

Performance Issues and Solutions: SAS[®] with Hadoop

Anjan Matlapudi and Travis Going
Corporate Medical Informatics
AmeriHealth Caritas Family of Companies
3rd Floor, 200 Stevens Drive, Philadelphia, PA 19113

ABSTRACT

Today many analysts in information technology are facing challenges to work with large amount of data. Most analysts are smart enough to find their way out and write queries using appropriate table join conditions, data mining, index keys and hash object for quick data retrieval. Recently SAS system has collaborated with Hadoop data storage and started providing efficient processing power to SAS[®] analytics.

In this paper we demonstrate the differences in data retrieval with in the SAS and Hadoop environment. In order to test data retrieval time, we used the following code to randomly generate one hundred million observations with character and numeric variables using RANUIN function. DO LOOP=1 to 10e7 will generate ten million records however this code can generate any number of records by changing exponential log. We utilize most commonly used functions and procedure to retrieve records on test dataset and we illustrate real time CPU processing. All PROC SQL and Hadoop queries are tested on SAS 9.4 to record processing time.

INTRODUCTION

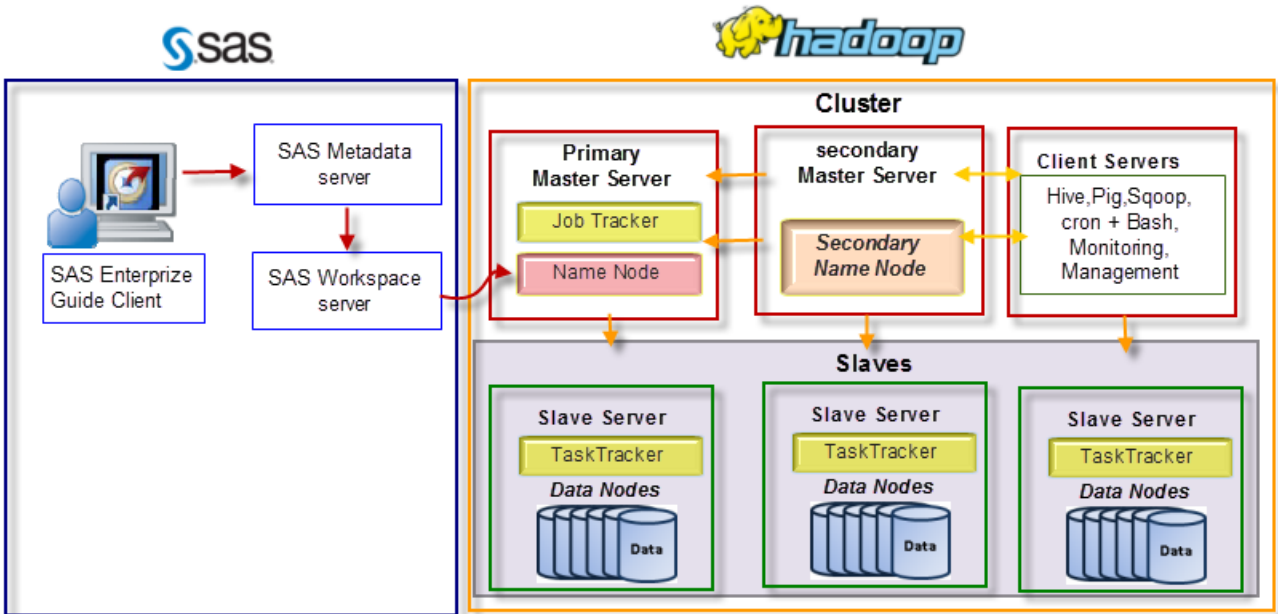
A common challenge many companies are contending solutions to deal with big-data. Challenges include data analysis, data capture, search, transfer and handling unstructured data and optimizations. It is a very common question to many people what Big Data is and how it is related to Hadoop. Big Data, the name itself tells us dealing with big data stored in datasets or tables. Today many corporations including Oracle Corporation, IBM, Microsoft, SAS, SAP, EMC, HP and DELL are exploring Big Data technologies in managing data and analytics. When it comes to Hadoop, the storage part is known as Hadoop Distributed File System (HDFS) while the processing aspect is known as MapReduce. In 2004 Google published a paper on a parallel process model to handle large amount of data using Map Reduce technology. This frame work was very successful and was implemented by an Apache open source called Hadoop. The Hadoop framework is mostly written in Java programming language with some native code in C and command line utilities written as shell scripts.

In this paper we discuss the Hadoop system configuration and software packages in SAS environment and brief description of various modules that can be installed on top or alongside the Hadoop environment. We also discuss Hadoop connecting statements using pass through facility and LIBNAME statements. We further illustrate query processing time using some of the commonly used functions and procedures in SAS 9.4 and Hadoop environment. Finally, we discuss some of the noted performance issues.

SYSTEM CONFIGURATION

The distribution of Hadoop being used Cloudera-CDH 5.3.3. Hadoop cluster consists of NameNode (root node) that acts as master of the system. It maintains directories, files and manages the blocks which are present on the DataNode. The DataNodes (work nodes) are slave nodes which are deployed on each machine and provide the actual storage and serve as read and write request for the clients. Secondary NameNodes are responsible for performing periodic check points in the event of NameNode failures. There are about 6 data nodes parallel processing in SAS metadata server and work as space server in Enterprise Guide client environment.

Below figure showing SAS/ACCESS connects to Hadoop NameNode to send data to be distributed within Hadoop or retrieve data from Hadoop system.



The Hardware configuration includes HP DL380 G9, 2 X 8 Cores CPU (Intel) Xeon E5-2650, 8 X 2TB 6G SAS 7.2K HDD, 2 X 10GigE Intel NICs using 256 GB Ram hard disk space .

HADOOP ECOSYSTEM

Hadoop managed by Apache Foundation is a powerful open-source platform written in java that is capable of processing large number of datasets in a distributed fashion on a cluster of computes using simple programming models. It is designed to scale up from single server to thousands of machines, each offering local computation and storage.








Hadoop Distributed File System (HDFS) designed to store very big datasets by distributing storage and computing across many servers and to stream the data from big datasets at high bandwidth to user applications.

Hadoop YARN is a resource management framework responsible for job scheduling and cluster resource management.

Hadoop MapReduce is a YARN based programming model for parallel processing of large datasets.

Sqoop name comes as the combination of first two letters from **Sq(l)** and last three letters of **(Hd)oop** which serves the most important function to import data from relational database into Hadoop using JDBC.

Beyond HDFS, YARN and Map Reduce, the entire Apache Hadoop platform consists of number of other related projects as well as below mentioned framework.

 <h1>Apache Hadoop Ecosystem</h1>	
Hive SQL Query 	Hive is data warehousing framework to query and manage large data sets stored in Hadoop. It is data warehouse system that facilitates easy data summarization, ad-hoc quires and analysis of large datasets stored in Hadoop compatible file system. It provides a mechanism to structure the data and query the data using an SQL-like language called HaviQL . quires are compiled into MapReduce programs.
Pig Scripting 	Pig platform for data analysis that includes stepwise procedural programming that converts to MapReduce. Hive and Pig provides less complex higher-level programming methods for parallel processing of Hadoop data files.
Apache Impala 	Apache Impala is an open source, native analytic database for Apache Hadoop. Impala is shipped by Cloudera, MapReduce, Oracle and Amazon.
Mahout  Machine Learning	Mahout is a data mining library for Hadoop particularly used for recommended engines. Just like mahout drive the real elephant Hadoop Mahout drives and provide scalable MapReduce algorithms.
APACHE HBASE	HBase is a column-oriented database management system. HBase is not operational without Zookeeper.
Sqoop 	Sqoop is used to import data from relational database into Hadoop using JDBC.
Oozie Workflow 	Oozie is a workflow coordination system to manage Hadoop jobs.

SAS/ACCESS INTERFACE TO HADOOP

SAS/ACCESS to Hadoop provides efficient way to access data stored in Hadoop via HiveQL. In order to connect to Hadoop you can use either LIBNAME or pass-through statement. The below LIBNAME statement is used to connect to Hadoop database. Using LIBNAME you can use PROC COPY in LIBNAME statement in DATA step to store SAS datasets into Hadoop environment.

```

*---LIBNAME statement to connect to Hadoop---*;
LIBNAME HDP hadoop server= 'server name'
port=10000
schema=HDPWRK
user=&USER_ID
subprotocol=hive2
CFG= 'e:\sas\hadoop\config\core-site.xml';
run;

```

SAS log showing Hadoop connectivity

```

NOTE: Libref HDP was successfully assigned as follows:
      Engine:          HADOOP
      Physical Name:  jdbc: hive2://Server Name:10000/HDP

```

SQL procedure pass-through facility enables you to communicate with Hadoop Hive using native HiveQL syntax without leaving SAS your session. The HiveQL queries are passed directly through Hive for processing. HiveQL code syntax is MYSQL however you can use PROC SQL functions as well as most of the SAS functions and procedures to output results.

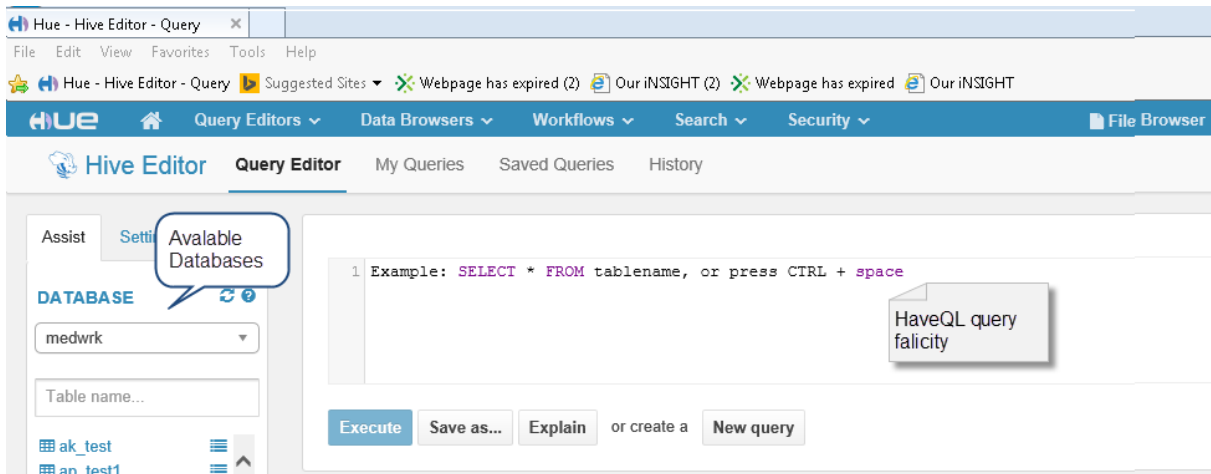
```

*---Pass-through query facility---*;
options sastrace=',,,' sastraceloc=saslog nostsuffix;
Proc Sql;
  Connect to Hadoop
    (server = 'server name'
     subprotocol=hive2 schema=hdp user=&user);
  Create table hdp.hundredmillion AS
  Select * from Connection to Hadoop
    (select name,SSN
     from work.hundredmillion );
  disconnect from hadoop;
quit;

```

HiveQL pass-through macro is provided in the appendix section. In order to create an efficient datasets, we leverage the hive table attributes to define the SAS formats in the macro.

You can execute HiveQL queries using via the edge node or HUE as shown below.



SAS AND HADOOP QUREIS

We demonstrated some tests to compare record retrieval time between SAS 9.4 and the Hadoop environment. In order to test data retrieval time we used the code to randomly generate one hundred million observations dataset have two variables, one character and numeric variables using RANUIN function (Global SAS paper 1786-2014). We used a few commonly used functions to retrieve data from one hundred million records dataset in SAS EG and Hadoop environment and noted CPU time. We noticed that record retrieval time is much faster when we compare running same quires in SAS EG 6.1 environment. Hadoop MapReduce CPU time is in seconds and milliseconds to operate some of the common functions such as SUBSTR, FIND, TRIM, LIKE and some of the authentic functions run against test dataset which contains hundred million records whereas the same queries taking several minutes in SAS EG environment. We also noticed that MapReduce CPU time dropped significantly when run one or many table join conditions in Hadoop environment. Keep in mind that we cannot make correct judgment in terms of record retrial time as the current Hadoop environment and SAS EG environment configured two separate platforms.

PERFORMANCE ISSUES

- Hadoop distribution computing model is more powerful in terms of data processing when compare to most of the current models.
- Organizing and making partitions data into Hive tables, the query runs faster.
- We can use data compress which reduces data storage and can able to store and process huge amounts of data.
- Hadoop distribution computing model is more powerful in terms of data processing when compare to most of the current models.
- You can store any kind of unprocessed as well unstructured data such as text, images and videos, and organize once get loaded in Hadoop.

SOLUTIONS

- The open source framework is cost free and uses commodity hardware to store large quantities of data.
- You add additional storage components by adding additional nodes as data grow day by day. Very little administration is required for maintenance and configuration.
- You can perform complex analytical computations on Hadoop tables with in the data nodes of Hadoop distribution via SAS procedure language.
- SAS Visual Analytics web interface to generate graphical visualizations of data distributions and rations on Hadoop tables pre-loaded into memory within the data nodes of Hadoop distribution.

CONCLSIONS

We hope this paper has provided you with an introduction power of Hadoop with SAS as well as understanding how you can use SAS/ACCESS in Hadoop and get familiarize with some of the modules that you can work with Hadoop environment.

REFERENCES

Sanjay Ghemawat, Howard Gobioff and Shun-tak Leung. The Google File System. [Paper 2003](#).

De Capite. Techniques in Processing Data on Hadoop : [SAS033-2014](#)

Anjan Matlapudi. Tips to Use Character String Functions in Record Lookup: [Global SAS 1786-2014](#)

ACKNOWLEDGMENTS

We would like to acknowledge Thomas Donia, Vice President of Corporate Medical Economics in AmeriHealth Caritas for his full support and encouragement.

We would like to thank Ram Varma, Data Scientist in AmeriHealth Caritas for providing us macro code mentioned in the appendix section.

AmeriHealth Caritas is the nation's leader in the health care solutions with more than 30 years of experience managing care for individuals and families in publicly funded program.

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration. Other brand and product names are trademark of their respective companies.

CONTACT INFORMATION

Anjan Matlapudi

Senior Medical Economics Analyst
AmeriHealth Caritas Family of Companies
email: amatlapudi@amerihealthcaritas.com

Travis Going

Manager Medical Economics
AmeriHealth Caritas Family of Companies
email: travis.goings@amerihealthcaritas.com

APPENDIX - PROGRAMS

```
*****
Purpose: Read in Hadoop table or query
Limitations: Since we are not relying on table statistics for column
attributes, fields with
Transcriptions/notes need to be handled differently. These can be adjusted
for in the first "case statement".
Ideally, any format mapping specifications should rely on table statistics in
Hadoop.
For more information on table stats:
http://www.cloudera.com/documentation/archive/manager/4-x/4-8-6/Cloudera-
Manager-Installation-Guide/cmig\_hive\_table\_stats.html
Macro Parameters:
out_tbl: Output dataset/table in SAS
in_tbl: Hive Table
Schema: Hive DB of interest
Limit: enter '0' if you want a select * OR enter any numeric value to limit
the data
*****;
%let config=
server          ='Provide Server Name'
subprotocol     =hive2
schema         =&schema
user           =&sysuserid;

%macro hdp_intk (out_tbl, in_tbl, schema, limit);
  Proc Sql noprint;
    Connect to Hadoop (&config);
    drop table      hive_map1;
    create table hive_map1 (drop=comment) as
      select *
        from Connection to Hadoop
          (describe &&schema..&&in_tbl);
    disconnect from hadoop;
quit;
proc sql;
  create table hive_maps (drop=comment) as
  select *,
    catx (" ", col_name, sas_f) as sas_sel
  from (select *,
    case
      when data_type = 'int' then "length 7 format 7. informat 7."
      when data_type = 'bigint' then "length 8 format 8. informat 8."
      when data_type = 'double' then "length 7 format 7. informat 7."
      when (col_name like '%dt%' or col_name like '%date%') then
        "length 22 format $22. informat $22."
      else "length 30 format $30. informat $30."
    end as sas_f,
    monotonic () as nvar
  from hive_map1);
select max(nvar) into :num_vars from hive_maps;
%put Number of variables/fields: &num_vars in hive table: &&schema..&&in_tbl;
  select distinct (sas_sel) into: sas_vars SEPARATED by"," from
hive_maps;
quit;
%syndel;
```

```

Proc Sql noprint;
  Connect to Hadoop (&config);
  /*---drop existing out table---*/
  drop table &out_tbl;
  create table &out_tbl (compress=yes) as
    select    &sas_vars
             "&&schema..&&in_tbl." as source
    from Connection to Hadoop
  /*---run PORC SQL statements---*/
    (select *
     from &&schema..&&in_tbl
     %if &limit ne 0 %then
       %do;
         limit &limit
       %end;
    );
  disconnect from hadoop;
quit;
%mend;
/*---passing Macro variable output, input SAS datasets, schema and limit
observations---*/
%hdp_intk (OutSASData,InHiveTble,hdpSchema,100);

```