

SCSUG-2016

## Key Reasons for SAS® Data Set Size Difference by SAS® Grid Migration

Prasoon Sangwan, Piyush Singh, Tanuj Gupta  
TATA Consultancy Services Ltd.

### ABSTRACT

When SAS users move from traditional departmental server to Grid environment they face many challenges with new SAS® Grid Platform settings. This paper describes one of such issue as “SAS Data Set Size” difference from non-Grid SAS server to SAS® Grid Platform. This paper explains few key reasons like SAS Encoding, 32bit vs 64 bit SAS, BUFSIZE, Database Encoding etc. which affect the data set size during executing SAS job to SAS Grid Platform. It explains how these factors impact the data size and how we can address these issues to match to original size of SAS data sets. This will help users and SAS administrators to adopt new SAS® Grid platform easily and in efficient manner.

### INTRODUCTION

If users are moving from non-grid to SAS Grid platform, then size difference is one of the common issues. In fact, if users are migrating from one environment to other where both are differing by operating system, SAS options etc. then there is more chance to have the size difference issue. Sometime database configuration also becomes important factor which affect the size of data set. In other words, there can be many reasons for data size issue which may be because of input data source, destination location and execution platform.

Apart from above given factors there may be many other factors. But, we are going to discuss the following key factors responsible for data set size issue, in this paper:

- BUFSIZE Option Impact
- SAS Encoding Impact
- 32bit vs 64bit SAS Software
- Database Encoding Impact

## 1. SAS BUFSIZE OPTION IMPACT ON DATASET SIZE

This section explains how BUFSIZE impact the data set size. Before discussing the examples to demonstrate the impact, let's see what different ways to use BUFSIZE option in SAS are. This can be used in two ways, as system and data set options in SAS session.

**1.1. SYSTEM OPTION** – If we will use BUFSIZE as system option, then new value for BUFSIZE will be effective for entire SAS session. Data set created in this SAS session will use the new value from system option. As system option, it can be defined in <SASHome>/SASFoundation/9.4/sasv9.cfg file. Below code can be run to verify the SAS session encoding:

```
proc options option=encoding;  
run;
```

In SAS Grid environment BUFSIZE can be defined in multiple ways. If there are multiple SAS Application servers, then this can be defined in <SASHome>/sasconfig/Levl/<SASApp>/

sasv9\_usermods.cfg file. This provides the flexibility to have multiple BUFSIZE is a SAS Grid platform.

**1.2. DATA SET OPTION** - If BUFSIZE will be used as data step, it will be effective only with specified data set. Proc contents can be used with data set to check the encoding:

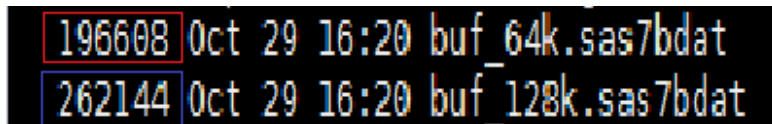
```
data <DataSet>(bufsize=<>);

proc contents data=test;
run;
```

Below SAS code is creating two data sets, buf\_64k with BUFSIZE 64K and buf\_128k with BUFSIZE 128K. Both data steps are using BUFSIZE and writing the data sets with same input source.

```
libname home "$HOME";
data home.buf_64k(bufsize=64k);
    set sashelp.cars;
run;
data home.buf_128k(bufsize=128k);
    set sashelp.cars;
run;
```

As shown below, there are two data sets created from above SAS code. Data set buf\_64k was created with disc size 196608. Data set buf\_128k was created with size 262144. Below output shows that if we increased the BUFSIZE for data step then it increases the data set size created from that data step.

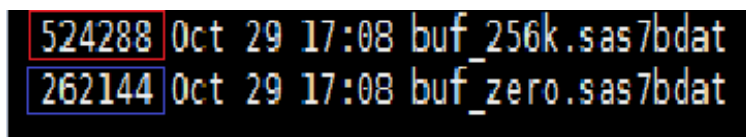


A Linux terminal screenshot showing the output of the SAS code. The first line shows '196608 Oct 29 16:20 buf\_64k.sas7bdat' and the second line shows '262144 Oct 29 16:20 buf\_128k.sas7bdat'. The numbers 196608 and 262144 are highlighted with red and blue boxes respectively.

Display 1. Data Set from Linux Screen

In below code, there are two data steps. First data step is creating the data set buf\_256k with BUFSIZE=256K and next one buf\_zero with BUFSIZE=0. Display2 shows that as we increased the BUFSIZE to 256K then it increased the data set size by two times of buf\_128k. When we give BUFSIZE with ZERO in data step then it takes the default value of SAS system. As shown in Display1 buf\_128k size is 262144 and buf\_zero in below screen print is with same size, which shows the default BUFSIZE value set for SAS system is 128K.

```
data home.buf_256k(bufsize=256k);
    set sashelp.cars;
run;
data home.buf_zero(bufsize=0);
    set sashelp.cars;
run;
```



A Linux terminal screenshot showing the output of the SAS code. The first line shows '524288 Oct 29 17:08 buf\_256k.sas7bdat' and the second line shows '262144 Oct 29 17:08 buf\_zero.sas7bdat'. The numbers 524288 and 262144 are highlighted with red and blue boxes respectively.

Display 2. SAS Data Sets Size from Linux Screen

**Note** – In above examples, BUFSIZE option is used as data set option and that's why each data set is being created with different disc size. If we don't use BUFSIZE with data steps, then default BUFSIZE (in this case it's 128K) will be used for each data steps execution.

## 2. SAS ENCODING IMPACT ON DATA SET SIZE

Whenever SAS read or write SAS data sets, SAS insures that Data Set has been converted to current session encoding before further processing. But user can give any representation to SAS to create the target data set. For example, if user is running window SAS, the new data sets can be created for Linux or another platform format. If user will not provide any format, then new data set will be created in current session format/ encoding. If source data set encoding doesn't match to session encoding, then it brings additional task for SAS engine to perform. SAS use CEDA (cross environmental data access) to convert the foreign data set to native format (where SAS is running) and then use for further processing.

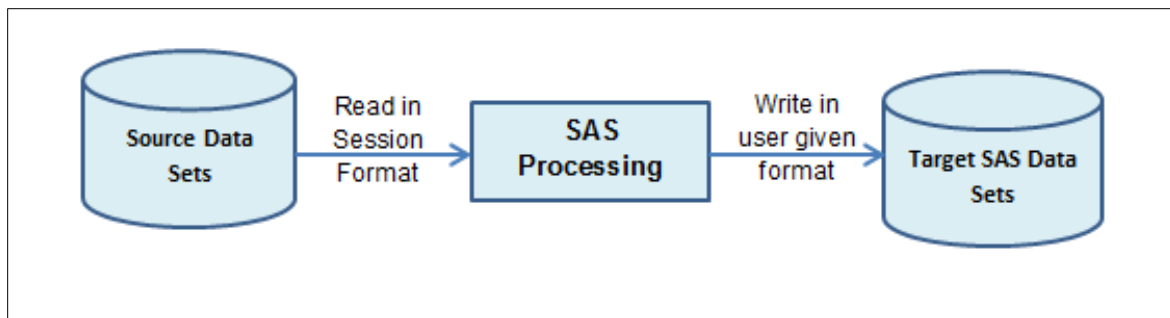


Figure 1. Read Data to SAS Session

When user move to SAS Grid from departmental SAS servers, it become difficult for SAS administrator to configure the SAS grid to fulfill all different encoding. Encoding difference become more critical if organization is running SAS on multiple operating system. For example, if few users are using PC SAS then most of the time they use "WLATIN1" encoding. If these users are moving to SAS Grid installed in Linux environment, then they will face encoding issue in previously written code.

### UTF-8 vs Latin1

Utf-8 has been one of the commonly used SAS encoding options, which take care of multiple types of data sources. It takes four bytes to store the data, which help to print the special character properly (generally special character are properly stored in single byte) without any data loss. So, if you are reading a data source which contains special character, in single byte SAS session (Latin1) then you may face data truncation. Reason is simple; SAS could not assign sufficient space for character to store in SAS session (This will be further explained in 4<sup>th</sup> topic).

Latin1 stores the data in single byte character. Since UTF-8 takes the four bytes to store the character whereas Latin1 takes only single byte so if we create a data set in UTF-8 SAS it will take more disc space compare to if the same data set is being created in Latin1 SAS environment.

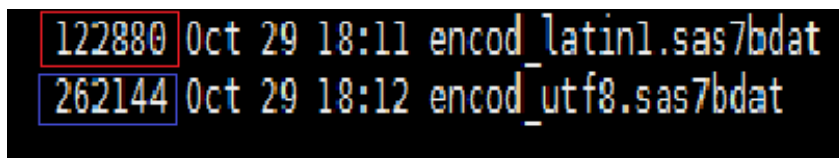
As shown in below SAS code `encode_latin1` data set is created with encoding Latin1. Then in next data step `encod_utf8` is created with UTF-8 encoding by reading `encod_latin1`. So, content wise both data sets are exactly same but as per display 2, the data size for these two data sets are completely different. The example clearly shows that, UTF-8 encoded data set takes more space to save compare to what Latin1 encoded data set. Many times, when user read Latin1 encoded data set in UTF-8 SAS then

data set size increased by almost four times. The fundamental reason behind this is, when data set moved to UTF-8 SAS session, then SAS started assigning 4 bytes to store a single character, where as it was taking single byte in Latin1 SAS.

```
/* Below code is executed in SAS System with Latin1 encoding. */
data home.encod_latin1;
    set sashelp.zhc;
run;

/* Below code is executed in SAS System with UTF-8 encoding. */
data home.encod_utf8;
    set home.encod_latin1;
run;
```

As shown in below screen print, two data sets were created with different disc size. In above example encod\_latin1 data set is created on SAS system with encoding Latin1. In Latin1 environment data set size is 122880. Then we created a new data set from encod\_latin1 to encod\_utf8 but this time data set size changed to 262144. So, as we changed the encoding, data set size on disc increased.



Display 3. Data Set from Linux Screen

**Note** – This impact needs to be taken care before migration to new SAS environment if there encoding is not same. If there is any downstream application reading the SAS data then we need to make sure that if there is increase in data size then next application should be able to handle increased SAS Data Sets.

### 3. 32BIT VS 64BIT SAS SOFTWARE/ OPERATING SYSTEM

#### SAS Bit for Linux SAS Install

We can use 'file' command at Linux command line against the SAS executable:

```
file <SASHOME>/SASFoundation/9.x/sasexe/sas
```

The 64-bit SAS install returns something similar to:

```
SASFoundation/9.4/sasexe/sas: ELF 64-bit LSB executable, x86-64, version 1
(SYSV), dynamically linked (uses shared libs), for GNU/Linux 2.6.18, not
stripped
```

For 32 bit SAS install, the above command returns similar to:

```
SASFoundation/9.4/sasexe/sas: ELF 32-bit LSB executable, version 1 (SYSV),
dynamically linked (uses shared libs), for GNU/Linux 2.6.4, not stripped
```

#### SAS Bit for Window SAS Install

As shown below, when we launch PC SAS then it is to find SAS bit from startup log message.

For 64 bit SAS log, will show like:

```
X64_7PRO WIN 6.1.7601 Service Pack 1 Workstation
```

For 32 bit SAS, it will show like:

```
W32_7PRO WIN 6.1.7601 Service Pack 1 Workstation
```

Many times, it has been observed that when we move from 32 bit SAS to 64 bit then data size get increased. This is because of the SAS engine architecture. There is no straight formula to represent the data increased but this happens 64 bit SAS take more space for descriptor portion. In most of the cases 32-bit created data is not supported in 64 bit SAS. So, that migration become necessary to use these data to 64 bit SAS.

## 4. DATABASE ENCODING IMPACT

Here we discuss how database encoding impacts the data set size. If database encoding doesn't match to SAS session, it may cause some data size differences. If there is mismatch in encoding, then it depends how SAS is reading the data in current session. The easiest way to help in this problem is to match the SAS encoding to database value. If changing the SAS system encoding is not possible then we can use `dbclient_max_bytes` and `bserver_max_bytes` to help in this situation.

Before checking the effect of these options on variable length in SAS, we need to check the database encoding which help to decide how to use database options in SAS code.

### 4.1. CHECK DATABASE ENCODING

#### Using Base SAS Code

Below SAS code can be run into SAS session and it will return the database encoding to output window screen. This helps SAS user to check the database without connecting to database from command line. If we can fetch the encoding value in SAS session itself, then it can be further used in SAS code if required. Database name can be given for "path" SQL statement.

```
proc sql;
    connect to oracle(user="<admapp>" password="<Password>"
path="<STP322>");
    select * from connection to oracle(select * from
nls_database_parameters where parameter = 'NLS_CHARACTERSET');
    disconnect from oracle;
quit;
```

PARAMETER	VALUE
NLS_CHARACTERSET	<Database_Encoding>

**Output 1. SAS Output for Database Encoding**

#### Using Database Command Prompt

Below code can be run from database command line to fetch the database encoding.

```
STP322> select * from nls_database_parameters where parameter =
'NLS_CHARACTERSET';
```

PARAMETER	VALUE
NLS_CHARACTERSET	<Database_Encoding>

**Output 2. Database Encoding from Command Line**

## 4.2. DBSERVER\_MAX\_BYTES AND DBCLIENT\_MAX\_BYTES

As discussed before, if there is mismatch in database and SAS encoding then there may be data size difference and we can use `dbclient_max_bytes` and `bserver_max_bytes` options with SAS Libname statement to fetch the record with appropriate length.

*DBCLIENT\_MAX\_BYTES* gives the maximum numbers of bytes to store a character to database client side. This is same to the SAS encoding if it's not changed explicitly. Section 4.2.1 SAS log shows the default value of `dbclient_max_bytes` which is FOUR but it can be changed in SAS Libname statement.

*DBSERVER\_MAX\_BYTES* gives the maximum numbers of bytes to store a character in database server side. Section 4.2.1 SAS log clearly shows the default value of `dbserver_max_bytes` to 4.

Below examples demonstrate the effect different values of `dbclient_max_bytes` and `bserver_max_bytes` on data set size from SAS session. If new SAS environment data set does not match to previous then we can decide what should be the value for `dbclient_max_bytes` to be used in SAS.

### 4.2.1. Libname without DBSERVER\_MAX\_BYTES and DBCLIENT\_MAX\_BYTES

Below example shows the Libname reference without using `dbclient_max_bytes` or `bserver_max_bytes` in SAS code. As shown in the partial log message, the default values of these two options are 4, which created the STPNAME and STPOPTNAME with the length 80 and 252. This is the default Libname behaviour in SAS (in this SAS session the default value for `dbclient_max_bytes` or `bserver_max_bytes` is 4). In subsequent examples, we will see the effect of explicit use of these options.

```
options sastrace='d,,, ' sastraceloc=saslog;
LIBNAME testdb ORACLE PATH=STP322 SCHEMA=TEST_T5 USER=admapp
PASSWORD="Password";

proc contents data=testdb.db_cdnm;
run;
```

**Partial SAS Log:**

. . .

```
ORACLE: DBSERVER_MAX_BYTES=4 44 1418950073 no_name 0 Program
ORACLE: DBCLIENT_MAX_BYTES=4 45 1418950073 no_name 0 Program
```

. . .

Alphabetic List of Variables and Attributes						
#	Variable	Type	Len	Format	Informat	Label
1	STPNAME	Char	80	\$80.	\$80.	STPNAME
2	STPOPTNAME	Char	252	\$252.	\$252.	STPOPTNAME

**Output 3. Output from SAS Enterprise Guide Proc Contents Results****4.2.2. LIBNAME WITH DBCLIENT\_MAX\_BYTES=1**

In below code `dbclient_max_bytes` option is used with the value 1 in Libname statement. We can see the partial log message to check the value of these options. In output window, we can see that changing the value of `dbclient_max_bytes` to ONE changed the variable length of STPNAME and STPOPTNAME to 20 and 63. If value of `dbclient_max_bytes` is changed from 4 (default) to one (explicit) then variable length changed to one fourth.

We need to give appropriate value for "Path" (Database Name) and "Schema" (Database Schema name) in below Libaname statement.

```
options sastrace='d,,,' sastraceloc=saslog;
LIBNAME testdb ORACLE PATH=STP322 SCHEMA=TEST_T5 USER=admapp
PASSWORD="Password"
    ADJUST_BYTE_SEMANTIC_COLUMN_LENGTHS=NO
    ADJUST_NCHAR_COLUMN_LENGTHS=YES
    DBCLIENT_MAX_BYTES=1 DBMAX_TEXT=32767;

proc contents data=testdb.db_cdm;
run;
```

**Partial SAS Log:**

. . .

```
ORACLE: DBSERVER_MAX_BYTES=4 112 1418955281 no_name 0 Program
ORACLE: DBCLIENT_MAX_BYTES=1 113 1418955281 no_name 0 Program
```

. . .

Alphabetic List of Variables and Attributes						
#	Variable	Type	Len	Format	Informat	Label
1	STPNAME	Char	20	\$20.	\$20.	STPNAME
2	STPOPTNAME	Char	63	\$63.	\$63.	STPOPTNAME

**Output 4. Output from SAS Enterprise Guide Proc Contents Results**

#### 4.2.3. LIBNAME WITH DBCLIENT\_MAX\_BYTES=2

In below code `dbclient_max_bytes` option is used with value 2. Output window shows that changing the value of `dbclient_max_bytes` to TWO changed the variable length of STPNAME and STPOPTNAME to 40 and 126.

```
options sastrace='d,,,' sastraceloc=saslog;
LIBNAME testdb ORACLE PATH=STP322 SCHEMA=TEST_T5 USER=admapp
PASSWORD="Password"
    ADJUST_BYTE_SEMANTIC_COLUMN_LENGTHS=NO
    ADJUST_NCHAR_COLUMN_LENGTHS=YES
    DBCLIENT_MAX_BYTES=2 DBMAX_TEXT=32767;

proc contents data=testdb.db_cdm;
run;
```

##### Partial SAS Log:

```
. . .
ORACLE: DBSERVER_MAX_BYTES=4 129 1418955499 no_name 0 Program
ORACLE: DBCLIENT_MAX_BYTES=2 130 1418955499 no_name 0 Program
. . .
```

Alphabetic List of Variables and Attributes						
#	Variable	Type	Len	Format	Informat	Label
1	STPNAME	Char	40	\$40.	\$40.	STPNAME
2	STPOPTNAME	Char	126	126.	\$126.	STPOPTNAME

Output 5. Output from SAS Enterprise Guide Proc Contents Results

#### 4.2.4. LIBNAME WITH DBCLIENT\_MAX\_BYTES=3

In below code `dbclient_max_bytes` option is used with value 3. Output window shows that changing the value of `dbclient_max_bytes` to two changed the variable length of STPNAME and STPOPTNAME to 60 and 189. Variables created by this code were length of three time compare to the data set created with `DBCLIENT_MAX_BYTES=1`.

```
options sastrace='d,,,' sastraceloc=saslog;
LIBNAME testdb ORACLE PATH=STP322 SCHEMA=TEST_T5 USER=admapp
PASSWORD="Password"
    ADJUST_BYTE_SEMANTIC_COLUMN_LENGTHS=NO
    ADJUST_NCHAR_COLUMN_LENGTHS=YES
    DBCLIENT_MAX_BYTES=3 DBMAX_TEXT=32767;

proc contents data=testdb.db_cdm;
run;
```

##### Partial SAS Log:

```
. . .
ORACLE: DBSERVER_MAX_BYTES=4 146 1418955654 no_name 0 Program
ORACLE: DBCLIENT_MAX_BYTES=3 147 1418955654 no_name 0 Program
. . .
```



Alphabetic List of Variables and Attributes						
#	Variable	Type	Len	Format	Informat	Label
1	STPNAME	Char	60	\$60.	\$60.	STPNAME
2	STPOPTNAME	Char	189	189.	\$189.	STPOPTNAME

**Output 6. Output from SAS Enterprise Guide Proc Contents Results**

**Note** – From 4.2.2 to 4.2.4 the different value of `dbclient_max_bytes` show the change in variable length accordingly. Based on previous SAS environment, user can select the appropriate value for `dbclient_max_bytes` to match the data set size and variable length.

#### 4.2.5. LIBNAME WITH DBSERVER\_MAX\_BYTES=1/2/3

SAS code in this section was executed multiple times with different values of `dbserver_max_bytes=1/2/3`. We observed that there was no impact of using `dbserver_max_bytes` on variable length in SAS data set. As per "Output 7", for each value of `dbserver_max_bytes`, the STPNAME and STPOPTNAME were created with length 80 and 252. There is no impact of using `dbserver_max_bytes` on variable length.

```
options sastrace='d,,, ' sastraceloc=saslog;
LIBNAME testdb ORACLE PATH=STP322 SCHEMA=TEST_T5 USER=admapp
PASSWORD="Password"
        ADJUST_BYTE_SEMANTIC_COLUMN_LENGTHS=NO
ADJUST_NCHAR_COLUMN_LENGTHS=YES
        DBSERVER_MAX_BYTES=3 DBMAX_TEXT=32767;

proc contents data=testdb.db_cdm;
run;
```

Alphabetic List of Variables and Attributes						
#	Variable	Type	Len	Format	Informat	Label
1	STPNAME	Char	80	\$80.	\$80.	STPNAME
2	STPOPTNAME	Char	252	\$252.	\$252.	STPOPTNAME

**Output 7. Output from SAS Enterprise Guide Proc Contents Results**

## CONCLUSION

When we move/ migrate from one SAS platform to another or SAS Grid then data size issue is one of the common issues what application teams face. Though this paper explains many key reasons for data size but there may be another reason as well. SAS Administrator should analyze these factors before installing or migrating new servers, to avoid this issue.

## ACKNOWLEDGEMENT

Authors would like to thank Lisa Mendez, and Lizette Alonzo, SCSUG 2016 Educational Forum Co-Chair, for accepting our abstracts, paper and for organizing a great conference.

## REFERENCE

Prasoon Sangwan, Tanuj Gupta, Piyush Singh, “Key Requirements for SAS® Grid Users” Proceedings SAS Global Forum 2016, Las Vegas, NV, USA.

<http://support.sas.com/resources/papers/proceedings16/7140-2016.pdf>

Prasoon Sangwan, Piyush Singh, Shiv Govind, “Integrating SAS® and the R Language with Microsoft SharePoint” Proceedings SAS Global Forum 2015, Dallas, TX, USA.

<https://support.sas.com/resources/papers/proceedings15/2500-2015.pdf>

Piyush Singh, Gerhardt M Pohl, “Enhancing SAS® Piping Through Dynamic Port Allocation” Proceedings of SAS Global Forum 2014, Washington, DC, USA.

<http://support.sas.com/resources/papers/proceedings14/1826-2014.pdf>

[https://en.wikipedia.org/wiki/SAS\\_\(software\)](https://en.wikipedia.org/wiki/SAS_(software))

## CONTACT INFORMATION

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

Prasoon Sangwan

[prasoon.sangwan@tcs.com](mailto:prasoon.sangwan@tcs.com)

Piyush Singh

[piyushkumar.singh@tcs.com](mailto:piyushkumar.singh@tcs.com)

Tanuj Gupta

[tanuj.gupta@tcs.com](mailto:tanuj.gupta@tcs.com)

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 trademarks of their respective companies.