar /usr/local/hbase/lib/hadoop-mapreduce-client-app-2.
2.0.jar /usr/local/hbase/lib/hadoop-mapreduce-client-common-2.2.0.jar /usr/local/hbase/lib/hadoop-mapreduce-client-core-2.2.0.jar /usr/local/hbase/
lib/hadoop-mapreduce-client-jobclient-2.2.0.jar /usr/local/hbase/lib/hadoop-mapreduce-client-shuffle-2.2.0.jar /usr/local/hbase/lib/hadoop-yarn-api
-2.2.0.jar /usr/local/hbase/lib/hadoop-yarn-client-2.2.0.jar /usr/local/hbase/lib/hadoop-yarn-common-2.2.0.jar /usr/local/hbase/lib/hadoop-yarn-ser
ver-common-2.2.0.jar /usr/local/hbase/lib/hadoop-yarn-server-nodemanager-2.2.0.jar /usr/local/hbase/lib/hamcrest-core-1.3.jar /usr/local/hbase/lib/
hbase-annotations-0.98.14-hadoop2.jar /usr/local/hbase/lib/hbase-checkstyle-0.98.14-hadoop2.jar /usr/local/hbase/lib/hbase-client-0.98.14-hadoop2.j
ar /usr/local/hbase/lib/hbase-common-0.98.14-hadoop2.jar /usr/local/hbase/lib/hbase-common-0.98.14-hadoop2-tests.jar /usr/local/hbase/lib/hbase-exa
mples-0.98.14-hadoop2.jar /usr/local/hbase/lib/hbase-hadoop2-compat-0.98.14-hadoop2.jar /usr/local/hbase/lib/hbase-hadoop-compat-0.98.14-hadoop2.ja
r /usr/local/hbase/lib/hbase-it-0.98.14-hadoop2.jar /usr/local/hbase/lib/hbase-it-0.98.14-hadoop2-tests.jar /usr/local/hbase/lib/hbase-prefix-tree-
0.98.14-hadoop2.jar /usr/local/hbase/lib/hbase-protocol-0.98.14-hadoop2.jar /usr/local/hbase/lib/hbase-resource-bundle-0.98.14-hadoop2.jar /usr/loc
al/hbase/lib/hbase-rest-0.98.14-hadoop2.jar /usr/local/hbase/lib/hbase-server-0.98.14-hadoop2.jar /usr/local/hbase/lib/hbase-server-0.98.14-hadoop2
-tests.jar /usr/local/hbase/lib/hbase-shell-0.98.14-hadoop2.jar /usr/local/hbase/lib/hbase-testing-util-0.98.14-hadoop2.jar /usr/local/hbase/lib/hb
ase-thrift-0.98.14-hadoop2.jar /usr/local/hbase/lib/high-scale-lib-1.1.1.jar /usr/local/hbase/lib/htrace-core-2.04.jar /usr/local/hbase/lib/httpcli
ent-4.1.3.jar /usr/local/hbase/lib/httpcore-4.1.3.jar /usr/local/hbase/lib/jackson-core-asl-1.8.8.jar /usr/local/hbase/lib/jackson-jaxrs-1.8.8.jar
/usr/local/hbase/lib/jackson-mapper-asl-1.8.8.jar /usr/local/hbase/lib/jackson-xc-1.8.8.jar /usr/local/hbase/lib/jamon-runtime-2.3.1.jar /usr/local
/hbase/lib/jasper-compiler-5.5.23.jar /usr/local/hbase/lib/jasper-runtime-5.5.23.jar /usr/local/hbase/lib/javax.inject-1.jar /usr/local/hbase/lib/j
avax.servlet-3.1.jar /usr/local/hbase/lib/javax.servlet-api-3.0.1.jar /usr/local/hbase/lib/jaxb-api-2.2.2.jar /usr/local/hbase/lib/jaxb-impl-2.2.3-
1.jar /usr/local/hbase/lib/jcodings-1.0.8.jar /usr/local/hbase/lib/jersey-client-1.8.jar /usr/local/hbase/lib/jersey-core-1.8.jar /usr/local/hbase/
lib/jersey-grizzly2-1.9.jar /usr/local/hbase/lib/jersey-guice-1.9.jar /usr/local/hbase/lib/jersey-json-1.8.jar /usr/local/hbase/lib/jersey-server-1
.8.jar /usr/local/hbase/lib/jersey-test-framework-core-1.9.jar /usr/local/hbase/lib/jersey-test-framework-grizzly2-1.9.jar /usr/local/hbase/lib/jet
s3t-0.6.1.jar /usr/local/hbase/lib/jettison-1.3.1.jar /usr/local/hbase/lib/jetty-6.1.26.jar /usr/local/hbase/lib/jetty-sslengine-6.1.26.jar /usr/lo
cal/hbase/lib/jetty-util-6.1.26.jar /usr/local/hbase/lib/joni-2.1.2.jar /usr/local/hbase/lib/jruby-complete-1.6.8.jar /usr/local/hbase/lib/jsch-0.1
.42.jar /usr/local/hbase/lib/jsp-2.1-6.1.14.jar /usr/local/hbase/lib/jsp-api-2.1-6.1.14.jar /usr/local/hbase/lib/jsr305-1.3.9.jar /usr/local/hbase/
lib/junit-4.11.jar /usr/local/hbase/lib/libthrift-0.9.0.jar /usr/local/hbase/lib/log4j-1.2.17.jar /usr/local/hbase/lib/management-api-3.0.0-b012.ja
r /usr/local/hbase/lib/metrics-core-2.2.0.jar /usr/local/hbase/lib/netty-3.6.6.Final.jar /usr/local/hbase/lib/paranamer-2.3.jar /usr/local/hbase/li
b/protobuf-java-2.5.0.jar /usr/local/hbase/lib/ruby /usr/local/hbase/lib/servlet-api-2.5-6.1.14.jar /usr/local/hbase/lib/slf4j-api-1.6.4.jar /usr/l
ocal/hbase/lib/slf4j-log4j12-1.6.4.jar /usr/local/hbase/lib/snappy-java-1.0.4.1.jar /usr/local/hbase/lib/xmlenc-0.52.jar /usr/local/hbase/lib/xz-1.
0.jar /usr/local/hbase/lib/zookeeper-3.4.6.jar
[acadgild@localhost dataset]$
```


Step 7: Create the FileToHBase.JAR file of the project and run the RecreateTable class to recreate the table.

hadoop jar FileToHBase.jar assignment2.RecreateTable

Goto hbase shell and run scan command.

scan ‘customer’

```
hbase(main):007:0> scan 'customer'
ROW                                   COLUMN+CELL
0 row(s) in 0.3220 seconds

hbase(main):008:0>
```

Original table was dropped and new ‘customer’ table was recreated.

Step 8: run HBaseBulkLoad class to generate HFile in hdfs.

hadoop jar <<JAR FILE>> <<package.ClassName>> <<HDFS FILE PATH>> <<HFILE PATH>> <<HBASE TABLE NAME>>

hadoop jar FileToHBase.jar assignment2.HBaseBulkLoad hadoop/dataset/customers.dat /hbase_output_dir customer

```
17/08/19 13:53:27 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1503128111050_0001
17/08/19 13:53:27 INFO impl.YarnClientImpl: Submitted application application_1503128111050_0001 to ResourceManager at /0.0.0.0:8032
17/08/19 13:53:27 INFO mapreduce.Job: The url to track the job: http://http://localhost:8088/proxy/application_1503128111050_0001/
17/08/19 13:53:27 INFO mapreduce.Job: Running job: job_1503128111050_0001
17/08/19 13:53:38 INFO mapreduce.Job: Job job_1503128111050_0001 running in uber mode : false
17/08/19 13:53:38 INFO mapreduce.Job:  map 0% reduce 0%
17/08/19 13:53:43 INFO mapreduce.Job:  map 100% reduce 0%
17/08/19 13:53:51 INFO mapreduce.Job:  map 100% reduce 100%
17/08/19 13:53:51 INFO mapreduce.Job: Job job_1503128111050_0001 completed successfully
```

Step 9: Check files under directory /hbase_output_dir in HDFS

hadoop fs -ls /hbase_output_dir

hadoop fs -ls /hbase_output_dir/details

hadoop fs -cat /hbase_output_dir/details/d794c339a3ca41708a0957f6ff5c5245

```
[acadgild@localhost dataset]$ hadoop fs -ls /hbase_output_dir
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/local/hbase/lib/slf4j-log4j12-1.6.4.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/local/hadoop-2.6.0/share/hadoop/common/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
17/08/19 13:55:53 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Found 2 items
-rw-r--r--   1 acadgild supergroup          0 2017-08-19 13:53 /hbase_output_dir/_SUCCESS
drwxr-xr-x   - acadgild supergroup          0 2017-08-19 13:53 /hbase_output_dir/details
[acadgild@localhost dataset]$ hadoop fs -ls /hbase_output_dir/details
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/local/hbase/lib/slf4j-log4j12-1.6.4.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/local/hadoop-2.6.0/share/hadoop/common/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
17/08/19 13:55:58 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Found 1 items
-rw-r--r--   1 acadgild supergroup       1509 2017-08-19 13:53 /hbase_output_dir/details/d794c339a3ca41708a0957f6ff5c5245
[acadgild@localhost dataset]$ hadoop fs -cat /hbase_output_dir/details/d794c339a3ca41708a0957f6ff5c5245
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/local/hbase/lib/slf4j-log4j12-1.6.4.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/local/hadoop-2.6.0/share/hadoop/common/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
17/08/19 13:56:09 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
DATABLK*▒▒▒▒▒▒▒▒▒▒@▒1detailsage]▒L▒181detailslocation]▒L▒IND1detailsname]▒L▒Amit2detailsage]▒L▒202detailslocation]▒L▒PAK2detailsname]▒L▒Sumit3detai
lsage]▒L▒263detailslocation]▒L▒AUS3detailsname]▒L▒Rohit4detailsage]▒L▒244detailslocation]▒L▒UK4detailsname]▒L▒Namit▒▒
                                                                                                                    BLMFBLK2
▒▒▒▒▒▒▒▒@)▒(▒▒B▒▒KDXROOT2($▒▒▒▒▒▒▒▒@E▒1detailsage]▒L▒ki▒cIDXROOT2@!▒_▒▒FILEINF2▒▒▒▒▒▒▒▒▒@PBUF▒

BLOOM_FILTER_TYPEROW
=
BULKLOAD_SOURCE_TASK%attempt_1503128111050_0001_r_000000_0

BULKLOAD_TIMESTAM]▒P▒

DELETE_FAMILY_COUNT
```

/hbase_output_dir/details/d794c339a3ca41708a0957f6ff5c5245 **this is HFILE**

Step 10: HBase Hadoop jar execution to load HFile in the HBase table.

Hadoop jar <<HBase Hadoop Jar>> completebulkload <<HFILE PATH>> <<HBASE TABLE NAME>>

hadoop jar /usr/local/hbase/lib/hbase-server-0.98.14-hadoop2.jar completebulkload /hbase_output_dir customer

```
17/08/19 14:03:14 INFO zookeeper.ClientCnxn: EventThread shut down
17/08/19 14:03:14 INFO zookeeper.ZooKeeper: Session: 0x15df96bbaab0015 closed
17/08/19 14:03:14 WARN mapreduce.LoadIncrementalHFiles: Skipping non-directory hdfs://localhost:9000/hbase_output_dir/_SUCCESS
17/08/19 14:03:14 INFO Configuration.deprecation: hadoop.native.lib is deprecated. Instead, use io.native.lib.available
17/08/19 14:03:14 INFO util.ChecksumType: Checksum using org.apache.hadoop.util.PureJavaCrc32
17/08/19 14:03:14 INFO util.ChecksumType: Checksum can use org.apache.hadoop.util.PureJavaCrc32C
17/08/19 14:03:14 INFO mapreduce.LoadIncrementalHFiles: Trying to load hfile=hdfs://localhost:9000/hbase_output_dir/details/d794c339a3ca41708a0957f
6ff5c5245 first=1 last=4
[acadgild@localhost dataset]$ _
```

Step 11: scan 'customer' table in HBase shell.

scan 'customer'

```
hbase(main):008:0> scan 'customer'
ROW                                   COLUMN+CELL
 1                                    column=details:age, timestamp=1503131028718, value=18
 1                                    column=details:location, timestamp=1503131028718, value=IND
 1                                    column=details:name, timestamp=1503131028718, value=Amit
 2                                    column=details:age, timestamp=1503131028718, value=20
 2                                    column=details:location, timestamp=1503131028718, value=PAK
 2                                    column=details:name, timestamp=1503131028718, value=Sumit
 3                                    column=details:age, timestamp=1503131028718, value=26
 3                                    column=details:location, timestamp=1503131028718, value=AUS
 3                                    column=details:name, timestamp=1503131028718, value=Rohit
 4                                    column=details:age, timestamp=1503131028718, value=24
 4                                    column=details:location, timestamp=1503131028718, value=UK
 4                                    column=details:name, timestamp=1503131028718, value=Namit
4 row(s) in 0.0620 seconds
```