INFILE STATEMENT

Infile statement identifies an external file or raw data
Identifies an external file to read data with an INPUT statement
Infile statement can also use a file reference using the Filename statement.

Syntax:-

INFILE file-specification <options > <operating-environment-options>;

Infile Statement Options: -

DSD (delimiter – sensitive – data):-

->It reads delimiter is a comma

->It reads missing values with delimiter is a comma

-> SAS treats two consecutive delimiters as a missing value

->It removes double quotation marks from character values.

Examples:-

```
Data ds1;
Infile datalines;
Input id name$ sex$ age sal;
Datalines;
001 abc m 23 45000
002 def f 34 67000
003 mno m 21 36000
004 xyz f 27 45000
;
Run;
```

```
Data ds1a;
Infile "C:\Documents and Settings\Administrator\Desktop\sample.txt";
Input id name$ sex$ age sal;
Run;
```

In above case default delimiter is space so you no need to use any Infile options
But see below example there is a comma delimiter so you should use Infile options to read data properly

```
Data ds2;
Infile datalines dsd;
Input id name$ sex$ age sal;
Datalines;
001,abc,m,23,45000
002,def,f,34,67000
003,mno,m,21,36000
004,xyz,f,27,45000
;
Run;
```

```
Data ds2b;
Infile "C:\Documents and Settings\Administrator\Desktop\sample.txt" dsd;
Input id name$ sex$ age sal;
Run;
```

```
Data ds2c;
Infile datalines dsd;
Input id name$ sex$ age sal;
Datalines;
001,abc,m,23,
002,def,f,34,67000
003,mno,m,21,36000
004,xyz,f,27,45000
;
Run;
```

```
Data ds2d;
Infile datalines dsd;
Input id name$ sex$ age sal;
Datalines;
001,"abc",m,23,
002,"def",f,34,67000
003,"mno",m,21,36000
004,"xyz",f,27,45000
;
Run;
```

```
Data ds2e;
Infile datalines dsd;
Input Name: $9. Score Team: $25. Div $;
Datalines;
Joseph,76,"Red Racers, Washington",AAA
Mitchel,82,"Blue Bunnies, Richmond",AAA
Sue Ellen,74,"Green Gazelles, Atlanta",AA
;
Run;
```

DLM (or) DELIMITER (Delimiter):-

When data values having special characters in raw data/an external file. Than we use DLM option to read data. This special character must be enclosed with quotes.

Examples:-

```
Data ds3;
Infile datalines dlm= ',';
Input id name$ sex$ age sal;
Datalines;
001,abc,m,23,45000
002,def,f,34,67000
003,mno,m,21,36000
004,xyz,f,27,45000
;
Run;
```

```
Data ds3a;
Infile datalines dlm='*';
Input id name$ sex$ age sal;
```

```
Datalines;
001*abc*m*23*45000
002*def*f*34*67000
003*mno*m*21*36000
004*xyz*f*27*45000
;
Run;
Data ds3b;
Infile datalines dlm='* ,' ;
Input id name$ sex$ age sal;
Datalines;
001*abc*m*23*45000
002*def,f*34*67000
003*mno*m*21,36000
004,xyz*f*27*45000
;
Run;
Data ds3c;
Infile datalines dlm='* , #'  ;
Input id name$ sex$ age sal;
Datalines;
001*abc,m*23,45000
002*def,f*34*67000
003#mno*m*21,36000
004*xyz#f*27,45000
;
Run;
Data ds3c1;
Infile datalines DELIMITER='* ,'  ;
Input id name$ sex$ age sal;
Datalines;
001*abc,m*23*45000
002*def,f*34*67000
003*mno*m*21,36000
004*xyz*f*27,45000
;
Run;
Data ds3d;
Infile "C:\Documents and Settings\Administrator\Desktop\sample.txt" dlm='* ,' ;
Input id name$ sex$ age sal;
Run;
```

Difference between DSD and DLM

DSD – It reads delimiter as a comma
DLM – It reads delimiter as any special character

DSD - It removes double quotes from character values
DLM - It won't removes double quotes from character values

DSD – It treats two consecutive delimiters as a missing value

DSD – It treats two consecutive delimiters as a missing value when we assign at least one blank space between consecutive delimiters

Main Difference between DSD and DLM

When you specify DSD, SAS treats two consecutive delimiters as a missing value and removes quotation marks from character values. But when you specify DLM=',' it won't treats two consecutive delimiters as a missing value but if we specify space between it works. And dlm won't remove quotation marks from character values.

Execute Below codes and observes difference between dsd and dlm.

```
Data ds1;
Infile datalines dsd;
Input id name$ sex$ age sal;
Datalines;
001,abc,m,23,45000
002,def,,34,67000
003,,,,36000
004,xyz,f,27,45000
;
Run;
```

```
Data ds2;
Infile datalines dlm=',';
Input id name$ sex$ age sal;
Datalines;
001,abc,m,23,45000
002,def,,34,67000
003,,,,36000
004,xyz,f,27,45000
;
Run;
```

```
Data ds3;
Infile datalines dlm=',';
Input id name$ sex$ age sal;
Datalines;
001,abc,m,23,45000 S
002,def, ,34,67000
003, , , ,36000
004,xyz,f,27,45000
;
Run;
```

DLMSTR:-

When data values having strings as a delimiter in raw data/an external file. Than we use DLMSTR option to read data

This string must be enclosed with quotes and case sensitive.

Example:-

```
Data ds3e;
Infile datalines dlm='a' ;
Input X Y Z;
Datalines;
1a2a3
4a5a6
7a8a9
;
Run;
Data ds3e1;
Infile datalines dlmstr='a' ;
Input X Y$ Z;
Datalines;
1ama3
4afa6
7ama9
;
Run;
Data ds3e2;
Infile datalines dlmstr='PRD' ;
Input X Y Z;
Datalines;
1PRD2PRD3
4PRD5PRD6
7PRD8PRD9
;
Run;
```

DLMSOPT=Options :-

When data values having strings as a delimiter that should be in one case in all places we can use dlm or dlmstr but if it is not in one case we should use dlmstr along with dlmsopt=’I’ to read data properly.

Options=i
specifies that case-insensitive comparisons will be done.

Options=t
specifies that trailing blanks of the string delimiter will be removed.

Example:-

```
Data ds3e4;
Infile datalines dlmstr='PRD' dlmsopt='i';
Input X Y Z;
Datalines;
1PRD2PRd3
4PrD5Prd6
7pRd8pRD9
;
Run;
```

Difference between DLM and DLMSTR

When our raw data has delimiter as a special character or string (same case) we should use DLM or DLMSTR to read the data properly

But when our raw data has delimiter as a string and that is in multiple cases we should use DLMSTR='String' and DLMSOPT='I' to read the data properly

We must use DLMSTR along with DLMSOPT.

FIRST OBS:-

Specify the first observation at which processing starts

Examples:-

```
Data ds4;
Infile datalines dlm='*' firstobs=2;
Input id name$ age sex$ sal;
Datalines;
001*Joseph*25*m*4500
002*Mitchel*24*m*3500
003*john*21*f*2500
004*miller*22*f*3000
005*brans*30*m*5000
;
Run;
```

```
Data ds4a;
Infile "C:\Documents and Settings\Administrator\Desktop\sample.txt" dlm='*'
firstobs=2;
Input id name$ sex$ age sal;
Run;
```

OBS:-

Specify the observation at which processing ends.

Examples:-

```
Data ds5;
Infile datalines dlm='*' obs=3;
Input id name$ age sex$ sal;
Datalines;
001*Joseph*25*m*4500
002*Mitchel*24*m*3500
003*john*21*f*2500
004*miller*22*f*3000
005*brans*30*m*5000
;
Run;
```

```
Data ds5a;
Infile "C:\Documents and Settings\Administrator\Desktop\sample.txt" dlm='*'
firstobs=4;
Input id name$ sex$ age sal;
Run;
```

```
Data ds5b;
Infile datalines dlm='*' firstobs=2 obs=4;
Input id name$ age sex$ sal;
Datalines;
001*Joseph*25*m*4500
002*Mitchel*24*m*3500
003*john*21*f*2500
004*miller*22*f*3000
005*brans*30*m*5000
;
Run;

Data ds5c;
Infile "E:\Stansys\sas\sample.txt" dlm='*' firstobs=2 obs=4;
Input id name$ sex$ age sal;
Run;

Data Health1;
Infile Datalines firstobs=3 obs=7;
Input idno name$ team $ strtwght endwght;
x=_n_;
Datalines;
1023 David  red    189 165
1049 Amelia yellow 145 124
1219 Alan   red    210 192
1246 Ravi   yellow 194 177
1078 Ashley red    127 118
1221 Jim    yellow 220   .
1095 Susan  blue   135 127
1157 Rose   green  155 141
1331 Jason  blue   187 172
;
Run;
Data Health2;
Infile Datalines ;
Input idno name$ team $ strtwght endwght;
x=_n_;
If _n_ in(3:7);
Datalines;
1023 David  red    189 165
1049 Amelia yellow 145 124
1219 Alan   red    210 192
1246 Ravi   yellow 194 177
1078 Ashley red    127 118
1221 Jim    yellow 220   .
1095 Susan  blue   135 127
1157 Rose   green  155 141
1331 Jason  blue   187 172
;
Run;
```

FLOWOVER

It is default. Causes the INPUT statement to jump to the next record if it doesn't find values for all variables.

If you specify or without specify It gives same result, it is default.

Examples:-

```
Data ds6a;
Infile datalines flowover;
Input  Id Type$ Amount;
Datalines;
101 x
102 x 2000
103 y 3400
104 y
105 x 3000
;
Run;
```

```
Data ds6;
Infile datalines flowover;
Input  Id Type$ Amount;
Datalines;
101 x 3400
102 x 2000
103 y 3400
104 y 2500
105 x 3000
;
Run;
```

MISSOVER:-

When we have missing values in raw data or external file at the end of a data record is encountered than we will use Missover in Infile statement.
Missing value are represented for

1. Numeric values denoted by period (.)
2. Character values denoted by blank

Examples:-

```
Data ds7;
Infile datalines Missover;
Input  Id Type$ Amount;
Datalines;
101 x
102 x 2000
103 y 3400
104 y
105 x 3000
;
Run;
```

```
Data ds7a;
Infile datalines Missover;
Input id name$ sex$ age sal;
Datalines;
001 ABC M 23 50000
002 DEF F 27
003 MNO F 21 70000
004 XYZ M 25 58000
;
Run;
```

```
Data ds7b;
Infile datalines Missover ;
Input Lname$ Fname$ Emp_id$ Job_code$;
Datalines;
LANGKAMM SARAH E0045 Mechanic
TORRES JAN E0029 Pilot
SMITH MICHAEL E0065
LEISTNER COLIN E0116 Mechanic
TOMAS HARALD
WAUGH TIM E0204 Pilot
;
Run;
```

In below example no need to specify MISSOVER because data is separating with special character so it will read missing values without missover.

```
Data ds7c;
Infile datalines dlm='*' ;
Input id name$ sex$ age sal;
Datalines;
001*abc*m*23*45000
002* * *34*67000
003*mno*m*21*
004*xyz* *27*45000
;
Run;
```

STOPOVER

Stops the DATA step when it reads a short line.

Causes the DATA step to stop execution immediately and write a note to the SAS log.

Example:-

```
Data ds8;
Infile datalines stopover;
Input  Id Type$ Amount;
Datalines;
101 x
102 x 2000
103 y 3400
104 y
```

```
105 x 3000
;
Run;
```

```
ERROR: INPUT statement exceeded record length.
       INFILE CARDS OPTION STOPOVER specified.
RULE:      ----+----1----+----2----+----3----+----4----+----5---
168        101 x
Id=101 Type=x Amount=. _ERROR_=1 _N_=1
NOTE: The SAS System stopped processing this step because of errors.
WARNING: The data set WORK.DS6 may be incomplete.  When this
         step was stopped there were 0 observations and 3variables.
WARNING: Data set WORK.DS6 was not replaced because this step was stopped.
NOTE: DATA statement used (Total process time):
      real time           0.01 seconds
      cpu time            0.00 seconds
```

But it won't give error if you take the data properly without missing values because there is no short line. See below example

```
Data ds8a;
Infile datalines stopover;
Input Id Type$ Amount;
Datalines;
101 x 3400
102 x 2000
103 y 3400
104 y 2500
105 x 3000
;
Run;
```

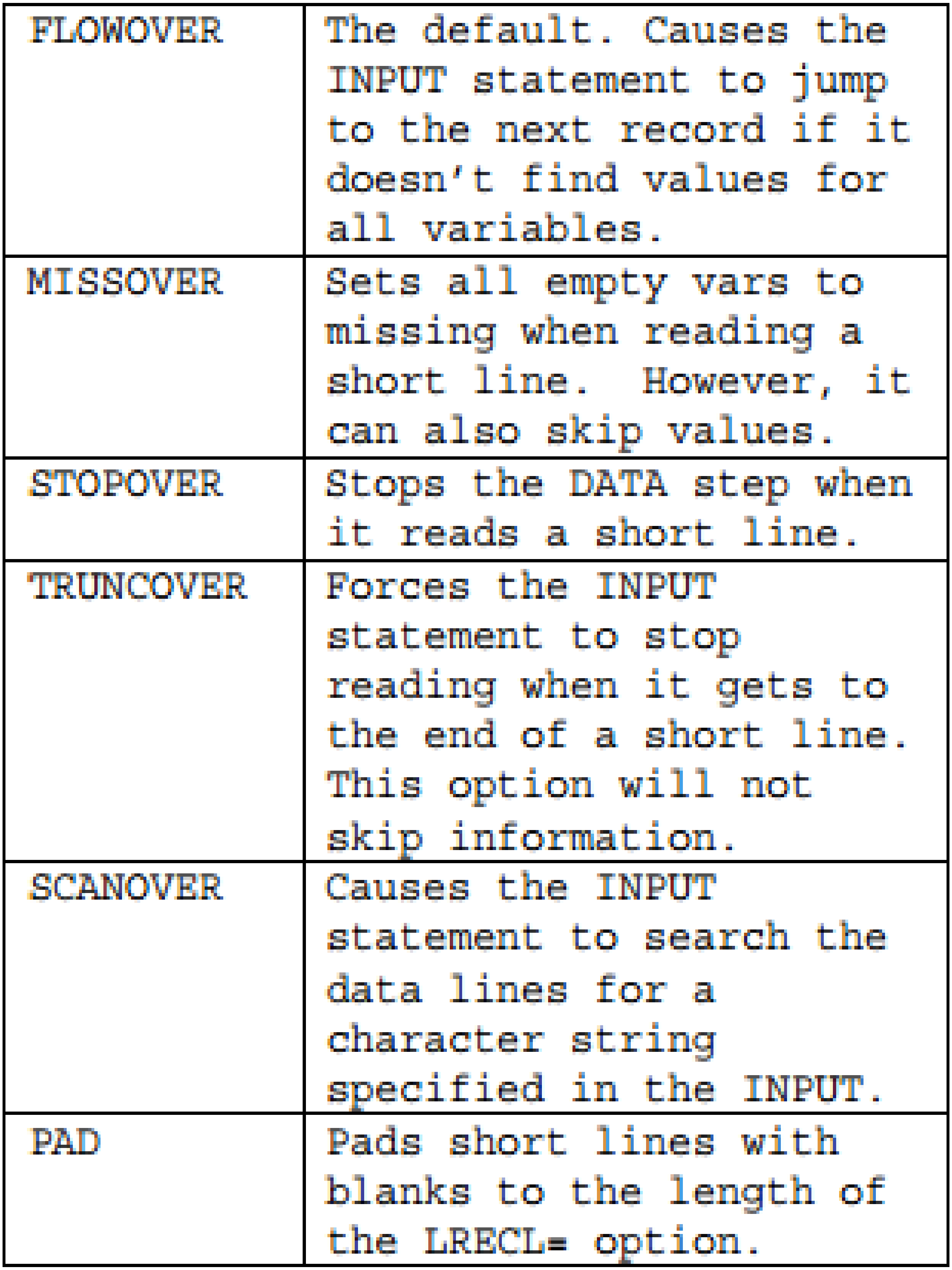
SCANOVER

causes the INPUT statement to scan the input data records until the character string that is specified in the @'character-string' expression is found.

SAS went to a new line when INPUT @'CHARACTER_STRING' scanned past the end of a line.

Example:-

```
Data ds;
Infile datalines scanover ;
Input @'Mobile' contact_details$;
Datalines;
Mobile 9542195422
Phone 040-42204449
Mobile 7671076710
Website stansys.co.in
Mail stansys.sas@gmail.com
;
Run;
```



FLOWOVER	The default. Causes the INPUT statement to jump to the next record if it doesn't find values for all variables.
MISSOVER	Sets all empty vars to missing when reading a short line. However, it can also skip values.
STOPOVER	Stops the DATA step when it reads a short line.
TRUNCOVER	Forces the INPUT statement to stop reading when it gets to the end of a short line. This option will not skip information.
SCANOVER	Causes the INPUT statement to search the data lines for a character string specified in the INPUT.
PAD	Pads short lines with blanks to the length of the LRECL= option.

END=Variable

Specifies a variable that SAS sets to 1 when the current input data record is the last in the input file. Until SAS processes the last data record, the END= variable is set to 0. Like automatic variables, this variable is not written to the data set.

Syntax:- End=Varuable_Name

```
Data ds1;
Infile "C:\Documents and Settings\Administrator\Desktop\sample.txt" end=x;
Input id name$ sex$ age sal add$;
Put x;
/*If x=1 then output;*/        * Select last record into dataset.
Run;
```

```
WARNING: The value of the INFILE END= option cannot be set for CARDS or DATALINES
```

FILENAME=Variable

Specifies a variable that SAS sets to the physical name of the currently opened input file. Like automatic variables, the FILENAME= variable is not written to the data set.

Syntax:- Filename=Varuable_Name

```
Data ds1;
Length x $100.;
Infile "C:\Documents and Settings\Administrator\Desktop\sample.txt" filename=x;
Input id name$ sex$ age sal add$;
Put x;
Run;
```

```
NOTE: The Infile "C:\Documents and Settings\Administrator\Desktop\sample.txt" is:
      Filename=C:\Documents and Settings\Administrator\Desktop\sample.txt,
      RECFM=V,LRECL=256,File Size (bytes)=96,Last Modified=15Jan2009:08:55:08,
      Create Time=15Jan2009:08:54:53

C:\Documents and Settings\Administrator\Desktop\sample.txt
```

COLUMN=Variable

Names a variable that SAS uses to assign the current column location of the input pointer. The COLUMN= variable is not written to the data set. Like automatic variable.

Syntax:- Column=Varable_Name

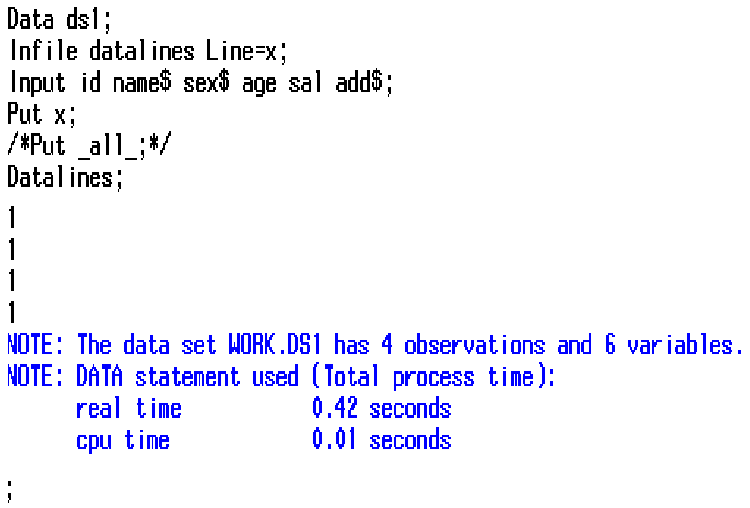
```
Data ds1;
Infile datalines column=x;
Input id name$ sex$ age sal add$;
Put x;
/*Put _all_;*/
Datalines;
001 abc m 23 45000 hyd
002 def f 34 67000 bang
003 mno m 21 36000 hyd
004 xyz f 27 45000 bang
;
Run;
```

LINE=*variable*

Specifies a variable that SAS sets to the line location of the input pointer in the input buffer. The LINE= variable is not written to the data set. Like automatic variable.

Syntax:- **Column=Varable_Name**

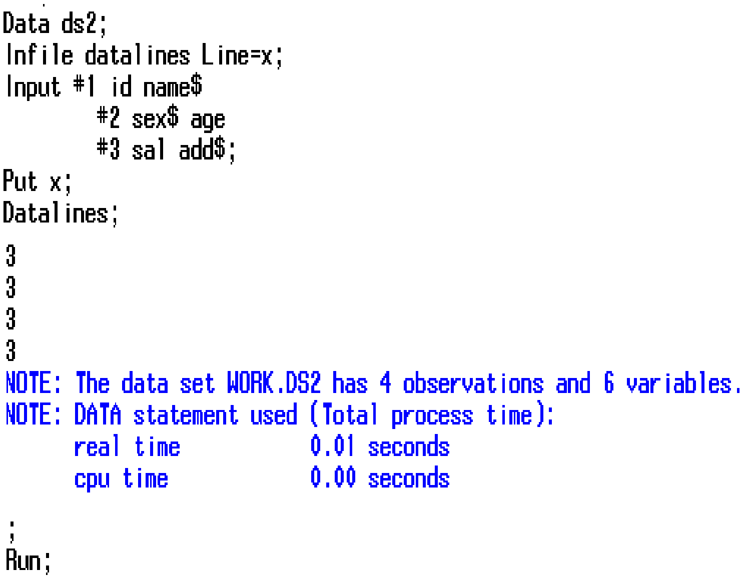
```
Data ds1;
Infile datalines Line=x;
Input id name$ sex$ age sal add$;
Put x;
/*Put _all_;*/
Datalines;
001 abc m 23 45000 hyd
002 def f 34 67000 bang
003 mno m 21 36000 hyd
004 xyz f 27 45000 bang
;
Run;
```



```
Data ds1;
Infile datalines Line=x;
Input id name$ sex$ age sal add$;
Put x;
/*Put _all_;*/
Datalines;
1
1
1
1
NOTE: The data set WORK.DS1 has 4 observations and 6 variables.
NOTE: DATA statement used (Total process time):
      real time           0.42 seconds
      cpu time            0.01 seconds

;
```

```
Data ds2;
Infile datalines Line=x;
Input #1 id name$
      #2 sex$ age
      #3 sal add$;
Put x;
Datalines;
001 abc
m 23
45000 hyd
002 def
f 34
67000 bang
003 mno
m 21
36000 hyd
004 xyz
f 27
45000 bang
;
Run;
```



```
Data ds2;
Infile datalines Line=x;
Input #1 id name$
        #2 sex$ age
        #3 sal add$;
Put x;
Datalines;
3
3
3
3
NOTE: The data set WORK.DS2 has 4 observations and 6 variables.
NOTE: DATA statement used (Total process time):
      real time           0.01 seconds
      cpu time            0.00 seconds

;
Run;
```

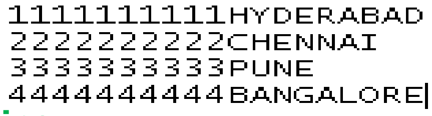
LENGTH=variable

Specifies a variable that SAS sets to the length of the current input line. SAS does not assign a value to variable until an INPUT statement executes. The LENGTH= variable is not written to the data set. Like automatic variable.

Syntax:- **Length=Varable_Name**

```
Data ds1;
Infile "C:\Documents and Settings\Administrator\Desktop\sample.txt" length=X;
Input id name$ sex$ age sal ;
Put x;
Run;
Data ds;
```

```
Infile "C:\Documents and Settings\Administrator\Desktop\sample1.txt" length=linelen;
Input firstvar 1-10 @;
      varlen=linelen-10;
Input @11 secondvar $varying500. varlen ;
Put linelen varlen;  *Observe log to get length of input line
Run;
```



```
1111111111HYDERABAD
2222222222CHENNAI
3333333333PUNE
4444444444BANGALORE
```

```
Data ds;
Infile datalines Length=x;
Input id name$ sex$ age sal add$;
Put x;
Datalines;
001 abc m 23 45000 hyd
002 def f 34 67000 bang
003 mno m 21 36000 hyd
004 xyz f 27 45000 bang
;
Run;
```

LINESIZE=line-size

Specifies the record length that is available to the INPUT statement.

Line size must between >=10 and <= 32767

Syntax:- **LINESIZE=line-size**

```
Data ds;
Infile "C:\Documents and Settings\Administrator\Desktop\sample.txt" Linesize=20;
Input id name$ sex$ age sal add$;
Run;
```

```
Data ds;
Infile "C:\Documents and Settings\Administrator\Desktop\sample.txt" Linesize=11;
Input id name$ sex$ age sal add$;
Run;
```

Linesize can’t work with cards/datalines.

LRECL=logical-record-length

Specifies the logical record length.

Syntax:- **LRECL=Logical record length**

Logical record length should be 1 to 32767

```
Data ds;
Infile "C:\Documents and Settings\Administrator\Desktop\sample.txt" Lrecl=256;
Input id name$ sex$ age sal add$;
Run;
```

lrecl can’t work with cards/datalines.

N=available-lines

Specifies the number of lines that are available to the input pointer at one time.

The highest value following a # pointer control in any INPUT statement in the DATA step. If you omit a # pointer control, then the default value is 1.

Syntax:- N=Input record lines

```
Data ds2;
Infile datalines n=3 line=lineno column=colno;
Input #1 id name$
        #2 sex$ age
        #3 sal add$;
Put lineno= colno=;
Datalines;
001 abc
m 23
45000 hyd
002 def
f 34
67000 bang
003 mno
m 21
36000 hyd
004 xyz
f 27
45000 bang
;
Run;
```

FILENAME STATEMENT

Associates a fileref for external file

Infile statement can also use a file reference using the file name statement.

Syntax:- FILENAME fileref 'external-file'

```
Filename krish
"C:\Documents and Settings\Administrator\Desktop\sample.txt";
```

```
Data ds1;
Infile krish;
Input id name$ sex$ age sal;
Run;
```

IMPORTING

```
Data _null_;
Set sashelp.class;
File krish;
Put name sex age height weight;
Run;
```

EXPORTING

FILE STATEMENT

It writes the information from SAS to external file.

Syntax:- File file-specification <options>

```
DATA DS;
INFILE DATALINES;
INPUT ID NAME$ SEX$ AGE SAL;
DATALINES;
001 ABC M 20 2500
002 DEF M 22 3000
003 XYZ F 21 5000
;
RUN;
```

File File-specification

Identifies an external file that the DATA step uses to write output from Put statement.

```
DATA _NULL_;
SET DS;
FILE "C:\Documents and Settings\Stansys\Desktop\sas\sample.rtf";
PUT @1 ID @5 NAME @10 SEX @14 AGE @18 SAL;
RUN;
```

File Filref

Specifies the fileref of an external file.

```
Filename krish
"C:\Documents and Settings\Administrator\Desktop\sas\sample.txt";
DATA _NULL_;
SET DS;
FILE krish;
PUT @1 ID @5 NAME @10 SEX @14 AGE @18 SAL;
RUN;
```

File print

It is a reserved fileref that directs the output into Output window, produced by PUT statements.

```
Data _null_;
Set sashelp.class;
File print;
Put name sex age height weight;
Run;
```

File log

It is a reserved fileref that directs the output into log, produced by any PUT statements.

```
Data _null_;
Set sashelp.class;
File log;
Put name sex age height weight;
Run;
```

Options

MOD

Writes the output lines after any existing lines in the file.

```
DATA _NULL_;
SET DS;
FILE "C:\Documents and Settings\Stansys\Desktop\sas\sample.rtf";
PUT @1 'ID' @5 'NAME' @10 'SEX' @14 'AGE' @18 'SAL';
RUN;
```

```
DATA _NULL_;
SET DS;
FILE "C:\Documents and Settings\Stansys\Desktop\sas\sample.rtf" MOD;
PUT @1 ID @5 NAME @10 SEX @14 AGE @18 SAL;
RUN;
```

DROPOVER

Discards data items that exceed the output line length
(as specified by the LINESIZE= or LRECL= options in the FILE statement).

```
Data _null_;
Set sashelp.class;
File print Dropover Linesize=20;
Put @1 name @10 sex @15 age @20 height @30 weight;
Run;
```

TITLES| NOTITLES

Controls whether currently defined titles are printed

FOOTNOTES | NOFOOTNOTES

Controls whether currently defined footnotes are printed

Default it is printing Titles | Footnotes
If you don't want to print use Notitles | Nofootnotes

```
Data _null_;
Set sashelp.class;
File print title footnotes ;
Put @1 name @10 sex @15 age @20 height @30 weight;
Title1 'Student Data';
Title2 'Stansys Software Solutions';
Footnote 'Generated by Mr.Krishna';
Run;
```

```
Data _null_;
Set sashelp.class;
File print notitles nofootnotes;
Put @1 name @10 sex @15 age @20 height @30 weight;
/*Title1 'Student Data';*/
/*Title2 'Stansys Software Solutions';*/
/*Footnote 'Generated by Mr.Krishna';*/
Run;
Filename krish
```

```
"C:\Documents and Settings\Administrator\Desktop\sas\sample.txt";
```

```
DATA _NULL_;
SET SASHELP.CLASS;
FILE KRISH DSD;
PUT NAME SEX AGE HEIGHT WEIGHT;
RUN;
```

```
DATA _NULL_;
SET SASHELP.CLASS;
FILE KRISH DLM='|';
PUT NAME SEX AGE HEIGHT WEIGHT;
RUN;
```

Note: All Infile statement options works in File statement also

ODS < = (ODS-suboptions) >

Specifies to use the Output Delivery System to format the output from a DATA step. It defines the structure of the data component and holds the results of the DATA step and binds that component to a table definition to produce an output object. ODS sends this object to all open ODS destinations like html ,pdf, rtf etc…

```
DATA _NULL_;
INFILE DATALINES;
INPUT ID NAME$ SEX$ AGE SAL;
FILE PRINT ODS;
PUT ID NAME SEX AGE SAL;
DATALINES;
001 ABC M 20 2500
002 DEF M 22 3000
003 XYZ F 21 5000
;
RUN;
```

Creating a Report with the DATA Step and the Default Table Definition

```
DATA _NULL_;
INFILE DATALINES;
INPUT ID NAME$ SEX$ AGE SAL;
FILE PRINT ODS;
PUT _ODS_;
DATALINES;
001 ABC M 20 2500
002 DEF M 22 3000
003 XYZ F 21 5000
;
RUN;
```

Producing ODS Output That Contains Selected Variables

```
DATA _NULL_;
INFILE DATALINES;
INPUT ID NAME$ SEX$ AGE SAL;
```

```
FILE PRINT ODS=(VARIABLES=(NAME SAL));
PUT _ODS_;
DATALINES;
001 ABC M 20 2500
002 DEF M 22 3000
003 XYZ F 21 5000
;
RUN;
```

Producing ODS Output That Contains Selected Variables into HTML format

```
ODS HTML FILE=
"C:\Documents and Settings\Administrator\Desktop\STANSYS\SAMPLE.HTML";
DATA _NULL_;
INFILE DATALINES;
INPUT ID NAME$ SEX$ AGE SAL;
FILE PRINT ODS=(VARIABLES=(NAME AGE SAL));
PUT _ODS_;
DATALINES;
001 ABC M 20 2500
002 DEF M 22 3000
003 XYZ F 21 5000
;
RUN;
```

Not only HTML we can send into all ods file formats.

ODS OUTPUT DESTINATIONS

If we don’t specify a destination, your output will be sent, by default, to the listing. The LISTING destination is what you see in the Output window.

FILE FORMAT	FILE EXTENSION	FILE DESCRIPTION
HTML	.HTM/.HTML	HYPER TEXT MARKUP LANGUAGE
PDF	.PDF	PORTABLE DOCUMENT FORMAT
RTF	.RTF	RICH TEXT FORMAT
EXCEL	.XLS/.XLSX	EXCEL
CSV	.CSV	COMMA SEPARATED VALUE
PS	.PS	POST SCRIPT
XML	.XML	EXTENSIBLE MARKUP LANGUAGE
MARKUP	.MARKUP	MARKUP LANGUAGE
CSS	.CSS	CASCADING STYLE SHEETS
LISTING		STANDARD SAS OUTPUT
PRINTER		HIGH RESOLUTION PRINTER OUTPUT

Assigning Attributes to Columns and Object label in ODS Output

```
ODS PDF FILE=
"C:\Documents and Settings\Administrator\Desktop\STANSYS\SAMPLE.PDF";

PROC FORMAT;
VALUE $GEN 'F'='FEMALE' 'M'='MALE';
RUN;
DATA _NULL_;
```

```
INFILE DATALINES;
INPUT ID NAME$ SEX$ AGE SAL;
FORMAT SEX $GEN.;
LABEL SEX=GENDER;
IF SEX='M';
FILE PRINT ODS=(OBJECTLABEL='MALE DATA');
PUT _ODS_;
DATALINES;
001 ABC M 20 2500
002 DEF M 22 3000
003 XYZ F 21 5000
;
RUN;
ODS PDF CLOSE;
```

STANSYS
SOFTWARE SOLUTIONS

Interview Questions

Q1) What is INFILE statement? Syntax? Example?

Q2) What is FILE statement? Syntax? Example?

Q3) What is FILENAME statement? Syntax? Example?

Q4) What are the INFILE options do you know?

Q5) What is DSD Option? Syntax? Example?

Q6) What is DLM Option? Syntax? Example?

Q7) What is DMSTR Option? Syntax? Example?

Q8) What is DLMSOPT Option? Syntax? Example?

Q9) What is FIRSTOBS Option? Syntax? Example?

Q10) What is OBS Option? Syntax? Example?

Q11) What is FLOWER Option? Syntax? Example?

Q12) What is MISSOVER Option? Syntax? Example?

Q13) What is TRUNCOVER Option? Syntax? Example?

Q14) What is STOPOVER Option? Syntax? Example?

Q15) What is SCANOVER Option? Syntax? Example?

Q16) What is END=Variable ? Syntax? Example?

Q17) What is EOF=Label ? Syntax? Example?

Q18) What is EOV=Variable ? Syntax? Example?

Q19) What is LRCEL ? Syntax? Example?

Q20) What is PAD ? Syntax? Example?

Q21) What is PRINT ? Syntax? Example?

Q22) What is _INFILE_=Variable? Syntax? Example?

Q23) What is LINE=Variable ? Syntax? Example?

Q24) What is LINESIZE=line-size Syntax? Example?

Q25) What is FILENAME=Variable ? Syntax? Example?

Q26) What is COLUMN=Variable ? Syntax? Example?

Q27) What is LENGTH=Variable ? Syntax? Example?

Q28) What is N=AVAILABLE LINES? Syntax? Example?

Q23) What is difference between DSD and DLM

Q24) What s difference between MISSOVER and TRUNCOVER?

Q25) Explain about below file statement options with syntax and example?
MOD, DROPOVER, TITLES, FOOTNOTES, FILE PRINT, FILE LOG & FILE PRINT ODS

Q25) Data is available in the location of "E:\SAS\IQ\INFILESTATEMENT"
tasks are available at TASK notepad in above same location?
read that data into SAS and write the program for each task ?

STANSYS
SOFTWARE SOLUTIONS