Paper SAS033-2014

Techniques in Processing Data on Hadoop

Donna De Capite, SAS Institute Inc., Cary, NC

ABSTRACT

Before you can analyze your big data, you need to prepare the data for analysis. This paper discusses capabilities and techniques for using the power of SAS® to prepare big data for analytics. It focuses on how a SAS user can write code that will run in a Hadoop cluster and take advantage of the massive parallel processing power of Hadoop.

WHAT IS HADOOP?

Hadoop is an open-source Apache project that was developed to solve the big data problem. How do you know you have a big data problem? In simple terms, when you have exceeded the capacity of conventional database systems, you're dealing with big data. Companies like Google, Yahoo, and Facebook were faced with big data challenges early on. The Hadoop project resulted from their efforts to manage their data volumes. It is designed to run on a large number of machines that don't share any memory or disks. Hadoop allows for handling hardware failures through its massively parallel architecture. Data is spread across the cluster of machines using HDFS—Hadoop Distributed File System. Data is processed using MapReduce, a Java-based programming model for data processing on Hadoop.

WHY ARE COMPANIES TURNING TO HADOOP?

Put simply, companies want to take advantage of the relatively low-cost infrastructure available with a Hadoop environment. Hadoop uses lower-cost commodity hardware to store and process data. Users with a traditional storage area network (SAN) are interested in moving more of their data into a Hadoop cluster.

WHAT CAN SAS DO IN HADOOP?

SAS can process all your data on Hadoop. SAS can process your data feeds and format your data in a meaningful way so that you can do analytics. You can use SAS to query and use the SAS DATA step against your Hive and Impala data. This paper takes you through the steps of getting started with Hadoop. If you already have experience with the basics of MapReduce and Pig, you can jump to the methods more centric to SAS of processing your data using the SAS language.

GETTING STARTED WITH HADOOP

In a Hadoop cluster, the configuration file is key to communicating with the Hadoop cluster. The examples in this paper use a basic configuration file.

Here is an example config.xml file. The settings should be updated to point to the specific Hadoop cluster. This file is located at \\machine\config.xml.

A macro variable also references this file:

```
%let cfgFile="\\machine\config.xml";
```

```
<configuration>
   <property>
      <name>fs.default.name</name>
      <value>hdfs://servername.demo.sas.com:8020</value>
   </property>
```

```
        <property>
           <name>mapred.job.tracker</name>
           <value>servername.demo.sas.com:8021</value>
        </property>
      </configuration>
```

HELLO WORLD!

The “Hello World” of Hadoop is working with the Shakespeare examples on the Apache Hadoop pages. Here’s an example to get started with using SAS to process this data. In this example, the Shakespeare data is moved to c:\public\tmp.

The first thing that you need to do is to land your Shakespeare data out on your cluster. There is always more than one way to do things, so this is not an exhaustive list of techniques. The file system for Hadoop is called the Hadoop Distributed File System (HDFS). Data needs to be in HDFS to be processed by Hadoop.

MOVING DATA TO HDFS

```
/*************************************************
*  Simple example of moving a file to HDFS       *
*  filename to Hadoop cluster configuration      *
*************************************************/
filename cfg "\\machine\config.xml";;
/* create authdomain in SAS Metadata named "HADOOP" *
 * copy file from my local file system to HDFS     *
 * HDFS location is /user/sas                      *
*************************************************/
proc hadoop options=cfg authdomain=”HADOOP” verbose;
   hdfs copyfromlocal=”c:\public\tmp\hamlet.txt”
        out="/user/sas/hamlet.txt" overwrite;
run;
```

The AUTHDOMAIN option is used. If you have set up AUTHDOMAIN, then you do not need to specify USERNAME= and PASSWORD=.

Once the data is in HDFS, you will want to read that data and process it in Hadoop to take advantage of the parallel processing power of Hadoop.

RUNNING MAPREDUCE IN A HADOOP CLUSTER FROM SAS

PROC HADOOP can submit the sample MapReduce program that is shipped with Hadoop via the following SAS code:

```
proc hadoop
   options = cfg  verbose
    AuthDomain="HADOOP"
   ;
      hdfs delete="/user/sas/outputs";
      mapreduce
         input="/user/sas/hamlet.txt"
         output="/user/sas/outputs"
         jar="C:\Public\tmp\wcount.jar"
         map="org.apache.hadoop.examples.WordCount$TokenizerMapper"
         reduce="org.apache.hadoop.examples.WordCount$IntSumReducer"
         combine="org.apache.hadoop.examples.WordCount$IntSumReducer"
         outputvalue="org.apache.hadoop.io.IntWritable"
         outputkey="org.apache.hadoop.io.Text"
```

```
      ;
run;
```

RUNNING PIG IN A HADOOP CLUSTER FROM SAS

Apache Pig is another technique available to Hadoop users for the simple processing of data. This example does a word count using Pig Latin:

```
filename W2A8SBAK temp ;
data _null_;
file W2A8SBAK;
put "A = load '/user/sas/hamlet.txt'; ";
put "B = foreach A generate flatten(TOKENIZE((chararray)$0)) as word; ";
put "C = Filter B by (word matches 'The');";
put "D = group C by word;";
put "E = foreach D generate COUNT(C), group;";
put "F = store E into '/user/sas/pig_theCount';";
run;


proc hadoop
   options = cfg  verbose
    AuthDomain="HADOOP";
      pig code=W2A8SBAK;
run;
```

The output file is writing in HDFS and shows the words and their occurrences in the source data. You can look at the Hadoop log files to see how many mappers and reducers were run on the data.

USING HDFS TO STORE SAS DATA SETS

What if you have already have SAS data and want to take advantage of the commodity hardware capabilities of your Hadoop cluster? In other words, use HDFS for the storage of your SAS data and run SAS against that data. SAS uses its distributed data storage format with the SAS Scalable Performance Data (SPDE) Engine for Hadoop. In the example below, you can move your SAS data set to HDFS and use SAS to work with that data. Currently, the processing of the data does not take place using MapReduce. This functionality uses HDFS. SAS Scalable Performance Data Engine (SPDE) provides parallel access to partitioned data files from the SAS client.

```
libname testdata spde '/user/dodeca' hdfshost=default;
libname cars base '\\sash\cardata';

proc copy in=cars out=testdata;
select cardata;
run;
```

RUNNING SAS AGAINST YOUR HADOOP DATA

```
proc freq data=testdata.Cardata;
tables Color;
run;
```

The FREQ Procedure

Color				
BEIGE	1584	2.17	1584	2.17
BLACK	7627	10.45	9211	12.62
BLUE	10347	14.18	19558	26.80
BROWN	436	0.60	19994	27.40
GOLD	5231	7.17	25225	34.56
GREEN	3194	4.38	28419	38.94
GREY	7887	10.81	36306	49.75
MAROON	2046	2.80	38352	52.55
NOT AVAIL	94	0.13	38446	52.68
NULL	8	0.01	38454	52.69
ORANGE	415	0.57	38869	53.26
OTHER	242	0.33	39111	53.59
PURPLE	373	0.51	39484	54.10
RED	6257	8.57	45741	62.67
SILVER	14875	20.38	60616	83.05
WHITE	12123	16.61	72739	99.67
YELLOW	244	0.33	72983	100.00

Figure 1. Results from PROC FREQ

ACCESSING HIVE DATA USING SAS/ACCESS INTERFACE TO HADOOP

SAS/ACCESS Interface to Hadoop provides a LIBNAME engine to get to your Hadoop Hive data. Below is a simple example using PROC SQL:

```
/* CTAS - create table as select */
proc sql;
   connect to hadoop (server=duped user=myUserID);

    execute (create table myUserID_store_cnt
                row format delimited fields terminated by '\001'
                stored as textfile

               as
                  select customer_rk, count(*) as total_orders
                    from order_fact
                   group by customer_rk) by hadoop;

   disconnect from hadoop;
quit;

/* simple libname statement */
libname myhdp hadoop server=duped user=myUserID ;

/* Create a SAS data set from Hadoop data */
proc sql;
   create table work.join_test as (
      select c.customer_rk, o.store_id
        from myhdp.customer_dim c
           , myhdp.order_fact o
       where c.customer_rk = o.customer_rk);
```

```
quit;
```

Below is a PROC FREQ example. The Hadoop LIBNAME exposes standard SAS functionality such as PROC FREQ against Hadoop data.

```
/* PROC FREQ example     */
/*   Create a Hive table */
data myhdp.myUserID_class;
   set sashelp.class;
run;

/* Run PROC FREQ on the class table */
proc freq data=myhdp.myUserID_class;
   tables sex * age;
   where age > 9;
   title 'Frequency';
run;
```

LEVERAGING THE POWER OF SAS TO PROCESS YOUR HADOOP DATA ACROSS THE CLUSTER

In this example, SAS Enterprise Miner is used to develop a model for scoring. The result is a score.sas and score.xml files. This functionality can be deployed to a Hadoop cluster for all of the processing to take place in Hadoop.

To run scoring in Hadoop, there are several prerequisites to processing the data. Metadata needs to be known about the input data. The PROC HDMD can be used to register this metadata. Credentials must be provided to connect to the Hadoop cluster using the INDCONN macro variable.

Scoring Accelerator for Hadoop

SAS Enterprise Miner can be used to build models. Using the Score Export node from SAS Enterprise Miner, the scoring code can be exported for scoring to take place on your Hadoop cluster. The code below sets up the connection to the cluster, it publishes the scoring code to HDFS, and it then runs the model.

```
%let modelnm=AL32;
%let scorepath=C:\Public\Hadoop\Sample Files\ALL\AL32;
%let metadir=%str(/user/hadoop/meta);
%let ds2dir=%str(/user/hadoop/ds2);
%let INDCONN=%str(HADOOP_CFG=&config
         USER=&user PASSWORD=&pw);

%indhd_publish_model(  dir=&scorepath.
                     , datastep=score.sas.
                     , xml=score.xml.
                     , modeldir=&ds2dir.
                     , modelname=&modelnm.
                     , action=replace
                     , trace=yes);

%indhd_run_model(
  inmetaname= &metadir./al32.sashdmd
, outdatadir=&datadir./&modelnm.out
, outmetadir=&metadir./&modelnm.out.sashdmd
, scorepgm=&ds2dir./&modelnm./&modelnm..ds2
, forceoverwrite=false
, showproc=yes /* showproc and trace are debug options */
, trace=yes);


/* Select the values from the scoring results on Hadoop*/

proc sql;
```

```
   select em_classification from gridlib.&modelnm.out;
quit;
```

At this point, the scoring algorithm runs through all of the data and provides a score for the data. The processing all takes place in Hadoop. The SAS term “in-database” is a carry-over term from the days of putting the processing power of SAS into databases. SAS embeds the process into the Hadoop cluster so that the cluster does all the work. There is no data movement back to SAS.

SAS provides the ability to embed user-written code into Hadoop. That functionality is provided through the SAS Code Accelerator for Hadoop. This functionality is currently in development and is due to be released in the summer of 2014.

RUNNING USER-WRITTEN SAS CODE

The SAS embedded process technology offers the ability for a user to run SAS code in the cluster and take advantage of the processing power of Hadoop. The first offering from SAS was the SAS Scoring Accelerator for Hadoop. The next offering from SAS (target release date around summer 2014) is the SAS Code Accelerator for Hadoop.

Code Accelerator

The example below takes data and spreads it across the cluster using BY-group processing and uses first/last dot notation to output the result. This technique takes advantage of Hadoop processing power to do all of the calculations in the mapper stage.

```
libname hdptgt hadoop server=&server port=10000 schema=sample user=hive password=hive
config="&hadoop_config_file";

proc ds2 indb=yes;
   thread tpgm / overwrite=yes;
   declare double highscore;
   keep gender highscore;
   retain highscore;
      method run();
         set hdptgt.myinputtable;
         by gender;
         if first.gender then highscore=0;
         if score1 <= 100 and score1>highscore then highscore=score1;
         if last.gender then output;
      end;
   endthread;
   run;

ds2_options trace;

   data hdptgt.myoutputscore(overwrite=yes);
      dcl thread tpgm tdata;
      method run();
         set from tdata;
      end;
   enddata;
   run;

quit;
```

Obs	highscore	gender
1	80	f
2	80	m

Figure 2. Result from the SAS Code Accelerator Program

PROCESSING FILES ON HADOOP TO LOAD SAS LASR ANALYTIC SERVER

The HDMD procedure enables you to register metadata about your HDFS files on Hadoop so that they can be easily processed using SAS.

The example below uses the Hadoop engine to parallel load your file data into the SAS LASR Analytic Server. This same technique can be used whenever you are working with Hadoop file data.

```
/* The HADOOP libname points to your Hadoop cluster */
libname hdp HADOOP
    /* connection options */
    config = &cfgFile
    hdfs_datadir = '/user/demo/files'
    hdfs_tempdir = '/user/demo/tmp'
    hdfs_metadir = '/user/demo/meta'
    hdfs_permdir = '/user/demo/perm';

/* Start up a LASR Analytic Server on port 1314 */
options set=GRIDHOST="servername.demo.sas.com";
options set=GRIDINSTALLLOC="/opt/TKGrid";

proc lasr create port=1314
                path="/tmp" noclass;
                performance nodes=all;
run;

/* Use proc HDMD to create a definition of the movies.txt file - delimited by comma */
/* This proc does not read the data, it just defines the structure of the data */
Proc HDMD name=hdp.movies
format=delimited encoding=utf8
sep = ',' text_qualifier='"'
DATA_FILE='movies.txt'
input_format='org.apache.hadoop.mapred.TextInputFormat'
;
      column userID int;
      column itemId int;
      column rating int;
      column timeID int;
run;
/* Now load the data into the LASR Analytic Server */
/* Parallel load of movies data into LASR */
proc lasr add port=1314   data=hdp.movies; run;

/* terminate the lasr server @ port 1314 */
proc lasr term port=1314; run;
```

SAS DATA DIRECTOR

The SAS Data Director (target release date around summer 2014) is a visual client that allows for processing Hadoop data on the cluster. It provides a point-and-click interface that makes it easy to work with Hadoop data. SAS Data Director is a code generator. Depending on the type of transformation required, different code generation techniques are used. SAS Data Director uses native MapReduce, SAS Code Accelerator for DS2, SAS/ACCESS Interface to Hadoop for Hive queries and Apache Sqoop. Below are a few examples of the type of processing provided by SAS Data Director.

Sqoop Data In and Out of Hadoop

Using SAS Data Director, the user selects the location of the data to copy and the target location. A fast parallel load is then initiated to copy the data from the relational database into Hadoop.

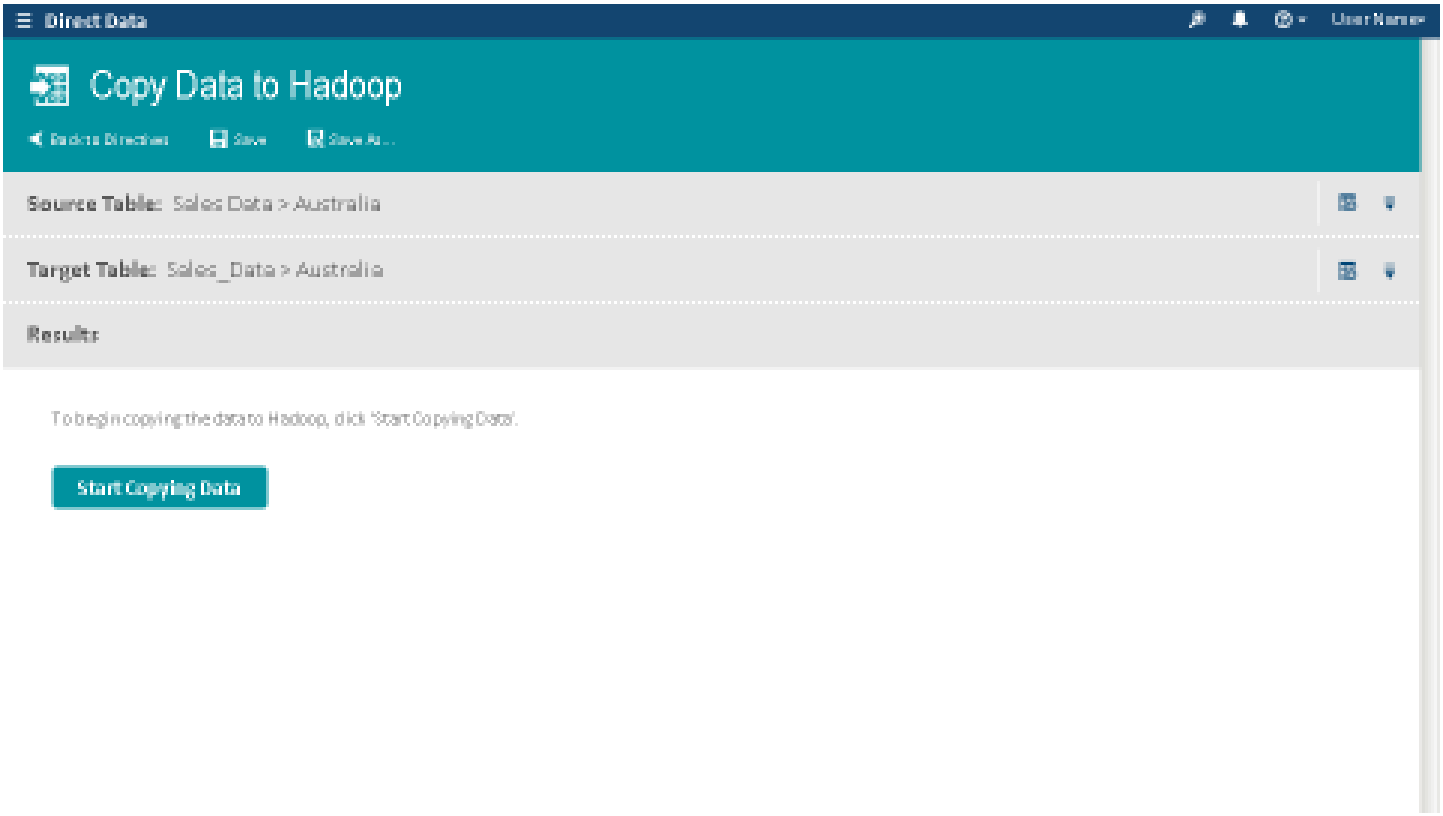

Figure 3. SAS Data Director Copies Data to Hadoop Using Sqoop

SAS Data Director transforms the data on your Hadoop cluster (using filter, subset, join, and sort).

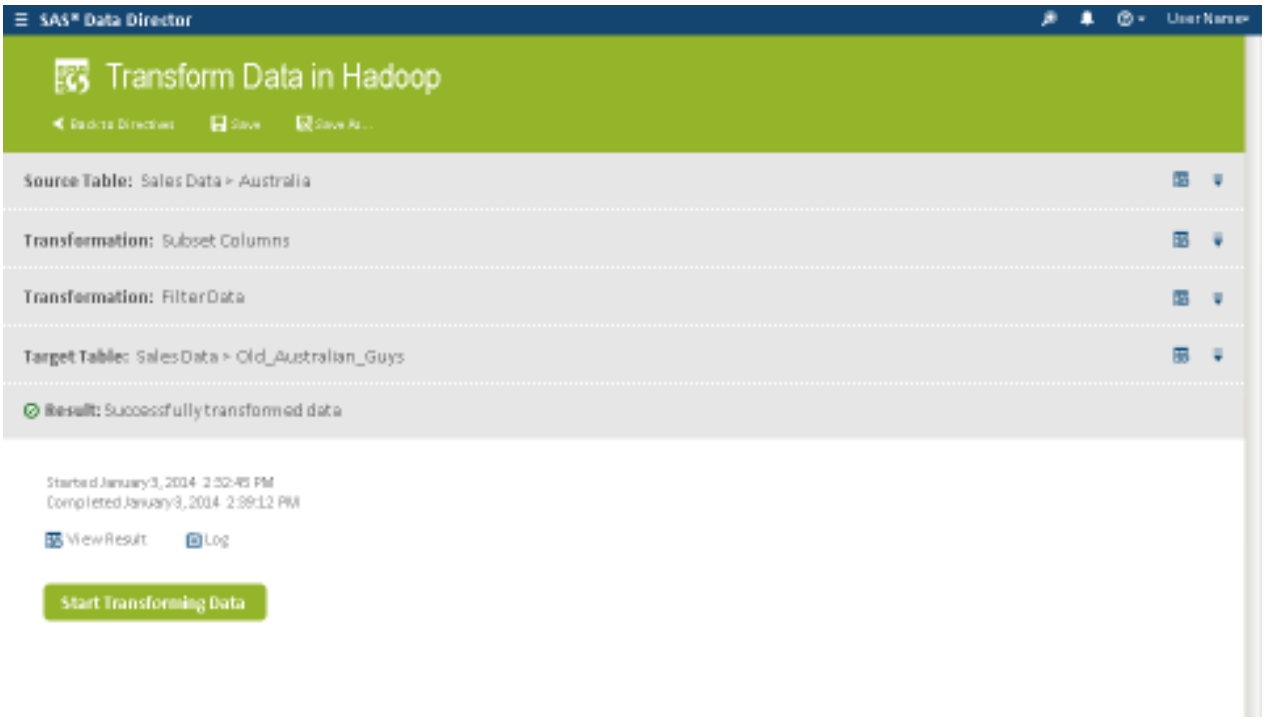

Figure 4. Example of SAS Data Director Transforming Data in Hadoop

Easy Movement of Data to SAS LASR Analytic Server and SAS Visual Analytics Server

Parallel loads into SAS LASR Analytic Server and SAS Visual Analytics Server are easy with SAS Data Director. Specify the source and the LASR target and data is copied into SAS LASR Analytic Server.

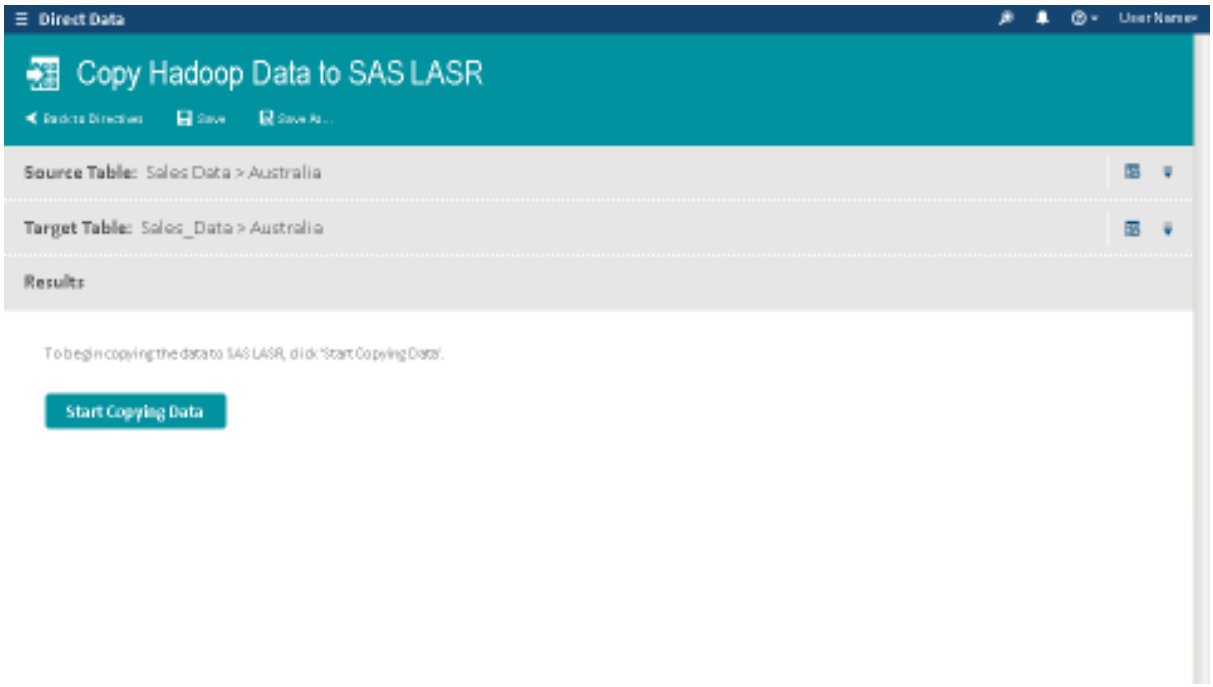

Figure 5. Example of SAS Data Director Copying Hadoop Data to SAS LASR Analytic Server

HELPFUL TECHNIQUES TO WORK WITH HADOOP DATA

The HDFS housekeeping routines that are provided by PROC HADOOP are handy to verify that the directory structure is in place or that old data is cleared out before running your SAS Hadoop job.

When preparing my Hadoop environment for scoring, I relied on PROC HADOOP to get my directory structure in place.

```
proc hadoop
   options = "\\sashq\root\u\sasdsv\dev\Hadoop\cdh431d1_config\core-site.xml"
   USER="&dbuser" PASSWORD="&dbpw"
   ;
     hdfs mkdir="/user/dodeca/meta";
     hdfs mkdir="/user/dodeca/temp";
        hdfs mkdir="/user/dodeca/ds2";
     hdfs copyFromLocal="c:\public\scoring.csv"
         out="/user/dodeca/data/scoring.csv"
      ;
run;
```

CONCLUSION

SAS continues to provide more features for processing data in the Hadoop landscape. Most recently, SAS can take advantage of the processing power of Hadoop with its embedded process technology. SAS will continue to offer more features as the Hadoop ecosystem delivers more functionality. For example, SAS/ACCESS Interface to Impala was released in early 2014.

RECOMMENDED READING

- Secosky, Jason, et al. 2014. “Parallel Data Preparation with the DS2 Programming Language.” *Proceedings of the SAS Global Forum 2014 Conference*. Cary, NC: SAS Institute Inc. Available at http://support.sas.com/resources/papers/proceedings14/SAS329-2014.pdf.
- Rausch, Nancy, et al. 2014. “What’s New in SAS Data Management.” *Proceedings of the SAS Global Forum 2014 Conference*. Cary, NC: SAS Institute Inc. Available at http://support.sas.com/resources/papers/proceedings14/SAS034-2014.pdf.
- Rausch, Nancy, et al. 2013. “Best Practices in SAS Data Management for Big Data.” *Proceedings of the SAS Global Forum 2013 Conference*. Cary, NC: SAS Institute Inc. Available at http://support.sas.com/resources/papers/proceedings13/077-2013.pdf.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Donna De Capite
100 Campus Drive
Cary, NC 27513
SAS Institute Inc.
Work Phone: (919) 677-8000
Fax: (919) 677-4444
E-mail: Donna.DeCapite@sas.com
Web: support.sas.com