Class : 3

SAS Tutorial

PRESENTED BY : SHASHI KUMAR

PDV : Program Data Vector

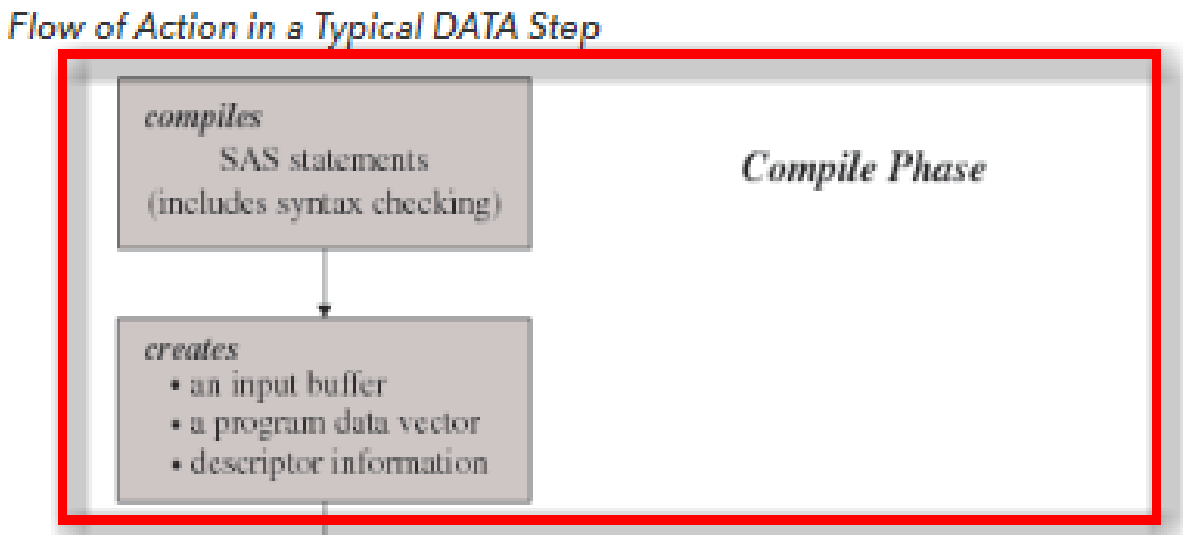
Before Actually moving to the PDV section let us try to explorer how SAS actually its data/steps in a sequence.

SAS initiate its code into two parts :

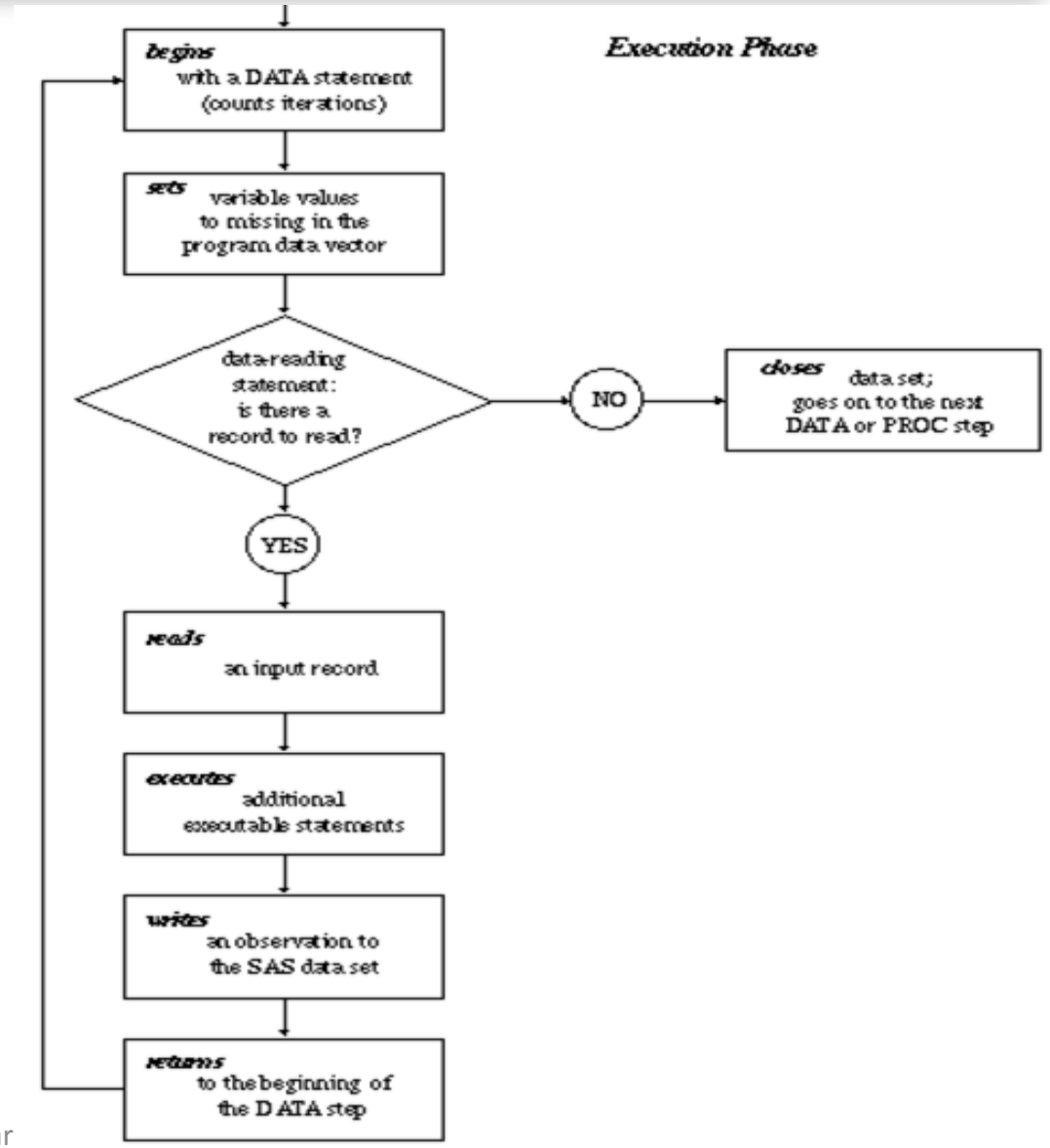
1. Compilation Phase
2. Execution Phase

- In the compilation phase SAS checks the syntax of the submitted code and if there is a syntax error then SAS stops the further process and at the end of the compilation phase (i.e. after checking syntax and good to go) SAS creates the descriptor portion of the dataset.

Compilation Phase :



1. Syntax Error
2. Create PDV
3. Create descriptor portion



Compilation Phase:

When you submit a DATA step for execution, SAS checks the syntax of the SAS statements and compiles them, that is, automatically translates the statements into machine code. In this phase, SAS identifies the type and length of each new variable, and determines whether a variable type conversion is necessary for each subsequent reference to a variable. During the compile phase, SAS creates the following three items:

1. Input buffer :

Is a logical area in memory into which SAS reads each record of raw data when SAS executes an INPUT statement. Note that this buffer is created only when the DATA step reads raw data. (When the DATA step reads a SAS data set, SAS reads the data directly into the program data vector.)

2. Program Data Vector (PDV) :

It is a logical area and virtual memory, which is used for manipulation purpose. It creates two automatically temporary variables.

1. **_N_** : It represent number of iteration done by data steps. Where one iteration equal to one observation.

2. **_ERROR_** : It represent data error in the record has two values.

a) **0** : It represent no error in particular record.

b) **1** : It represent error in the specific record. It does not represent that how many error are there in the record.

3. Descriptor Information :

Is information that SAS creates and maintains about each SAS data set, including data set attributes and variable attributes. It contains, for example, the name of the data set and its member type, the date and time that the data set was created, and the number, names and data types (character or numeric) of the variables.

Descriptor Portion :

- How to see descriptor portion of the dataset in sas ??

```
proc contents data=sashelp.class;
run;
```

The CONTENTS Procedure

Data Set Name	SASHELP.CLASS	Observations	19
Member Type	DATA	Variables	6
Engine	V9	Indexes	0
Created	02/12/2014 00:07:34	Observation Length	48
Last Modified	02/12/2014 00:07:34	Deleted Observations	0
Protection		Compressed	NO
Data Set Type		Sorted	NO
Label			
Data Representation	WINDOWS_64		
Encoding	wlatin1 Western (Windows)		

Engine/Host Dependent Information	
Data Set Page Size	65536
Number of Data Set Pages	1
First Data Page	1
Max Obs per Page	1361
Obs in First Data Page	19
Number of Data Set Repairs	0
ExtendObsCounter	YES
Filename	C:\Program Files\SASHome\SASFoundation\9.4\nls\en\SASCFG\class.sas7bdat
Release Created	9.0401M1
Host Created	X64_8PRO

Alphabetic List of Variables and Attributes

#	Variable	Type	Len
4	Age	Num	8
5	Height	Num	8
2	Name	Char	8
1	Obs	Num	8
3	Sex	Char	8
6	Weight	Num	8

Execution Phase:

- The DATA step begins with a DATA statement. Each time the DATA statement executes, a new iteration of the DATA step begins, and the _N_ automatic variable is incremented by 1.
- SAS sets the newly created program variables to missing in the program data vector (PDV).
- SAS reads a data record from a raw data file into the input buffer, or it reads an observation from a SAS data set directly into the program data vector. You can use an INPUT, MERGE, SET, MODIFY, or UPDATE statement to read a record.
- SAS executes any subsequent programming statements for the current record.
- At the end of the statements, an output, return, and reset occur automatically. SAS writes an observation to the SAS data set, the system automatically returns to the top of the DATA step, and the values of variables created by INPUT and assignment statements are reset to missing in the program data vector. Note that variables that you read with a SET, MERGE, MODIFY, or UPDATE statement are not reset to missing here.
- SAS counts another iteration, reads the next record or observation, and executes the subsequent programming statements for the current observation.
- The DATA step terminates when SAS encounters the end-of-file in a SAS data set or a raw data file.

1. Read data step and initializes with missing values.
2. Set statement read one observation
3. Implicit output/Return

PDV (Program Data Vector) :

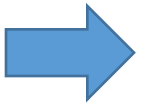
It is a logical area and virtual memory, which is used for manipulation purpose. It creates two automatically temporary variables.

1. **_N_** : It represent number of iteration done by data steps. Where one iteration equal to one observation.
2. **_ERROR_** : It represent data error in the record has two values.

a) **0** : It represent no error in particular record.

b) **1** : It represent error in the specific record. It does not represent that how many error are there in the record.

```
data a;
Put _all_;
set sashelp.class;
put _all_;
run;
```



```
Obs=. Name=  Sex=  Age=. Height=. Weight=. _ERROR_=0 _N_=1
Obs=1 Name=Alfred Sex=M Age=14 Height=69 Weight=112.5 _ERROR_=0 _N_=1
Obs=1 Name=Alfred Sex=M Age=14 Height=69 Weight=112.5 _ERROR_=0 _N_=2
Obs=2 Name=Alice Sex=F Age=13 Height=56.5 Weight=84 _ERROR_=0 _N_=2
Obs=2 Name=Alice Sex=F Age=13 Height=56.5 Weight=84 _ERROR_=0 _N_=3
```

PDV (Program Data Vector) :

Name	Age	Gender
Ram	21	M
Sita	20	F
Radha	19	F

Program:-

A=Data one**;**

B=Set test**;**

C=run;

➢1. Read data step and initialize with missing value.

	Name	Age	Gender	_Error_	_N_
A		.		0	1

➢2.Set statement read one observation

	Name	**Age**	**Gender**	**Error**	**_N_**
A		.		0	1
B	**Ram**	**21**	**M**	**0**	**1**

➢3. Implicit output

	Name	**Age**	**Gender**
C	Ram	21	M

➢4. Implicit Return

	Name	**Age**	**Gender**	**Error**	**_N_**
A		.		0	1
B	Ram	21	M	0	1
A	Ram	21	M	0	2

IF/Where/Keep/Drop Statement:

WHERE/(Keep/Drop in set statement)

Keep/Drop in dataset Statement

IF/Keep/Drop Statements

IF/Where/Keep/Drop Statement:

```
Data one;
set sashelp.class;
run;

Data two;
set sashelp.class;
Keep Name age ;
run;

Data three;
set sashelp.class(keep=Name age);
run;

Data four(keep=Name age);
set sashelp.class;
run;
```

```
Data four1(drop=sex);
set sashelp.class(Drop=Height Weight);
Salary= age*1000;
drop age;
run;

Data five;
set sashelp.class;
if age >14;
run;

Data six;
Set sashelp.class;
where age > 14;
run;
```

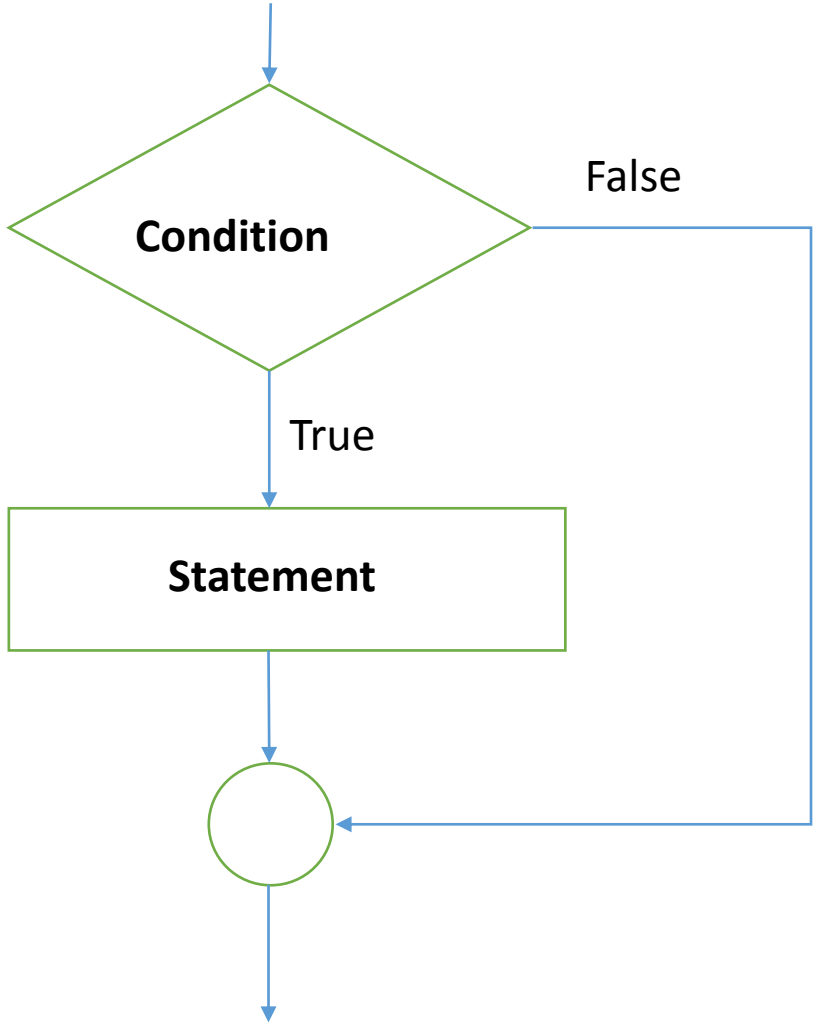
IF -THEN / IF -THEN –Else Statement :

IF Condition/expression THEN statement;

IF Condition/expression THEN statement1;
Else IF Condition/expression THEN statement2;
Else Statement3;

```
Data six1;
Set sashelp.class;
if age=15 then salary=age*1000;
run;
```

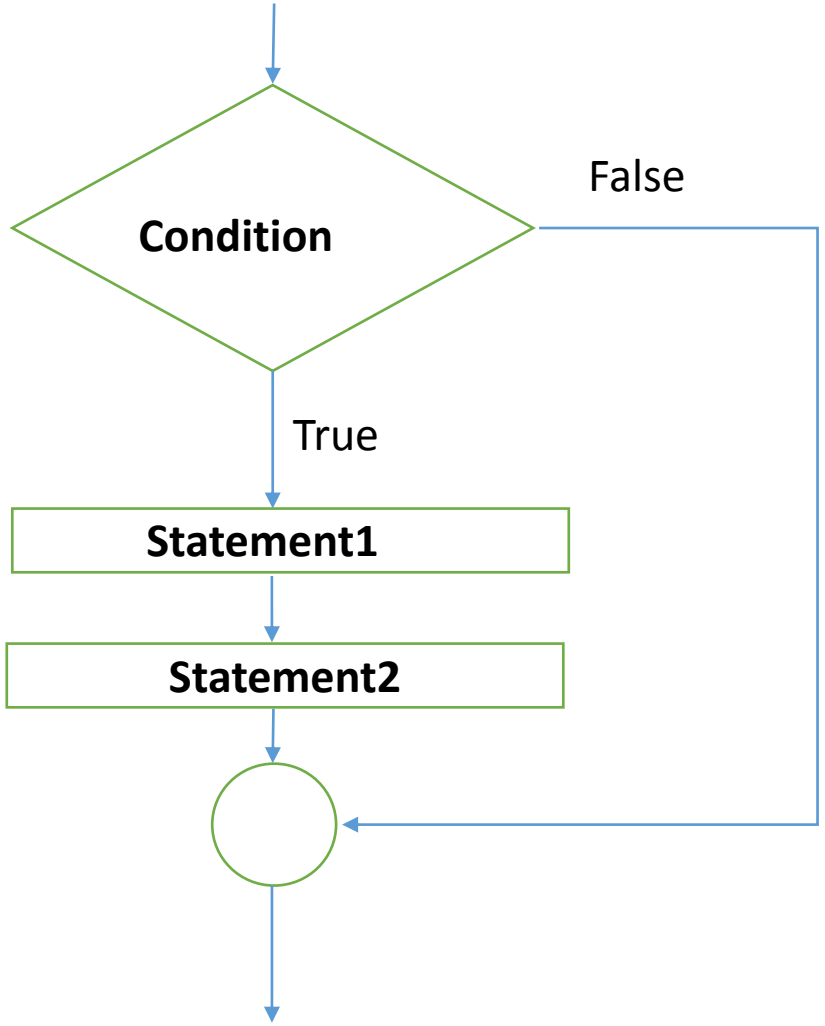
```
Data six1;
Set sashelp.class;
if age < 15 then salary=age*1000;
else if age=15 then salary=age*2000;
else salary=age*3000;
run;
```



IF -THEN-DO Statement :

```
IF Condition/expression THEN do;
        statement1;
        statement2;
End;
-----------------------------------------
IF Condition/expression THEN do;
        statement1;
        statement2;
End;
Else IF Condition/expression THEN do;
        statement1;
        statement2;
End;
```

```
Data six11;
Set sashelp.class;
if age=15 then do;
 Salary=age*1000;
Bonus= age *10;
End;
run;
```



Thank You ...

Presented By : Shashi Kumar
YouTube Channel : **https://lnkd.in/fNSUTDE**