You have some data that look as follows....

```
data study;
input
id    :   $3.
visit :   mmddyy.
chol
;
format visit mmddyy10.;
datalines;
001    10/15/2004      200
002    10/15/2004      200
003    10/15/2004      300
004    10/15/2004      275
005    10/15/2004      250
002    11/10/2004      175
002    11/10/2004      175
002    11/10/2004      175
002    11/10/2004      195
004    11/13/2004      275
003    11/14/2004      280
004    12/14/2004      275
;
run;
```

Each record in the data file has: an ID number; a date of visit; a cholesterol measurement. Some subjects have a single occurrence (IDs 001 and 005) while others have multiple occurrences. The following are some tasks that can be accomplished using FIRST. and LAST. variables. Use of these variables requires a SET statement + a BY statement, in this case...

```
set study;
by id;
```

Whenever you use a BY statement, SAS requires that the data set being used be sorted according to all the variables in the BY statement. Also, since you will be looking for first time you saw a subject, last time, etc., you also sort the data by VISIT within each ID...

```
proc sort data=study;
by id visit;
run;

proc print data=study;
run;
```

The data set now looks as follows...

Obs	id	visit	chol
1	001	10/15/2004	200
2	002	10/15/2004	200
3	002	11/10/2004	175
4	002	11/10/2004	175
5	002	11/10/2004	175
6	002	11/10/2004	195
7	003	10/15/2004	300
8	003	11/14/2004	280
9	004	10/15/2004	275
10	004	11/13/2004	275
11	004	12/14/2004	275
12	005	10/15/2004	250

One way to learn about FIRST. and LAST. variables is to print the data set showing the their values within each observation. Since FIRST. and LAST. variables ONLY EXIST FOR THE DURATION OF THE DATA STEP and are NOT ADDED TO THE DATA SET, you must create new variables that contain the values if the FIRST. and LAST. variables...

```
data fl;
set study;
by id;
first_id = first.id;
last_id  = last.id;
label
first_id = 'first.id'
last_id  = 'last.id'
;
run;

proc print data=fl label;
run;
```

Obs	id	visit	chol	first.id	last.id
1	001	10/15/2004	200	1	1
2	002	10/15/2004	200	1	0
3	002	11/10/2004	175	0	0
4	002	11/10/2004	175	0	0
5	002	11/10/2004	175	0	0
6	002	11/10/2004	195	0	1
7	003	10/15/2004	300	1	0
8	003	11/14/2004	280	0	1
9	004	10/15/2004	275	1	0
10	004	11/13/2004	275	0	0
11	004	12/14/2004	275	0	1
12	005	10/15/2004	250	1	1

The values of the FIRST. and LAST. variables are helpful in answering the questions about your data. For example, if you are looking for observations in which the variable FIRST.ID has a value of 1 (the first observation within each ID), you can use either...

```
if first.id;
```

or...

```
if first.id eq 1;
```

or...

```
if first.id ne 0;
```

#1 Create a new data set that contains one observation per ID --- *the FIRST time each ID participated in your study*.

look for observations where FIRST.ID has a value of 1

```
data study_f;
set study;
by id;
if first.id;
run;
```

```
FIRST VISIT
Obs    id          visit    chol
 1     001    10/15/2004     200
 2     002    10/15/2004     200
 3     003    10/15/2004     300
 4     004    10/15/2004     275
 5     005    10/15/2004     250
```

#2 Create a new data set that contains one observation per ID --- *the LAST time each ID participated in your study*.

look for observations where LAST.ID has a value of 1

```
data study_l;
set study;
by id;
if last.id;
run;
```

```
LAST VISIT
Obs    id          visit    chol
 1     001    10/15/2004     200
 2     002    11/10/2004     195
 3     003    11/14/2004     280
 4     004    12/14/2004     275
 5     005    10/15/2004     250
```

#3 Create a new data set that contains two observations per ID --- *the FIRST and LAST time each ID participated in your study*.

look for observations where FIRST.ID or LAST.ID has a value of 1

```
* first and last time you saw each ID;
data study_fl;
set study;
by id;
if first.id or last.id;
run;
```

```
FIRST AND LAST VISIT
Obs    id          visit    chol
 1     001    10/15/2004     200
 2     002    10/15/2004     200
 3     002    11/10/2004     195
 4     003    10/15/2004     300
 5     003    11/14/2004     280
 6     004    10/15/2004     275
 7     004    12/14/2004     275
 8     005    10/15/2004     250
```

#4 Create two data sets --- *one with all subjects who only have ONE observation in the data set, one with subjects who have MULTIPLE observations in the data set*.

identify ONE observation subjects as those with both FIRST.ID and LAST.ID having the value 1
all others are MULTIPLE observation subjects

```
data single multiple;
set study;
by id;
if first.id and last.id then output single;
else output multiple;
run;
```

```
SINGLE VISIT
Obs    id          visit    chol
 1     001    10/15/2004     200
 2     005    10/15/2004     250
```

```
MULTIPLE VISITS
Obs    id          visit    chol
  1    002    10/15/2004     200
  2    002    11/10/2004     175
  3    002    11/10/2004     175
  4    002    11/10/2004     175
  5    002    11/10/2004     195
  6    003    10/15/2004     300
  7    003    11/14/2004     280
  8    004    10/15/2004     275
  9    004    11/13/2004     275
 10    004    12/14/2004     275
```

#5 Create one data set from the original data set STUDY --- *the FIRST time each ID participated in your study for only those subjects with multiple visits.*

```
data study_fm;
set study;
by id;
if first.id and not last.id;
run;
```

```
FIRST VISIT OF MULTIPLE VISITS
Obs    id          visit    chol
 1     002    10/15/2004     200
 2     003    10/15/2004     300
 3     004    10/15/2004     275
```

NOTE: You could have used the new data set MULTIPLE created in #4 and just specify...

```
if first.id;
```

#6 There should not be any repeated dates with any ID --- *create a data set with repeated dates within any of the IDs.*

Once again, it is helpful to know the values of the FIRST. and LAST. variables in the data step...

```
data fl;
set study;
by id visit;
first_id = first.id;
last_id  = last.id;
first_visit = first.visit;
last_visit  = last.visit;
label
first_id = 'first.id'
last_id  = 'last.id'
first_visit = 'first.visit'
last_visit  = 'last.visit'
;
run;

proc print data=fl label;
run;
```

Obs	id	visit	chol	first.id	last.id	first. visit	last. visit
1	001	10/15/2004	200	1	1	1	1
2	002	10/15/2004	200	1	0	1	1
3	002	11/10/2004	175	0	0	1	0
4	002	11/10/2004	175	0	0	0	0
5	002	11/10/2004	175	0	0	0	0
6	002	11/10/2004	195	0	1	0	1
7	003	10/15/2004	300	1	0	1	1
8	003	11/14/2004	280	0	1	1	1
9	004	10/15/2004	275	1	0	1	1
10	004	11/13/2004	275	0	0	1	1
11	004	12/14/2004	275	0	1	1	1
12	005	10/15/2004	250	1	1	1	1

How can you identify repeated dates within each ID...

```
data repeats;
set study;
by id visit;
if not (first.visit and last.visit);
run;
```

REPEATED VISITS WITHIN AN ID

Obs	id	visit	chol
1	002	11/10/2004	175
2	002	11/10/2004	175
3	002	11/10/2004	175
4	002	11/10/2004	195

#7 What is the difference in the cholesterol reading between the first and last time I saw each subject? *How can I compute the difference when the values of the variable occur in different observations?*

Assume that you have sorted the data set by ID and by ascending values of VISIT within each ID so that the data set STUDY looks like this ...

```
Obs    id          visit    chol
  1    001    10/15/2004     200
  2    002    10/15/2004     200
  3    002    11/10/2004     175
  4    002    11/10/2004     175
  5    002    11/10/2004     175
  6    002    11/10/2004     195
  7    003    10/15/2004     300
  8    003    11/14/2004     280
  9    004    10/15/2004     275
 10    004    11/13/2004     275
 11    004    12/14/2004     275
 12    005    10/15/2004     250
```

Given the above question, what would you do if you were to compute the difference without SAS? One approach would be to find the first visit for a subject and write down the ID and value of CHOL (cholesterol). Then you would find the last visit for that subject, take the value of CHOL for that date and subtract the value of CHOL you had written down from the first visit. The task is to replicate that logic in a data step.

```
data diff;
retain firstchol; ①
set study;
by id; ②
if first.id then firstchol = chol; ③
if last.id then do; ④
   diffchol = chol - firstchol; ⑤
   output; ⑥
end;
run;
```

Ignore the RETAIN ① statement for now. The data set STUDY is read with a SET statement plus a BY variable ②. That allows you to easily find the first and last visit for each subject using FIRST. ID and LAST.ID. When the first observation for an ID is encountered, you 'write down' the value of CHOL by storing that value in a new variable name FIRSTCHOL ③ (you could use any name you want for that new variable, but since it is the first cholesterol value for a subject, FIRSTCHOL is used). When the last observation for and ID is encountered ④ you want to do two things: first, subtract the value of CHOL (that you 'wrote down' the first time you saw a subject) from the current value of CHOL (corresponding to the last time you saw the same subject) ⑤; second, you want to write an observation to the data set only after you have computed the difference in choloesterol ⑥.

Since you want to do two things when you read that last observation, you use DO and END statements. The on-line SAS documentation says ... *The DO statement specifies that the statements following the DO statement are executed as a group until a matching END statement appears*.

You could have written the following in place of the statements used above ...

```
if last.id then diffchol = chol - firstchol;
if last.id then output;
```

but DO and END statements are a common approach to executing two or more statements based on whether an IF statement is true or false.

The SAS code produces the following data set ...

```
DATA SET DIFF:  DIFFERENCE IN CHOLESTEROL
Obs    id     firstchol    chol    diffchol         visit
 1     001       200        200         0       10/15/2004
 2     002       200        195        -5       11/10/2004
 3     003       300        280       -20       11/14/2004
 4     004       275        275         0       12/14/2004
 5     005       250        250         0       10/15/2004
```

Before discussing the output, we go back to the data step and look at that RETAIN statement ①. What would data set DIFF look like with the RETAIN statement ...

```
DATA SET DIFF:  DIFFERENCE IN CHOLESTEROL, NO RETAIN STATEMENT
Obs    id     firstchol    chol    diffchol         visit
 1     001       200        200        0        10/15/2004
 2     002         .        195        .        11/10/2004
 3     003         .        280        .        11/14/2004
 4     004         .        275        .        12/14/2004
 5     005       250        250        0        10/15/2004
```

What is different? For each subject that was seen more than once, the value of FIRSTCHOL and DIFFCHOL is missing. Why? Each time SAS returns to the top of a data step, prior to reading another observation, some 'housekeeping' is done. That task includes setting to missing the value of any variable that is created within the data step. Yes, (of course) there are exceptions, but any variable created in a statement such as ...

```
firstchol = chol;
```

(where a variable is assigned a value using an equals sign) is set to missing at the beginning of each pass through the data step. A RETAIN statement tells SAS that you do not want the value of one or more variables set to missing. Since FIRSTCHOL was set to missing for subjects 2, 3, and 5, the equation ...

```
diffchol = chol - firstchol;
```

produces a missing result (remember, any missing data on the right of the equals sign produces a missing result). So, even without the RETAIN statement, why are the values of FIRSTCHOL and DIFFCHOL not missing for subjects 1 and 5? Since there is only one observation for each of those subjects, it is both the FIRST.ID and LAST.ID. Thus, SAS completes all the tasks for subjects 1and 5 in one pass through the data step ... there is no return to the top of the data step for these two subjects and no 'set to missing' for the variable FIRSTCHOL.

#8 What is the difference in the cholesterol reading between the first and last time I saw each subject and how many times has each subject been seen? *How can I compute the difference when the values of the variable occur in different observations and how can I count the number of occurrences of each individual ID ?*

In looking at the output at the top of the page, for subjects 1, 4, and 5 it appears that there was no change in cholesterol between the first and last visits. However, subjects 1 and 5 only had one measurement that served as both the first and last cholesterol readings. Subject 4 was seen three times and had no change in cholesterol. How could you add the number of visits to the output?

```
data diff;
retain firstchol;
set study;
by id;
if first.id then do; ①
   firstchol = chol;
   nvisits = 0; ②
end;
nvisits + 1; ③
if last.id then do;
   diffchol = chol - firstchol;
   output;
end;
run;
```

Now, the first time we encounter each subject, we want to do two things (use DO-END ①): first, 'write down' the value of CHOL by storing that value in a new variable name FIRSTCHOL; next, assign a value of zero to a new variable NVISITS that will be used to count the number of visits for each subject ②. The statement ...

```
nvisits + 1;
```

adds one to the value of the variable NVISITS at each pass through data step ③ (notice that the statement is not included as part of any statement or set of statements that checks if an observation is the first or last for a subject). Since the value of NVISITS is set to zero the first time we see a subject, NVISITS will count the number of visits for each subject.

Maybe you noticed (or maybe not) that NVISITS is not listed in the RETAIN statement even though it is created in the data step. There are several ways that you can count the number of visits and two of them are ...

```
nvisits + 1;
```

```
nvisits = nvisits + 1;
```

Notice the difference ... in the first statement THERE IS NO EQUALS SIGN. This is one of the exceptions referred to in the discussion of SAS 'housekeeping' and setting values to missing. A variable that is assigned a value in a statement of the form ...

```
nvisits + 1;
```

has an IMPLIED RETAIN and the value of the leftmost variable in the statement (here it is NVISITS) is automatically retained. A variable that is assigned a value in a statement of the form ...

```
nvisits = nvisits + 1;
```

would require listing in a RETAIN statement since THERE IS AN EQUALS SIGN and variables assigned a value within a data step using and equation with an equals sign are set to missing during 'housekeeping'.

#9 What is the difference in the cholesterol reading between the first and last time I saw each subject, how many times has each subject been seen, and how many days has each subject been in the study? *How can I compute two (not one) differences when the values of the variable occur in different observations and how can I count the number of occurrences of each individual ID?*

Keeping track of more than one variable across observations merely requires a bit more SAS code once you understand the principles covered in the last couple of examples.

```
data diff;
retain firstchol firstvisit; ①
set study;
by id;
if first.id then do;
   firstchol = chol;
   firstvisit = visit; ②
   nvisits = 0;
end;
nvisits + 1;
if last.id then do;
   diffchol = chol - firstchol;
   diffdays = visit - firstvisit; ③
   output;
end;
format firstvisit mmddyy10.; ④
run;
```

Now, you want to keep track of two values the first time you see a subject, the cholesterol (FIRSTCHOL) and the date (FIRSTVISIT). Just as you 'wrote down' the value of CHOL the firs time you saw a subject by assigning the value to a new variable (FIRSTCHOL), you do the same thing for the date of the first visit and 'write down' that date by storing it in another new variable, FIRSTVISIT ②. Since FIRSTVISIT is created within the data step and assigned a value using an equals sign, it must be added to the RETAIN statement ①. The last time a subject is seen, the difference in days can is computed ③. Finally, a format is assigned to FIRSTVISIT since it is a date.

```
DATA SET DIFF:  DIFFERENCE IN CHOLESTEROL, NUMBER OF VISITS, AND DAYS IN THE STUDY
Obs    id     nvisits    firstchol    chol    diffchol    firstvisit         visit    diffdays
 1     001       1          200        200         0      10/15/2004    10/15/2004        0
 2     002       5          200        195        -5      10/15/2004    11/10/2004       26
 3     003       2          300        280       -20      10/15/2004    11/14/2004       30
 4     004       3          275        275         0      10/15/2004    12/14/2004       60
 5     005       1          250        250         0      10/15/2004    10/15/2004        0
```

Notice that the date of the first visit for each subject was 10/15/2008. If you knew this ahead of time, you could change the data step.

```
data diff;
retain firstchol;
set study;
by id;
if first.id then do;
   firstchol = chol;
   nvisits = 0;
end;
nvisits + 1;
if last.id then do;
   diffchol = chol - firstchol;
   diffdays = visit - '15oct2004'd; ①
   output;
end;
run;
```

You can use what you know about date constants and subtract the same date from the last visit date ①.

#10 The following is a more sophisticated (more elegant ?) way to do everything that was done in example #9. It is shown here not so you have to understand the SAS code, but as example of *the more you know, the less SAS code you sometimes need to perform a task*.

```
data diff;
do until (last.id);
  set study;
  by id;
  if first.id then firstchol = chol;
  nvisits = sum(nvisits,1);
end;
diffchol = chol - firstchol;
diffdays = visit - '15oct2004'd;
run;
```

#11 As in #5, one common use of FIRST. and LAST. variables is to identify duplicate observations within a data set. For example, if you are working with the vital statistics death file, each observation contains a social security number (SSN). There should only be one observation per SSN --- no repeated 'deaths' or individuals with identical SSNs --- *create one data set with observations with duplicate SSNs and another with unique SSNs*.

```
* assume the data set with social security numbers is named DEATHS;
* assume the variable with the social security number is named SSN;

proc sort data=deaths;
by ssn;
run;

data duplicates unique;
set deaths;
by ssn;
if not (first.ssn and last.ssn) then output duplicates;
else output unique;
run;
```
