STAT 430
SAS Examples SAS9
==========

ssh abc@glue.umd.edu tap sas913 sas
https://www.statlab.umd.edu/sasdoc/sashtml/onldoc.htm

a. Reading external files using INFILE and INPUT (Ch 13)
b. Writing to an external file using FILE and PUT (Ch 13)
c. Subsetting (Ch 14)
d. Merging data sets (Ch 14)
e. Table look up (Ch 14)

a. Reading external files using INFILE and INPUT
==========

To use INFILE we give the complete path on wam
'/homes/bnk/driver' where the data are in "driver".

```
OPTION PS=45 LS=70;

DATA LOGISTIC;
INFILE '/homes/bnk/driver';
INPUT ACCIDENT AGE VISION DRIVE_ED;
DATALINES;  <----- Or RUN!!!!

PROC MEANS DATA=LOGISTIC MAXDEC=1;
VAR ACCIDENT AGE VISION DRIVE_ED;
RUN;
```

The MEANS Procedure

Variable	N	Mean	Std Dev	Minimum	Maximum
ACCIDENT	45	0.6	0.5	0.0	1.0
AGE	45	45.2	19.2	16.0	75.0
VISION	45	0.5	0.5	0.0	1.0
DRIVE_ED	45	0.4	0.5	0.0	1.0

Another example: File "mydata" in "/homes/bnk/mydata" on wam:

```
CONTROL 12 17 19
TREAT   23 25 29
CONTROL 19 18 16
TREAT   22 22 29
```

```
DATA EX1;
INFILE '/homes/bnk/mydata'; <--- Use quotes!!!
INPUT GROUP $ X Y Z;
RUN;

PROC MEANS DATA=EX1 N MEAN STD MAXDEC=1;
VAR X Y Z;
RUN;
```

The MEANS Procedure

Variable	N	Mean	Std Dev
X	4	19.0	5.0
Y	4	20.5	3.7
Z	4	23.3	6.8

Can create an alias for "mydata" using FILENAME:

```
DATA EX1B;
FILENAME GEORGE '/homes/bnk/mydata';
INFILE GEORGE;  <--- No quotes!!!
INPUT GROUP $ X Y Z;
RUN;

PROC MEANS DATA=EX1B N MEAN STD MAXDEC=1;
VAR X Y Z;
RUN;
```

Then get the same table as above!!!

INFILE Options
==========

1. Option End=variable name

First create a new data set "mydata1"

```
CONTROL 12 15 22
TREAT   34 23 12
CONTROL 77 53 69
TREAT   22 21 20
TREAT   10 11 12
```

Option: END="variable name". Will set the "variable name" to 0 (false) until INPUT statement finished to read the last record. Can read from several files!!! In the next example first read from "mydata" then from "mydata1".

```
DATA EX2E;
IF TESTEND NE 1 THEN INFILE '/homes/bnk/mydata' END=TESTEND;
ELSE INFILE '/homes/bnk/mydata1';
INPUT GROUP $ X Y Z;
RUN;

PROC print DATA=EX2E;
RUN;
```

Obs	GROUP	X	Y	Z
1	CONTROL	12	17	19
2	TREAT	23	25	29
3	CONTROL	19	18	16
4	TREAT	22	22	29
5	CONTROL	12	15	22
6	TREAT	34	23	12
7	CONTROL	77	53	69
8	TREAT	22	21	20
9	TREAT	10	11	12

```
PROC print DATA=EX2E;
ID GROUP;
run;
```

GROUP	X	Y	Z
CONTROL	12	17	19
TREAT	23	25	29
CONTROL	19	18	16
TREAT	22	22	29
CONTROL	12	15	22
TREAT	34	23	12
CONTROL	77	53	69
TREAT	22	21	20
TREAT	10	11	12

```
PROC print DATA=EX2E;
VAR X Y Z;
RUN;
```

Obs	X	Y	Z
1	12	17	19
2	23	25	29
3	19	18	16
4	22	22	29
5	12	15	22
6	34	23	12
7	77	53	69
8	22	21	20
9	10	11	12

```
PROC MEANS DATA=EX2E N MEAN STD MAXDEC=1;
VAR X Y Z;
RUN;
```

The MEANS Procedure

Variable	N	Mean	Std Dev
X	9	25.7	20.6
Y	9	22.8	12.1
Z	9	25.3	17.5

Now let's see what happens when we use "IF TESTEND NE 0"
First let's see what is in mydata1.

```
DATA EX3A;
IF TESTEND NE 0 THEN INFILE '/homes/bnk/mydata' END=TESTEND;
ELSE INFILE '/homes/bnk/mydata1';
INPUT GROUP $ X Y Z;
RUN;

PROC print DATA=EX3A;
ID GROUP;
run;
```

GROUP	X	Y	Z
CONTROL	12	15	22
TREAT	34	23	12
CONTROL	77	53	69
TREAT	22	21	20
TREAT	10	11	12

Get only the second data set "mydata1" but not "mydata"!!!!

2. Option MISSOVER

When we have missing data like file mydata2

```
CONTROL 11
TREAT   34 23 12
CONTROL 18
TREAT   11 12
TREAT    5  6  7
```

If use this: Get nonsense!

```
DATA EX4;
INFILE '/homes/bnk/mydata2';
INPUT GROUP $ X Y Z;
RUN;

PROC print DATA=EX4;
ID GROUP;
run;
```

```
 GROUP      X    Y     Z

CONTROL    11    .    34
CONTROL    18    .    11
TREAT       5    6     7
```

Now use MISSOVER:

```
DATA EX4;
INFILE '/homes/bnk/mydata2' MISSOVER;
INPUT GROUP $ X Y Z;
RUN;

PROC print DATA=EX4;
ID GROUP;
run;
```

Now get the correct file back:

```
 GROUP      X     Y     Z

CONTROL    11     .     .
TREAT      34    23    12
CONTROL    18     .     .
TREAT      11    12     .
TREAT       5     6     7
```

```
Note: It cannot distinguish from the following!!!!
CONTROL 11
TREAT   34 23 12
CONTROL       18
TREAT   11 12
TREAT    5  6  7
```

Now read "mydata" and "mydata2" (the last has mising values):

```
DATA EX4a;
IF TESTEND NE 1 THEN INFILE '/homes/bnk/mydata' END=TESTEND;
ELSE INFILE '/homes/bnk/mydata2' MISSOVER;
INPUT GROUP $ X Y Z;
RUN;

PROC print DATA=EX4a;
RUN;
```

```
Obs     GROUP      X     Y     Z

 1     CONTROL    12    17    19
 2     TREAT      23    25    29
 3     CONTROL    19    18    16
 4     TREAT      22    22    29
 5     CONTROL    11     .     .
 6     TREAT      34    23    12
 7     CONTROL    18     .     .
 8     TREAT      11    12     .
 9     TREAT       5     6     7
```

Now read all 3 files: mydata,mydata1,mydata2:

```
DATA EX4a;
IF TESTEND NE 1 THEN INFILE '/homes/bnk/mydata' END=TESTEND;
else if TEST NE 1 then infile '/homes/bnk/mydata1' end= TEST;
ELSE INFILE '/homes/bnk/mydata2' MISSOVER;
INPUT GROUP $ X Y Z;
RUN;

PROC print DATA=EX4a;
RUN;
```

Obs	GROUP	X	Y	Z
1	CONTROL	12	17	19
2	TREAT	23	25	29
3	CONTROL	19	18	16
4	TREAT	22	22	29
5	CONTROL	12	15	22
6	TREAT	34	23	12
7	CONTROL	77	53	69
8	TREAT	22	21	20
9	TREAT	10	11	12
10	CONTROL	11	.	.
11	TREAT	34	23	12
12	CONTROL	18	.	.
13	TREAT	11	12	.
14	TREAT	5	6	7

3. Option PAD

When use column pointers to prevent problems with short records.

```
Note: I now modify "mydata2" as follows:

"mydata2" contains GROUP X Y Z:
CONTROL 11
TREAT   34 23 12
CONTROL       18<---- Moved this to a Z-column
TREAT   11 12
TREAT    5  6  7


First the case without column pointers:

DATA EX5;
INFILE '/homes/bnk/mydata2' PAD;
INPUT GROUP $ X Y Z;
RUN;

PROC print DATA=EX5;
RUN;



Obs     GROUP      X    Y     Z

 1     CONTROL    11    .    34
 2     CONTROL    18    .    11
 3     TREAT       5    6     7

         The MEANS Procedure

Variable    N            Mean         Std Dev
-------------------------------------------
X           3            11.3             6.5
Y           1             6.0              .
Z           3            17.3            14.6
-------------------------------------------

Thus we get nonsense!!!
So, now use column pointers and PAD:
```

```
DATA EX5;
INFILE '/homes/bnk/mydata2' PAD;
INPUT GROUP $ 1-7
X 9-10
Y 12-13
Z 15-16;
RUN;

PROC print DATA=EX5;
RUN;

OK!!!!!

Obs     GROUP      X     Y     Z

 1     CONTROL    11     .     .
 2     TREAT      34    23    12
 3     CONTROL     .     .    18
 4     TREAT      11    12     .
 5     TREAT       5     6     7

PROC MEANS DATA=EX5 N MEAN STD MAXDEC=1;
VAR X Y Z;
RUN;

OK!!!!!
            The MEANS Procedure

Variable    N            Mean         Std Dev
-------------------------------------------
X           4            15.3            12.8
Y           3            13.7             8.6
Z           3            12.3             5.5
-------------------------------------------


DATA EX5a;
IF TESTEND NE 1 THEN INFILE '/homes/bnk/mydata' END=TESTEND;
ELSE INFILE '/homes/bnk/mydata2' PAD;
```

```
INPUT GROUP $ 1-7
X 9-10
Y 12-13
Z 15-16;
RUN;


PROC print DATA=EX5a;
RUN;
```

OK!!!

Obs	GROUP	X	Y	Z
1	CONTROL	12	17	19
2	TREAT	23	25	29
3	CONTROL	19	18	16
4	TREAT	22	22	29
5	CONTROL	11	.	.
6	TREAT	34	23	12
7	CONTROL	.	.	18
8	TREAT	11	12	.
9	TREAT	5	6	7

Now read all 3 files: Note that we must use a different variable name such as TEST1, TEST2, etc. for each data sets, but always use "NE 1".

```
DATA EX44;
     IF TEST1 NE 1 THEN INFILE '/homes/bnk/mydata' END=TEST1;
ELSE IF TEST2 NE 1 THEN INFILE '/homes/bnk/mydata1' END=TEST2;
ELSE INFILE '/homes/bnk/mydata2' PAD;
INPUT GROUP $ 1-7
X 9-10
Y 12-13
Z 15-16;
RUN;

PROC print DATA=EX44;
RUN;
```

```
Obs     GROUP      X     Y     Z

 1    CONTROL    12    17    19
 2    TREAT      23    25    29
 3    CONTROL    19    18    16
 4    TREAT      22    22    29
 5    CONTROL    12    15    22
 6    TREAT      34    23    12
 7    CONTROL    77    53    69
 8    TREAT      22    21    20
 9    TREAT      10    11    12
10    CONTROL    11     .     .
11    TREAT      34    23    12
12    CONTROL     .     .    18
13    TREAT      11    12     .
14    TREAT       5     6     7
```

b. Writing to an external file using FILE and PUT

==========

To read external files use INFILE and INPUT.
To write data into an external file use FILE and PUT.
Here is an example.

```
DATA EX6;
INFILE '/homes/bnk/mydata'; ***Input file;
FILE   '/homes/bnk/mydatN'; ***Output file;
INPUT GROUP $ X Y Z;
TOTAL= SUM (OF X Y Z);
PUT GROUP $ 1-10 @12 (X Y Z TOTAL) (5.);
RUN;

Let's see what was created:
DATA EX6a;
INFILE   '/homes/bnk/mydatN'; ***Now it is input file;
INPUT GROUP $ 1-10 @12 (X Y Z TOTAL) (5.);
RUN;
```

```
PROC print DATA=EX6a;
RUN;
```

	GROUP	X	Y	Z	TOTAL
1	CONTROL	12	17	19	48
2	TREAT	23	25	29	77
3	CONTROL	19	18	16	53
4	TREAT	22	22	29	73

More complicated example: Create a new data set from 3 files.

```
DATA EX44;
     IF TEST1 NE 1 THEN INFILE '/homes/bnk/mydata' END=TEST1;
ELSE IF TEST2 NE 1 THEN INFILE '/homes/bnk/mydata1' END=TEST2;
ELSE INFILE '/homes/bnk/mydata2' PAD;
FILE   '/homes/bnk/mydatN'; ***Output file;
INPUT GROUP $ 1-7
X 9-10
Y 12-13
Z 15-16;
TOTAL= SUM (OF X Y Z);
PUT GROUP $ 1-10 @12 (X Y Z TOTAL) (5.);
RUN;
```

Let's see what was created:

```
DATA EX44a;
INFILE   '/homes/bnk/mydatN';
INPUT GROUP $ X Y Z TOTAL;
RUN;

PROC print DATA=EX44a;
RUN;
```

Obs	GROUP	X	Y	Z	TOTAL
1	CONTROL	12	17	19	48
2	TREAT	23	25	29	77
3	CONTROL	19	18	16	53
4	TREAT	22	22	29	73

```
  5    CONTROL    12    15    22      49
  6    TREAT      34    23    12      69
  7    CONTROL    77    53    69     199
  8    TREAT      22    21    20      63
  9    TREAT      10    11    12      33
 10    CONTROL    11     .     .      11
 11    TREAT      34    23    12      69
 12    CONTROL     .     .    18      18
 13    TREAT      11    12     .      23
 14    TREAT       5     6     7      18
```

NOTE: There is "." for missing values!!!

c. Subsetting
==========

Example: We take a subst from dataset WOMEN using "SET".

```
DATA WOMEN;
INPUT GENDER $ AGE;
DATALINES;
M 60
M 90
F 56
M 44
F 23
F 55
F 59
M 38
F 77
;

PROC print DATA=WOMEN;
RUN;
```

```
Obs    GENDER    AGE
 1       M        60
 2       M        90
 3       F        56
 4       M        44
 5       F        23
```

```
 6       F        55
 7       F        59
 8       M        38
 9       F        77
```

Now create a new subset called FEM from old set WOMEN:

```
DATA FEM;  <-- New set
SET WOMEN; <-- Old set
IF GENDER = 'F';
RUN;

PROC print DATA=FEM;
RUN;
```

```
Obs    GENDER    AGE
 1       F        56
 2       F        23
 3       F        55
 4       F        59
 5       F        77
```

More complex subset:

```
DATA OLDFEM;
SET WOMEN;
IF GENDER = 'F' AND AGE > 57;  (Or: WHERE GENDER = 'F' AND AGE > 57;)
RUN;

PROC print DATA=OLDFEM;
RUN;
```

```
Obs    GENDER    AGE
 1       F        59
 2       F        77
```

```
Another subset:
DATA MIDAGE;
SET WOMEN;
IF AGE > 50 & AGE < 65;  ###When used "AND" had problems!!!
RUN;

PROC print DATA=MIDAGE;
RUN;

Obs    GENDER    AGE
 1       M        60
 2       F        56
 3       F        55
 4       F        59

Another subset:
DATA EXTREME;
SET WOMEN;
IF AGE < 30 OR  AGE > 70;
RUN;

PROC print DATA=EXTREME;
RUN;

Obs    GENDER    AGE
 1       M        90
 2       F        23
 3       F        77


In general can substitute WHERE for IF. But WHERE can be used
in SAS procedures. For example,
we use WHERE in PROC FREQ as a means for subsetting:


DATA WOMEN;
INPUT GENDER $ AGE RACE $ INCOME;
DATALINES;
M 60 B  50000
```

```
M 90 C  10000
F 56 C 100000
M 44 B  80000
F 23 B 123444
F 55 C  87000
F 59 C 200001
M 38 C 132000
F 77 B 120000
;


PROC FREQ DATA=women;
where gender= 'F';
tables race income;
run;
```

The FREQ Procedure

RACE	Frequency	Percent	Cumulative Frequency	Cumulative Percent
B	2	40.00	2	40.00
C	3	60.00	5	100.00

INCOME	Frequency	Percent	Cumulative Frequency	Cumulative Percent
87000	1	20.00	1	20.00
100000	1	20.00	2	40.00
120000	1	20.00	3	60.00
123444	1	20.00	4	80.00
200001	1	20.00	5	100.00

More complex necessitates SORT:

```
proc sort DATA=women;
```

```
by race;
run;

PROC FREQ DATA=women;
where gender= 'F';
by race ;
tables  income age;
run;
```

-------- RACE=B --------

The FREQ Procedure

INCOME	Frequency	Percent	Cumulative Frequency	Cumulative Percent
120000	1	50.00	1	50.00
123444	1	50.00	2	100.00

AGE	Frequency	Percent	Cumulative Frequency	Cumulative Percent
23	1	50.00	1	50.00
77	1	50.00	2	100.00

---------- RACE=C ----------

The FREQ Procedure

INCOME	Frequency	Percent	Cumulative Frequency	Cumulative Percent
87000	1	33.33	1	33.33
100000	1	33.33	2	66.67
200001	1	33.33	3	100.00

AGE	Frequency	Percent	Cumulative Frequency	Cumulative Percent
55	1	33.33	1	33.33

```
 56           1       33.33             2        66.67
 59           1       33.33             3       100.00
```

d. Merging data sets
=====================

Will combine or MERGE the following 2 data sets.

```
DATA MASTER;
INPUT ID  NAME $;
DATALINES;
123 CODY
987 SMITH
111 GREG
222 HAMER
777 JACK
;

DATA TEST;
INPUT ID SCORE;
DATALINES;
123 89
987 55
111 78
222 84
;
```

Note: ID and score for JACK are missing. This is OK!

```
PROC SORT DATA=MASTER;
BY ID;
RUN;

PROC SORT DATA=TEST;
BY ID;
RUN;
```

Now merging the data sets MASTER and TEST together:

```
DATA COMB;
MERGE MASTER TEST;
BY ID;
RUN;

PROC PRINT DATA=COMB;
RUN;


Obs     SS    NAME     SCORE
 1     111    GREG       78
 2     123    CODY       89
 3     222    HAMER      84
 4     777    JACK        .  <--- OK!!! Score was missing.
 5     987    SMITH      55


e. Table look up
==================
One can pull information from a data set based on some criteria
and add that information to the current data set. Here is an example
regarding ID,YEAR,WHITE BLOOD COUNT (WBC).

DATA WORKER;
INPUT ID YEAR WBC;
DATALINES;
1 1940 6000
2 1940 8000
3 1940 9000
1 1941 6500
2 1941 8500
3 1941 8900
;

Have also exposure data (the look up table):

DATA EXP;
INPUT YEAR EXPSR;
DATALINES;
1940 200
```

```
1941 150
1942 100
1943  80
;
```

We wish to to add the exposure to each observation in the WORKER data set using a MERGE statement as follows:

```
PROC SORT DATA=WORKER;
BY YEAR;
RUN;

PROC SORT DATA=EXP;
BY YEAR;
RUN;

DATA COMBINE;
MERGE WORKER (IN = INWORK) EXP;
BY YEAR;
IF INWORK;
RUN;

PROC PRINT DATA=COMBINE;
RUN;
```

Obs	ID	YEAR	WBC	EXPSR
1	1	1940	6000	200
2	2	1940	8000	200
3	3	1940	9000	200
4	1	1941	6500	150
5	2	1941	8500	150
6	3	1941	8900	150

Can do the same using BY YEAR and WORK assignment etc.
Two types of work: MIXER and SPREAD=SPREADER.

```
OPTION PS=45 LS=70;

DATA WORKER;
INPUT ID YEAR WORK $ WBC;
```

```
DATALINES;
1 1940 MIXER    6000
2 1940 SPREAD   8000
3 1940 MIXER    9000
1 1941 MIXER    8500
2 1941 MIXER    8500
3 1941 SPREAD   8900
;
```

Here is the look up table:

```
DATA EXP;
INPUT YEAR WORK $ EXPOSURE;
DATALINES;
1940 MIXER    190
1940 SPREAD   200
1941 MIXER    140
1941 SPREAD   150
1942 MIXER     90
1942 SPREAD   100
1943 MIXER     70
1943 SPREAD    80
;

PROC SORT DATA=WORKER;
BY YEAR WORK;
RUN;

PROC SORT DATA=EXP;
BY YEAR WORK;
RUN;

DATA COMBINE;
MERGE WORKER (IN = INWORK) EXP;
BY YEAR  WORK;
IF INWORK;
RUN;

PROC PRINT DATA=COMBINE;
VAR ID YEAR WORK WBC EXPOSURE;
```

```
RUN;
```

Obs	ID	YEAR	WORK	WBC	EXPOSURE
1	1	1940	MIXER	6000	190
2	3	1940	MIXER	9000	190
3	2	1940	SPREAD	8000	200
4	1	1941	MIXER	8500	140
5	2	1941	MIXER	8500	140
6	3	1941	SPREAD	8900	150

Note: If we forget to remember that WORK is character data and dont use "WORK $", then we get a completely wrong output as given below:

Obs	ID	YEAR	WORK	WBC	EXPOSURE	
1	1	1940	.	6000	190	
2	2	1940	.	8000	200	
3	3	1940	.	9000	200	<-- wrong!
4	1	1941	.	8500	140	
5	2	1941	.	8500	150	<-- wrong!
6	3	1941	.	8900	150	