White Paper

BULKLOADING USING GREENPLUM’S GPFDIST IN A SAS PROGRAM

Using Greenplum’s scatter-gather approach

Abstract

Data delayed is insight lost. The timeliness of data is extremely important for large predictive analytics environments. This paper presents a technique to load in timely manner large data sets by combining the power of Greenplum's scatter-gather (gpfdist) and SAS® bulk loading capabilities. An example on how to configure and test these capabilities is presented.

September 2012

Table of Contents

[Executive summary 4](#_Toc305498730)

[Audience 4](#_Toc305498731)

[Introduction 5](#_Toc305498732)

[Greenplum’s Scatter/Gather 5](#_Toc305498733)

[Parallel Loading 5](#_Toc305498734)

[External Tables 7](#_Toc305498735)

[SAS Access Bulkload 7](#_Toc305498736)

[SAS/ACCESS for Greenplum Engine 7](#_Toc305498737)

[Putting It All Together 9](#_Toc305498738)

[Test Environment 9](#_Toc305498739)

[Configuring the Database 9](#_Toc305498740)

[Creating and Testing the DSN 11](#_Toc305498741)

[Starting and testing gpfdist 12](#_Toc305498742)

[Sample Base SAS Program 16](#_Toc305498743)

[Execute the Test 17](#_Toc305498744)

[Conclusion 20](#_Toc305498745)

[References (optional heading) 20](#_Toc305498746)

[Appendix A 22](#_Toc305498747)

[Appendix B 24](#_Toc305498748)

# Executive summary

Historically, technology and process limitations tended to preclude the possibility of delivering timely[[1]](#footnote-1) data for business intelligence and predictive analysis. Improvements in data Integration tools such as SAS Data Integration Server (DIS) and new paradigms such as Greenplum’s scatter-gather have made a reality what once was a practical impossibility.

This paper provides and example of how to use key functionality from these technologies, namely bulk loading. I will start by walking you through the necessary steps to configure the Greenplum loader (gpfdist). Once the loader is configured, I will show how to test the HTTP server is up and listening trough the right port. Finally, using a simple Base SAS program I will show how to test SAS’ BULKLOAD feature. The program reads a SAS data set from the SASUSER library; creates a table with the same name in the Greenplum database and loads few rows.

Outside of the scope of this paper is to teach you how to program using Base SAS. Also, performance characterization of loads are not discussed in this paper, as it is heavily dependent on hardware, storage and network configurations, as well as, database model and data formats.

The main goal of the paper, is to provide a practical example on how to configure and test interoperability between a SAS program and Greenplum’s scatter-gather loaders using SAS’ bulk load feature to load a sample file.

## Audience

This paper is written for SAS developers who possess a basic understanding of Greenplum or any relational database management system (RDBMS). It is also intended for Greenplum data base administrators (DBA) and consultants that have little or no knowledge about Base SAS programming.

# Introduction

A lot of progress has been achieved in recent years, but the fact remains that having timely data continues to be a challenge. As Greg Papadopoulos, chief technology officer at Sun pointed out:

“The appetite for data collection, storage, and analysis is outstripping Moore’s law”

Meaning that the time required to load and analyze massive data sets is steadily growing.

EMC Greenplum’s scatter-gather streaming provides multiple loading utilities. The gpfdist loader is one of them. This utility provides high performance for data loads. The benefit is that you are guaranteed maximum parallelism while loading data.

On the other hand, SAS bulk loading provides high-performance access to external data sources. When you combine both, gpfdist and SAS bulk load, you are able to insert large data sets into Greenplum tables in the shortest span of time. You can also use bulk loading to execute high-performance SQL queries against external data sources, without first loading those data sources into a Greenplum database. These fast SQL queries enable you to optimize the extraction, transformation, and loading tasks that are common in data warehousing and predictive analytics.

In this paper I will start by giving you some background information about the parallel loading capabilities of Greenplum. Next I will provide and overview of how to use the bulk loading features in a Base SAS program. Using the concepts in those two sections I will walk you through an example to show all the steps to configure and execute the loader using a simple Base SAS program.

# Greenplum’s Scatter/Gather

## Parallel Loading

Greenplum's Scatter/Gather Streaming™ (SGS) technology eliminates the bottlenecks associated to data loading, enabling application like SAS DIS to stream data into the Greenplum database very fast. This technology is intended for loading big data sets that are normally used in large-scale analytics and data warehousing.

The SGS technology manages the flow of data into all nodes of the database. It does not require additional software or systems and takes advantage of the same parallel dataflow engine nodes in Greenplum database.

Figure 1 shows how Greenplum utilizes a parallel everywhere approach to loading. In this approach data flows from one or more source systems to every node of the database without any sequential choke points.

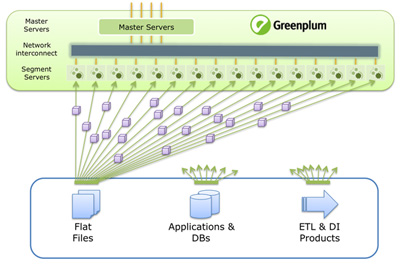


Figure 1

Greenplum’s SGS technology ensures parallelism by scattering data from source systems across 100s or 1000s of parallel streams that simultaneously flow to all nodes of the Greenplum Database. Performance scales with the number of Greenplum Database nodes, and the technology supports both large batch and continuous near-real-time loading patterns with negligible impact on concurrent database operations.

Figure 2 shows how the final gathering and storage of data to disk takes place on all nodes simultaneously, with data automatically partitioned across nodes and optionally compressed. This technology is exposed via a flexible and programmable external table (explained below) interface and a traditional command-line loading interface.

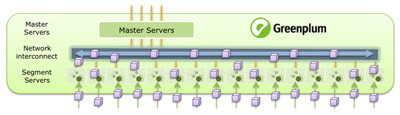


Figure 2

## External Tables

External tables enable you to access data in external sources as if it were in a table in the database. In Greenplum there are two types of external data sources, external tables and Web tables. They have different access methods, external tables contain static data that can be scanned multiple times. The data does not change during queries. Web tables provide access to dynamic data sources as if those sources were regular database tables. Web tables cannot be scanned multiple times. The data can change during the course of a query.

# SAS Access Bulkload

SAS Data Integration tools provide capabilities for enterprise data access and processing across systems and platforms. The key components of SAS Data Integration are Enterprise Data Integration, Data Quality and Data Access Engines. SAS/ACCESS for Greenplum is one of the data access engines.

## SAS/ACCESS for Greenplum Engine

SAS/ACCESS for Greenplum is an interface with which you can interact with data in the database from within SAS. SAS/ACCESS provides two methods for accessing relational DBMS data.

The first one is the LIBNAME statement to assign SAS librefs to DBMS objects such as schemas and databases. After you associate a database with a libref, you can use a SAS two-level name to specify any table or view in the database. You can then work with the table or view as you would with a SAS data set.

The second method is the SQL pass-through to interact with a data source using its native SQL syntax without leaving your SAS session. SQL statements are passed directly to the data source for processing.

In this paper I will use the first approach, namely using a libname statement to establish a connection to the database (see SAS Interoperability with EMC Greenplum whitepaper of a step-by-step connectivity settings and test).

Here is an example on how to connect to the database using the libname statement. It creates a library reference (or libref) named mydblib. It also specifies the engine name, in this case greenplm. Then it provides connectivity and credentials information such as: host name, port number where Greenplum is listening, the database and schema names and user credentials.

libname mydblib greenplm host=<host name> port=<port number> schema=<schema name> db=gpadmin user=gpadmin password=gpadmin;

Once connectivity has been established and the library has been created you can create new data sets (tables) and populate them using the bulk load utility.

The purpose of bulk loading is to provide the highest possible load performance. The ability to exploit Greenplum’s load utilities has huge performance implications when loading data in a large predictive analytics environment. The SAS/ACCESS for Greenplum engine allows you to easily invoke these native load extensions.

It is important to mention that bulk load facilities are not transactional. That is, they do not use programmatic SQL insert statements to load data. Rather, the input data set is copied as a unit to the DBMS table. As a result, error conditions behave differently under bulk load, i.e., no rollbacks are issued. Greenplum’s bulk loader can be used for initial loads of empty tables or to append rows to existing tables. It can be used for small (dynamic) or large (static) datasets.

Next section shows a step-by-step example on how to configure and test the gpfdist loader. It also explains how to use bulk loading.

# Putting It All Together

In this section I will walk you through an example on how to configure and test the concepts presented in previous sections. I will start by describing the test environment. Next, the database configuration settings required to allow remote connections and user privileges. Testing gpfdist is next, making sure the HTTP server is up and listening. A step-by-step example on how to perform the test is presented using a simple Base SAS program.

## Test Environment

Figure 3 depicts the test environment. The Greenplum database is installed in a Red Hat Linux VMWare Virtual Machine (host name is gpdb). SAS for Windows, SAS/ACCESS for Greenplum, the ODBC and the gpfdist loader are installed in the host machine (host name SASENV), in this example a Windows XP environment.

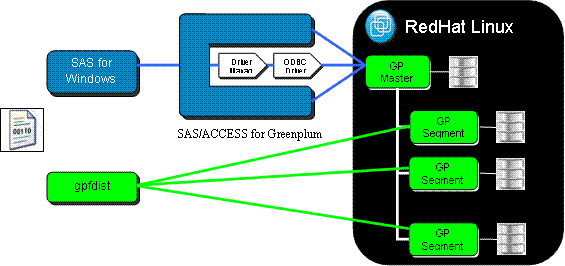


Figure 3

## Configuring the Database

First you need to install and configure your database to accept remote client connections. Follow these steps to enable remote client connections:

Edit the postgresql.conf file of the Greenplum master instance. Modify the *“listen\_addresses”* parameter to include the TCP/IP addresses or hostnames of the remote clients. The special entry *“\*”* corresponds to all hosts. For example:

listen\_addresses = 'localhost', 'remotehost1', 'remotehost2'

listen\_addresses = '\*‘

Also you need to enable create external tables capabilities for http/gpfdist to users other than the Greenplum administrator. Set the following parameter to on (it is commented and off by default)

gp\_external\_grant\_privileges = on

Save and close the postgresql.conf file.

Next, open the pg\_hba.conf file (located in the same directory) and validate the configuration to allow connections from users to the database(s) using the authentication method you want. For example to open access to everybody set it to these values:

local all all trust

#open to everybody

host all all 0.0.0.0/0 trust

Restart Greenplum after making these configuration changes. Make sure that the databases and roles you are using to connect exist in the system and that the roles have the correct privileges to the database objects.

GRANT ALL PRIVILEGES ON DATABASE gpadmin TO sasdemo;

GRANT ALL PRIVILEGES ON SCHEMA public TO sasdemo WITH GRANT OPTION;

GRANT ALL PRIVILEGES ON emp TO sasdemo WITH GRANT OPTION;

Next you need to configure the ODBC driver Database Source Name or DSN.

## Creating and Testing the DSN

Follow these steps to create and test the DSN in the Windows host:

1. Select *Start* ->*Settings* -> *Control Pannel*.
2. In the Control Pannel select *Administrative Tools* -> *Data Sources ODBC*.
3. Click on the *System DSN* tab and then click *Add*.
4. Select the SAS ACCESS to Greenplum driver.

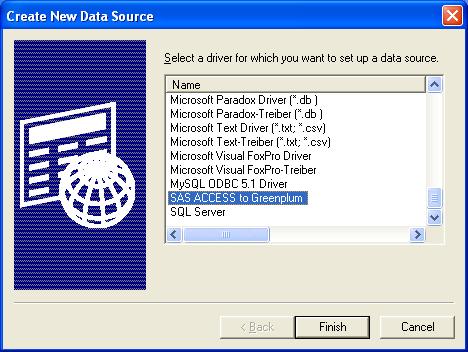


Figure 4

Configure the driver by providing the Data Source Name and a description. Also you need to specify the host name or IP address; the port number where Greenplum is listening and the database name. Click the Test Connect button to test the connection.

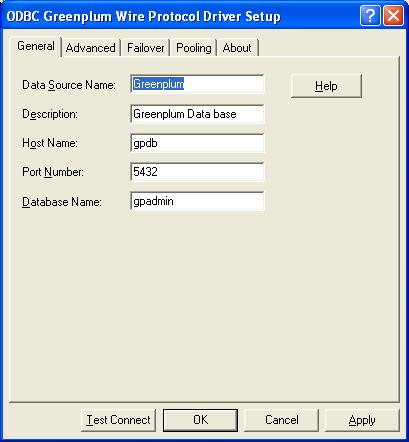


Figure 5

Provide the credentials to connect, that is, user name and password and click OK.

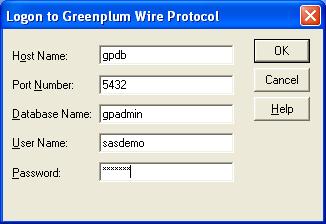


Figure 6

A message with the connection status will be displayed.

## Starting and testing gpfdist

Follow these steps to test gpfdist is working properly:

1. Make sure you can ping the server where Greenplum is installed

$ ping <host/IP Address>

1. Set the following environment variables to save some typing on the command-line:

PGDATABASE — the name of the default Greenplum database to connect to.

PGHOST — the Greenplum Database master host name or IP address.

PGPORT — the port number that the Greenplum master instance (postmaster process) is running on.

PGUSER — the default database role name login.

GPLOAD\_HOME - set the value of the variable to the directory that contains the external tables to be loaded. This is a required variable.

The steps to add a new system environment variable on Windows XP are:

* In Windows Explorer, go to the Control Panel.
* Double-click the System icon.
* On the Advanced tab, click Environment Variables (bottom).
* Click New.
* Define the new environment variable. For example:

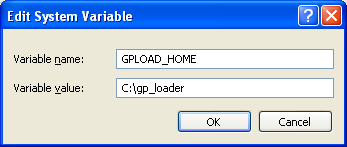


Figure 7

1. Check availability of the ports you intend to use for gpfdist. You can use netstat to validate if the port is free, for example:

$ netstat –a

Or, to check which ports are used that start with the number 80 (if you have cygwin installed) type:

$ netstat –a | grep 80

If the port is already in use gpfdist will not start and an error will be displayed.

1. Next, use a command line window to start the loader, for example.

C:\>gpfdist -d %GPLOAD\_HOME% -p 8085 -l %GPLOAD\_HOME%\gpfdist.log

The -d flag is used to provide the directory from which gpfdist.exe will serve files. If not specified, it defaults to the current directory. The -l log\_file flag provides a fully qualified path and log file name where standard output messages are to be logged. The HTTP port is set using the -p flag, this is the port on which gpfdist.exe will serve files. It defaults to port 8080.

If you have cygwin installed you can execute the command in the background by adding an ampersand at the end of the command.

$ gpfdist –d $GPLOAD\_HOME –p 8085 –l $GPLOAD\_HOME\gpfdist.log &

To verify that gpfdist is listening, execute the netstat command as explained above. The port number you specified in the –p flag should have the status LISTENING.

1. Now that the loader has been started, create a small sample pipe “|” delimited file with a few records, for example:

308|765|1

1534|765|1

1324|765|1

……

1. Use wget to test the HTTP server. GNU Wget is a free software package for retrieving files using HTTP, HTTPS and FTP. It is a non-interactive command line tool, so it may easily be called from scripts, cron jobs, terminals without X-Windows support, etc. You can download wget from <http://sourceforge.net/projects/gnuwin32/files>. Once installed you can test connectivity by typing the following command:

$ wget <http://hostname:8085/test.dat>

Your output should look something like this:

SYSTEM\_WGETRC = c:/progra~1/wget/etc/wgetrc

syswgetrc = C:\Program Files\GnuWin32/etc/wgetrc

--2011-09-28 17:33:31-- http://sasenv:8085/test.dat

Resolving sasenv... 192.168.238.134

Connecting to sasenv|192.168.238.134|:8085... connected.

HTTP request sent, awaiting response... 200 ok

Length: unspecified [text/plain]

Saving to: `test.dat.1'

[ <=> ] 27,501,177 2.26M/s in 11s

2011-09-28 17:33:42 (2.49 MB/s) - `test.dat.1' saved [27501177]

Look for the Connecting and HTTP messages, they should read *“connected”* and *“ok”.*

1. To stop gpfdist just close the *cmd* window or, if you used cygwing you can use the kill command to terminate execution of gpfdist.

$ kill -9 <process id>

Now we are ready to load some data from a Base SAS program.

## Sample Base SAS Program

A complete listing of the program is in Appendix A. First you need to assign the SAS/ACCESS engine name for Greenplum the name is GREENPLM.

%let dbms=greenplm;

Then assign the connection options for GREENPLM to the CONNOPT variable.

%let CONNOPT=%str(

host=gpdb /\*your hostname\*/

port=5432 /\*GPDB port number\*/

db=gpadmin /\*your database name\*/

user=sasdemo /\*your user for GPDB\*/

pwd=gpadmin /\*your user password\*/

);

Execute the libname statement with the above options

libname mydblib &dbms &CONNOPT;

The following procedure creates the *emp* table in the Greenplum database (SAS libname mydblib). The table schema is identical to the fields in the *emp* data set contained in the SASUSER library.

proc sql;

create table mydblib.emp (

bulkload=YES

bl\_host=SASENV /\*replace with the hostname where\*/

/\* gpfdist is running \*/

bl\_port='8085'

bl\_protocol='gpfdist')

as select \* from Sasuser.emp;

quit;

To disconnect and clear the libraries execute this statement.

libname mydblib CLEAR;

## Execute the Test

Open SAS for Windows and copy the program in Appendix B to create and populate the emp data set. The first step is to create the data set in the SASUSER library. Start by defining the data set layout:

data new;

input @1 fn $8.

@9 ln $10.

@19 id $6.

@25 hd $10.

@35 sa 8.

@43 de $4.

@47 ex $2.

@49 intrst $5.;

The first column in the *input* clause is the starting position of the column value. The second column is the field name and the third column is the size and data type. Add some records (observations) using the *cards* clause:

cards;

George Woltman E001271982-08-0753500 D1010100000

Adam Smith E635351988-01-1518000 D2020000000

David McClellan E042421982-07-2741500 D1010100000

...

When you execute this section of the program, it will produce the following NOTE in the log:

NOTE: The data set WORK.NEW has 17 observations and 8 variables.

NOTE: DATA statement used (Total process time):

real time 0.10 seconds

cpu time 0.00 seconds

The WORK.NEW is a temporary work space that SAS creates. There should be seventeen observations. The next step is to create and populate the *emp* data set:

data Sasuser.emp;

set new;

run;

After executing this section of the program the following notes will be displayed in the log:

NOTE: There were 17 observations read from the data set WORK.NEW.

NOTE: The data set SASUSER.EMP has 17 observations and 8 variables.

NOTE: DATA statement used (Total process time):

real time 0.03 seconds

cpu time 0.01 seconds

At this point the emp data set has been created in the Sasuser library. In SAS for Windows Explorer panel (left side) double click on the Sasuser library. Your screen should look similar to this:

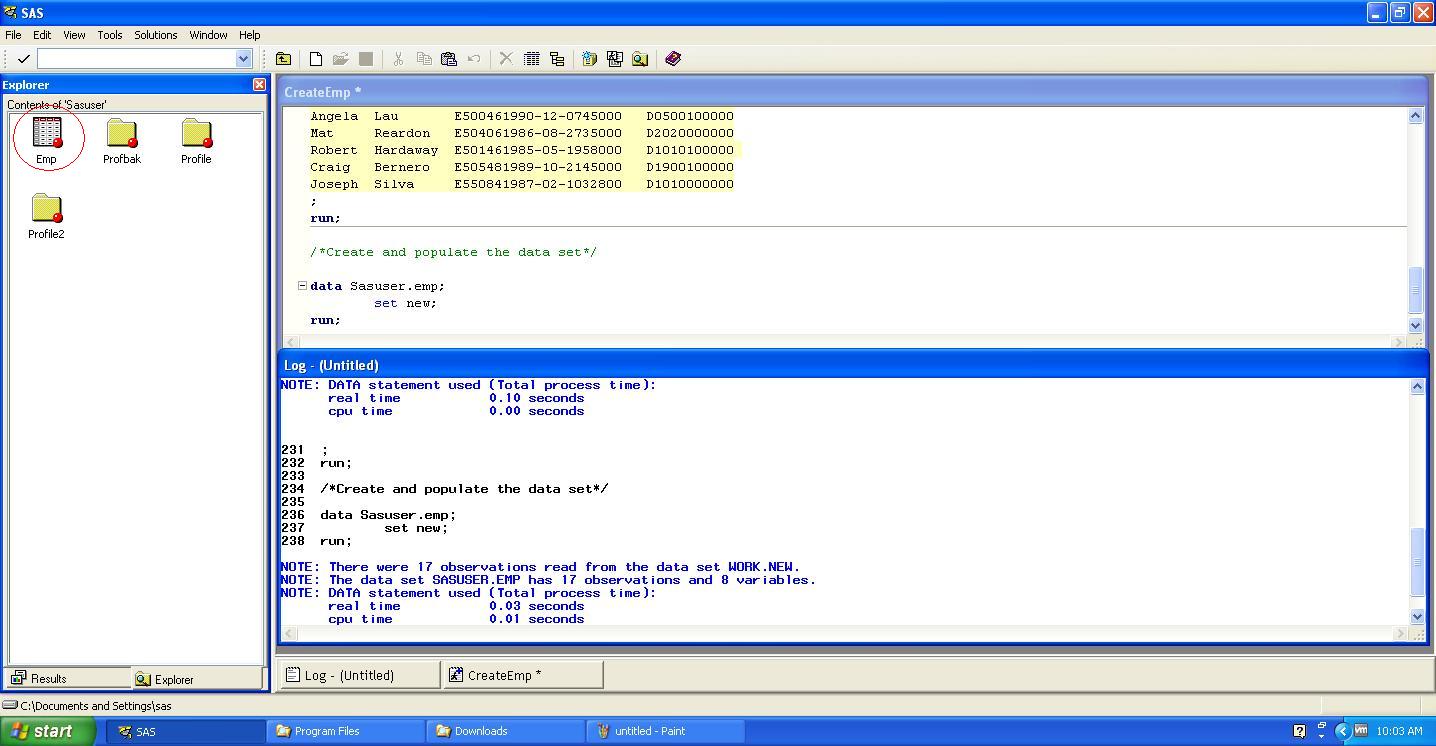


Figure 8

The emp data set should have been created. Click backspace to return to the Active Libraries pane.

To verify the table emp DOES NOT exist in the Greenplum database, create a new program in SAS for Windows (Ctrl+N). Type the following libname statement:

libname testlib greenplm host=<hostname> port=<port number> schema=<schema> db=<database name> user=<user id> password=<password>;

Where:

* <hostname> is the name (or IP address) where the Greenplum database is running
* <port number> is the port where the Greenplum database is listening, by default 5432.
* <schema> is the schema name in the database where you want to create the table
* <database name> is the database name you want to use
* <user id> is the user id to access the database
* <password> is the user’s password.

Double click on the *testlib* library. If the emp table exists you can either drop it (if it is yours) or create a new table with a different name. If you do the latter, make sure to change the programs in the appendices with the new table name. To drop the table, right click on the emp table and select delete. Once you have validated that the table does not exists, you can clear the library.

libname testlib CLEAR;

Execute the CreateDataSet.sas program and review the log for errors. The emp table should appear in the Explorer panel. Double click on it to display the observations that were loaded.



Figure 9

The example above shows how to use the bulk loading capability from a Base SAS program. Data Integration Studio (DIS) is a tool specifically designed to help simplify the ETL process. The idea behind the tool is that the various steps in an ETL process can be packaged into predefined units. These units of execution can be strung together, in a visual manner, to create a job. Jobs are then scheduled and automated. Each code unit can either use DIS generated code, or code provided by the user.

In the example below, the table loader transformation is used to read a source table (orders) from a MySQL database and write it to the to a target table (orders) in a Greenplum database.

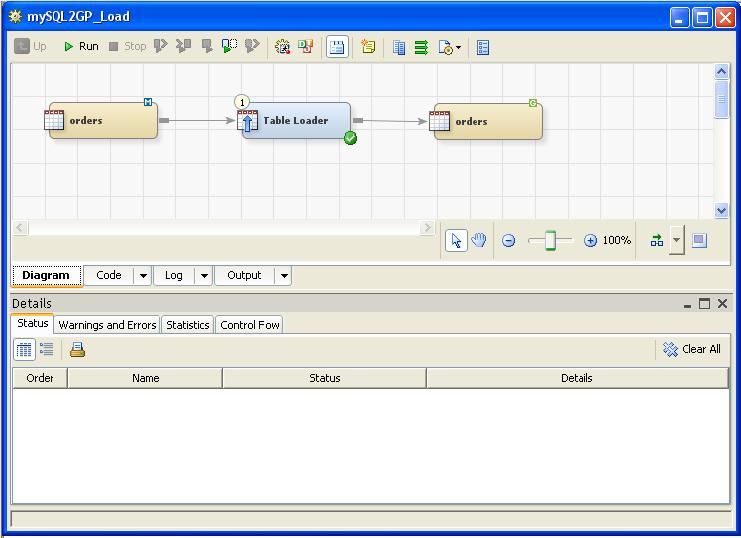


Figure 10

You define the table mappings in the loader transformation. A mapping is the ability to create a relationship between a source and target column as shown below.

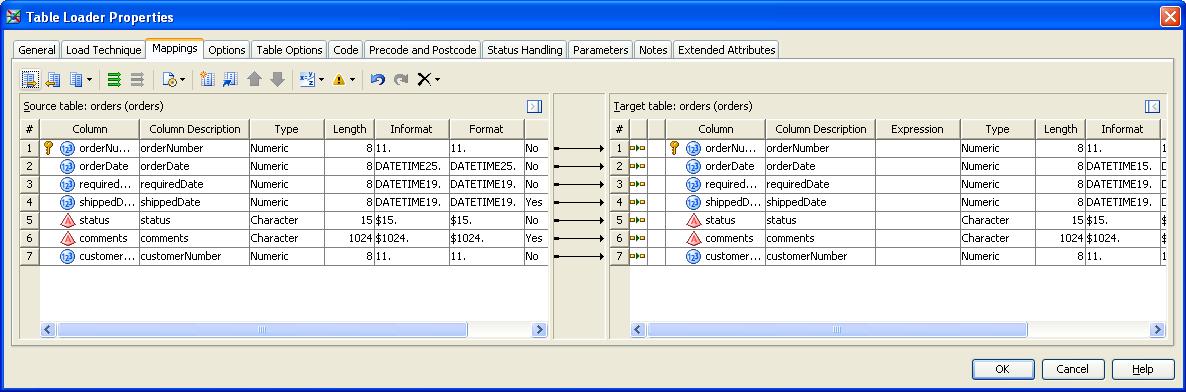


Figure 11

You can specify the BULKLOAD option to load on an individual table level by using the data set option. This data set option applies only to the data set on which it is specified, and it remains in effect for the duration of the DATA step or procedure. The screenshot below shows the minimum parameters required to use the Greenplum bulk loader.

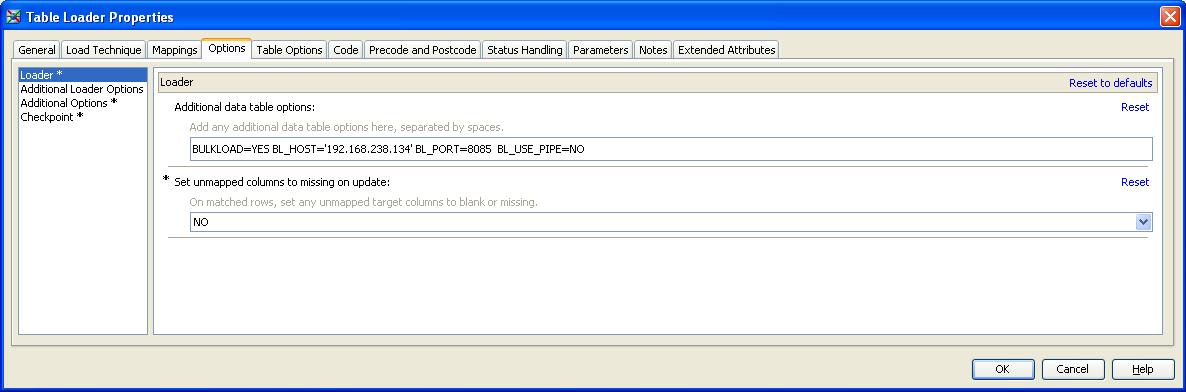


Figure 12

In this section, all the necessary steps to configure and test the bulk loading capabilities from a Base SAS program and a DIS job were covered.

# Conclusion

In this paper we have demonstrated two complementing technologies, namely SAS Bulk Loading capabilities and Greenplum’s scatter-gather streaming, to increase significantly the timeliness of data.

Bulk loading is a powerful feature to speed-up the process of building warehouses and processing data. Combining SAS’ Bulk Loading with Greenplum's Scatter/Gather Streaming™ technology eliminates the bottlenecks associated to data loading, enabling SAS applications to stream data into the Greenplum database very fast.

In concluding, SAS and Greenplum enable customers to gain timely insight and value from their data by providing fast loading and processing capabilities. In other words, timely data is insight gained.

# References

1. Greenplum® Database 4.0 Administrator Guide P/N: 300-011-538 Rev: A02
2. Greenplum Database 4.0 Greenplum 4.0 Loaders for Windows
3. SAS/ACCESS® 9.3 for Relational Databases Reference
4. SAS 9.3 Language Reference: Concepts
5. Wget: http://sourceforge.net/projects/gnuwin32/files/
6. Cygwin: http://www.cygwin.com/

# Appendix A

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* NAME: CreateDataSet.SAS \*/

/\* TITLE: Sample code for creating the emp data set in SASUSER. \*/

/\* PRODUCT: SAS/ACCESS Software for Relational Databases \*/

/\* SYSTEM: z/OS, UNIX, WINDOWS \*/

/\* DBMS: Greenplum DB \*/

/\* REF: SAS/ACCESS Software For Relational Databases: \*/

/\* Reference, Version 9 \*/

/\* NOTE: Replace dbms engine name assignment to greenplm \*/

/\* and provide specific connection information. \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* included for Year-2000 compliance \*/

options yearcutoff=1925 ls=120 nodate;

options sastrace=on;

options SASTRACE=',,,d' SASTRACELOC=SASLOG NOSTSUFFIX;

/\* dbms(engine) name assignment \*/

%let dbms=greenplm; /\* for example, GREENPLM \*/

/\* assign connection options - for GREENPLM, you can remove the \*/

/\* connections options uid= pwd= and dsn= completely \*/

/\* allowing the default Authid and Database to be used \*/

/\*%let CONNOPT=%str(uid=gpadmin

pwd=gpadmin

dsn=Greenplum

); \*/

%let CONNOPT=%str(host=gpdb

port=5432

db=gpadmin

user=sasdemo

pwd=gpadmin

);

/\* edit the following statement to include the path \*/

/\* to a local directory on the host where you will be \*/

/\* executing SAS; this will be used as a SAS library to\*/

/\* store SAS data sets and PROC SQL Views \*/

/\*libname samples base 'C:\cygwin\home\infana\scripts\sas'; \*/

/\* issue libname statement with the above options \*/

libname mydblib &dbms &CONNOPT;

proc sql;

create table mydblib.emp

(bulkload=YES

bl\_host=SASENV

bl\_port='8085'

bl\_protocol='gpfdist')

as select \* from Sasuser.emp;

quit;

libname mydblib CLEAR;

# Appendix B

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* NAME: CreateEmp.SAS \*/

/\* TITLE: Sample code for testing BULKLOAD statement. \*/

/\* PRODUCT: SAS/ACCESS Software for Relational Databases \*/

/\* SYSTEM: z/OS, UNIX, WINDOWS \*/

/\* DBMS: Greenplum DB \*/

/\* REF: SAS/ACCESS Software For Relational Databases: \*/

/\* Reference, Version 9 \*/

/\* NOTE: Replace dbms engine name assignment to greenplm \*/

/\* and provide specific connection information. \*/

/\* \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* included for Year-2000 compliance \*/

options yearcutoff=1925 ls=120 nodate;

options sastrace=on;

options SASTRACE=',,,d' SASTRACELOC=SASLOG NOSTSUFFIX;

/\* Data Set definition column position, name and size\*/

data new;

input @1 fn $8.

@9 ln $10.

@19 id $6.

@25 hd $10.

@35 sa 8.

@43 de $4.

@47 ex $2.

@49 intrst $5.;

cards;

George Woltman E001271982-08-0753500 D101 100000

Adam Smith E635351988-01-1518000 D202 000000

David McClellan E042421982-07-2741500 D101 100000

Rich Holcomb E012341983-06-0149500 D202 100000

Nathan Adams E412981988-02-1521900 D050 000000

Richard Potter E431281986-04-1215900 D101 000000

Kim Arlich E100011985-07-3057000 D190 100000

Tyler Bennett E102971977-06-0132000 D101 100000

John Rappl E214371987-07-1547000 D050 100000

David Motsinger E270021985-05-0519250 D202 000000

Tim Sampair E030331987-12-0227000 D101 100000

Timothy Grove E163981985-01-2129900 D190 100000

Angela Lau E500461990-12-0745000 D050 100000

Mat Reardon E504061986-08-2735000 D202 000000

Robert Hardaway E501461985-05-1958000 D101 100000

Craig Bernero E505481989-10-2145000 D190 100000

Joseph Silva E550841987-02-1032800 D101 000000

;

run;

/\*Create and populate the data set\*/

data Sasuser.emp;

set new;

run;

1. at the right or an opportune or appropriate time [↑](#footnote-ref-1)