Paper 839-2017

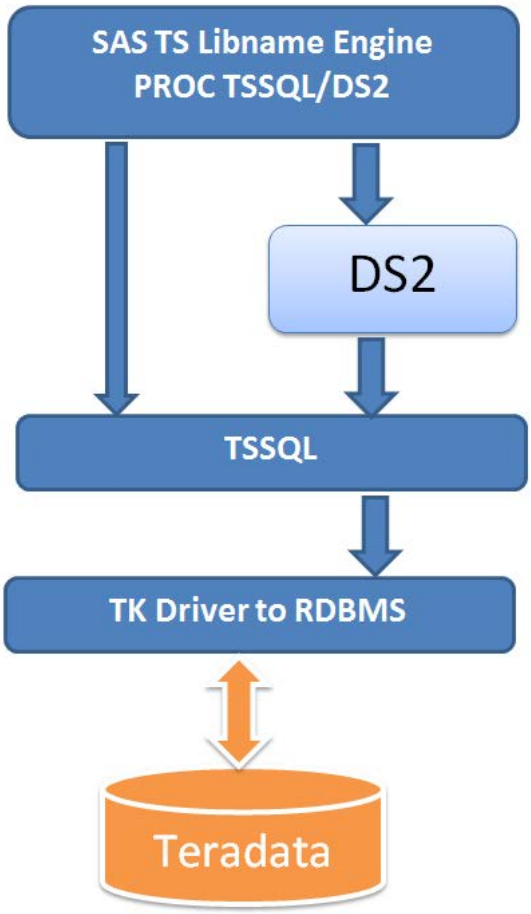
Accelerate Your Data Prep with SASÂ® Code Accelerator

Paul Segal Teradata Corporation.

DS2 OVERVIEW

The SAS DS2 language originated from an effort to enhance the SAS Data Step. DS2 syntax overlaps significantly with Data Step and DS2 include additional data types, ANSI SQL types, programming structure elements, user defined methods and packages. Data sources for DS2 are expanded to include several RDBMSs, hence, row, column and table are used to describe data elements as opposed to observation, variable and data set in Data Step. DS2 programs can be submitted using SAS/ACCESS LIBNAME connection, through the new language interface delivered in SAS Base procedure PROC DS2.

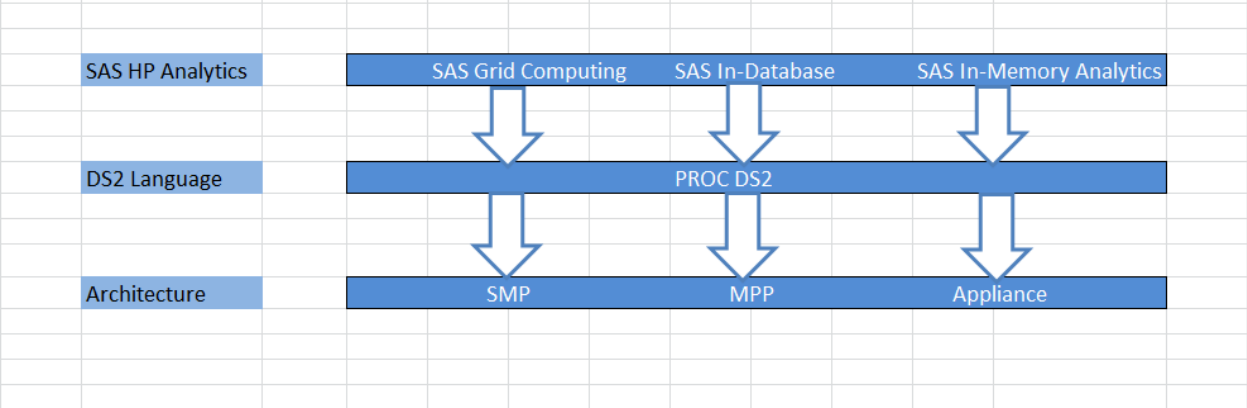
The diagram below shows the basic architecture which illustrates how SAS DS2 collaborates with SAS TK Data Access component and Teradata RDBMS. The Base SAS language interface to SAS Table Server (TS) uses the TS LIBNAME engine and TS SQL procedure. It can also use the threaded procedural language DS2 as an additional feature. TS SQL interface to the data source through the TK drivers, in this particular case SAS/Access to Teradata.



A DS2 program is a list of declare statements followed by list of method definitions. Declare statements allocate memory for the variable and define the data type for that variable. Methods are program units used to group related statements in one syntactically identifiable location. PROC DSTRANS is used to translate Data Step code into DS2 so that the program can be run by PROC DS2. For more information about the DS2 language, please refer

to “SAS DS2 Language Reference” which is available on support.sas.com.

SAS users are able to type and run PROC DS2 code from SAS and take advantage of the In-Database processing capacity of Teradata MPP. In fact DS2 Code runs outside of MVA SAS in the DataFlux Federated Server and in Grid computing environments such as the in-database SAS Embedded Process and the High Performance Analytics Grid.



SAS EP USING SAS DS2

SAS DS2 code can be submitted to the SAS Embedded Processes (EP) from any of the SAS tools to take advantage of existing Teradata processing capacity. The SAS EPs are installed and run on each node. DS2 IP uses its thread program framework to execute DS2 code in parallel inside Teradata and then optionally retrieve the thread results for final aggregation in the second stage of DS2 execution in SAS. DS2 IP will be launched when the input data set for thread program resides in Teradata when the SAS EP is deployed. If there is no EP, the thread programs will be running on the client machine outside of the database. When a DS2 IP program is run, SAS EP processes corresponding to each active AMP will be created to take advantage of the parallelism on both Teradata.

DS2 Implicit Pass-through (DS2 IP) provides a way to run PROC DS2 program from SAS in which DS2 IP publishes to Teradata RDBMS automatically, and then execute DS2 program in parallel within SAS EP installed on each node of the MPP system. Each DS2 request will start a SAS EP processing thread corresponding to each AMP. Each DS2 thread running in the SAS EP can only access and process its own partition of data from the corresponding AMP. The data will then be gathered in one place by PROC DS2 to do the final aggregation/processing.

DS2 EXECUTION IN TERADATA

DS2 is ideally suited to operate on data that may be ‘partitioned”, such as by group processing, transpositions, row independent operations such as scoring and some types of cpu intensive operations.

At the very basic, when executing DS2 in Teradata via the EP mechanism, there data is ‘fed’ into the EP threads in from the Teradata AMP. There is no need to specify the number of threads as the SAS Code Accelerator in conjunction with the EP, instantiates one thread per AMP on the Teradata system (over riding any manual thread number specification).

SIMPLE EXAMPLE OF SAS CODE ACCELERATOR

This example reads data from the table emp_donations from the predefined libname tdlib, and just calculates the totals.

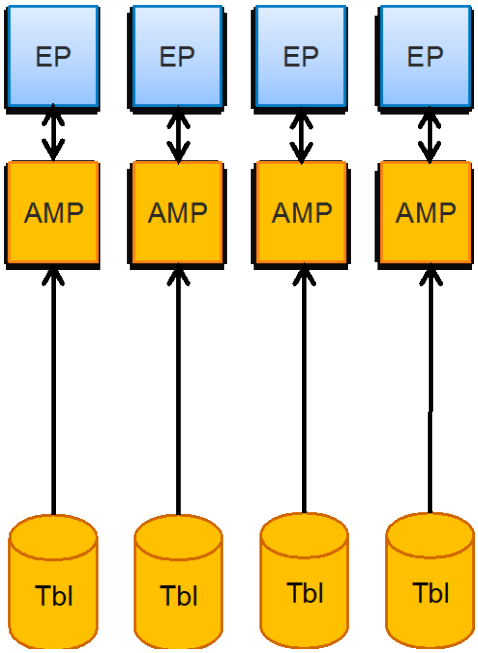
```
Libname tdlib TERADATA server='10.0.0.1' user=me password=****
database=mydb;

proc ds2 ds2accel=yes;
thread compute;
  method run();
    set tdlib.emp_donations;
```

```
    total = sum(jan--dec);
  end;
endthread;
data tdlib.totals;
  dcl thread compute t;
  method run();

  end;
enddata;
```

The below diagram show how the data gets to the EP's from the disk



MORE COMPLEX EXAMPLE

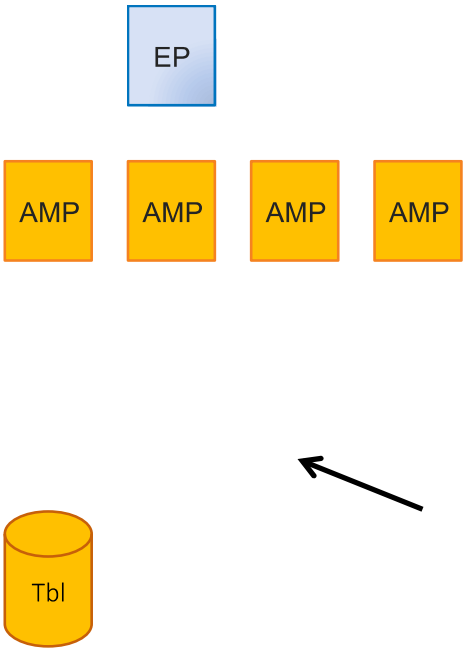
This example reads the data in (as per the other example), however in this case the total is generated for each region (by group processing)

```
proc ds2 dsaccel=yes ;
thread work.compute/overwite=yes;

  method run();
/* read data in from Teradata table */
    set tdlib.emp_donations;
    by region;
    if first.region then total = 0;
    total + sum(jan--dec);
    if last.region then output;
  end;
endthread;
data tdlib.totals;
  dcl thread compute t;
  method run();
```

```
  end;
enddata;
```

The below diagram shows how the data is passed to the EP to execute the DS2 shown above, the difference from the previous example is that the data is sorted (in parallel) by region before being passed into the EP.

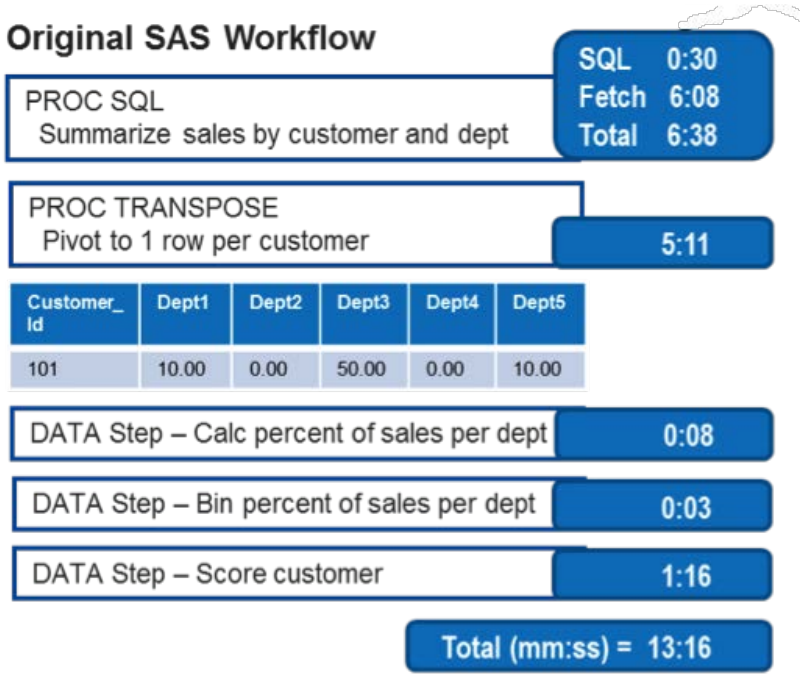


CASE STUDY

A retail customer had an existing SAS process (using PROC SQL , PROC TRANSPOSE and DATA steps).

The data consisted of 1 billion rows of transactions, for 5 million customers over 20 departments, and the process started off by rolling the data up to the 5 million customers, then did some basic calculations to calculate various metrics. The roll up was undertaken via PROC SQL and PROC TRANSPOSE, and the calculations were in DATA steps.

The following show the timings for each step:



As can be seen, the majority of the time taken up was with the extraction from the database via the PROC SQL and then the subsequent PROC TRANSPOSE (together a substantial 90% of the run time).

By rewriting this in DS2 and utilizing the SAS Code Accelerator, we substantially reduced the run time).

Step	Original	DS2 Code Accelerator
SQL Processing	0:30	0:30
SQL Fetch	6:08	
Transpose	5:11	
Datastep - Percentage	0:08	
Dataset - Binning	0:03	
Dataset – Scoring	1:16	
DS2 – Code Accelerator		0:12
Total Run Times	**13:16**	**0:42**

So as may be seen, the run time decreased from 13 minutes and 16 seconds down to 42 seconds.

CONTACT INFORMATION

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

Paul Segal
Teradata Corporation
paul.segal@teradata.com