Chapter 19

Creating a Single Observation from Multiple Records

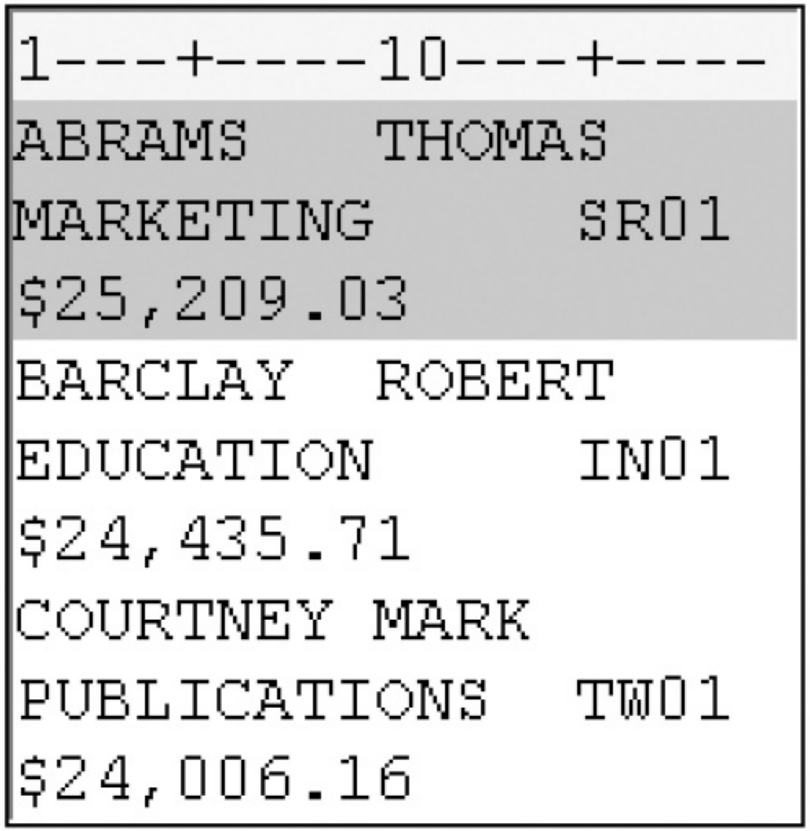
Overview

Information for one observation can be spread out over several raw data file records. For example, to read multiple records sequentially you can write multiple INPUT statements when multiple input records comprise a single observation, or alternatively you can use a line pointer control to achieve the same. You can also read multiple records non-sequentially.

Chapter topics:

- read multiple records sequentially and create a single observation
- read multiple records non-sequentially and create a single observation.

Using Multiple INPUT Statements




```
input Lname $ 1-8 Fname $ 10-15;
input Department $ 1-12 JobCode $ 15-19;
input Salary comma10.;
```

Using Line Pointer Controls to Read Multiple Records

There are two types of line pointer controls.

- The forward slash (/) specifies a line location that is relative to the current one. It advances the input pointer to the next record and must be specified *after* the instructions for reading the values in the current record.

- The *#n* specifies the absolute number of the line to which you want to move the pointer; therefore, it must be specified *before* the instructions for reading values in a specific record. It can read records in any order.

Using the Forward Slash (/) Line Pointer Control to Read Multiple Records Sequentially

The single INPUT statement below reads the values for Lname and Fname in the first record, followed by the values for Department and JobCode in the second record. After that, the value for Salary is read in the third record.

```
Input  Lname $ 1-8 Fname $ 10-15 /
       Department $ 1-12 JobCode $ 15-19 /
       Salary comma10.;
```

```
1---+----10---+----
ABRAMS    THOMAS
MARKETING      SR01
$25,209.03
BARCLAY   ROBERT
EDUCATION      IN01
$24,435.71
COURTNEY MARK
PUBLICATIONS   TW01
$24,006.16
```

Note: in this example we used column input and formatted input since it is fixed-field data.

Using the Forward Slash (/) Line Pointer Control to Read Multiple Records Sequentially

This is an example of reading non-fixed field data with the / line pointer control and different input styles based on the feature of the dataset.

```
data perm.members;
  infile memdata;
  input Fname $ Lname $ /
        Address $ 1-20 /
        City & $10. State $ Zip $;
run;
```

A
B
C

```
1---+----10---+----20
LEE ATHNOS
1215 RAINTREE CIRCLE
PHOENIX  AZ 85044
HEIDIE BAKER
1751 DIEHL ROAD
VIENNA  VA 22124
MYRON BARKER
131 DONERAIL DRIVE
ATLANTA  GA 30363
JOYCE BENEFIT
85 MAPLE AVENUE
MENLO PARK  CA 94025
```

A. Standard list input
B. Column input
C. Modified list input with the ampersand (&) modifier

Sequential Processing of Multiple Records in the DATA Step

The values in the first record are read, and the / line pointer control moves the input pointer to the second record.

```
input Fname $ Lname $ /
      Address $ 1-20 /
      City & $10. State $ Zip $;
```

```
>----+----10---+----20
LEE ATHNOS
1215 RAINTREE CIRCLE
PHOENIX  AZ 85044
HEIDIE BAKER
1751 DIEHL ROAD
VIENNA  VA 22124
```

Program Data Vector

_N_	Fname	Lname	Address	City	State	Zip
1	LEE	ATHNOS				

Sequential Processing of Multiple Records in the DATA Step

The value for Address is read, and the second / line pointer control advances the input pointer to the third record.

```
input Fname $ Lname $ /
      Address $ 1-20 /
      City & $10. State $ Zip $;
```

```
>----+----10---+----20
LEE ATHNOS
1215 RAINTREE CIRCLE
PHOENIX  AZ 85044
HEIDIE BAKER
1751 DIEHL ROAD
VIENNA  VA 22124
```

Program Data Vector

_N_	Fname	Lname	Address	City	State	Zip
1	LEE	ATHNOS	1215 RAINTREE CIRCLE			

Sequential Processing of Multiple Records in the DATA Step

The values for City, State, and Zip are read. The INPUT statement is complete.

```
input Fname $ Lname $ /
      Address $ 1-20 /
      City & $10. State $ Zip $;
```

```
>----+----10---+----20
LEE ATHNOS
1215 RAINTREE CIRCLE
PHOENIX  AZ 85044
HEIDIE BAKER
1751 DIEHL ROAD
VIENNA  VA 22124
```

Program Data Vector

_N_	Fname	Lname	Address	City	State	Zip
1	LEE	ATHNOS	1215 RAINTREE CIRCLE	PHOENIX	AZ	85044

Sequential Processing of Multiple Records in the DATA Step

The values in the program data vector are written to the dataset as the first observation.

Program Data Vector

_N_	Fname	Lname	Address	City	State	Zip
1	LEE	ATHNOS	1215 RAINTREE CIRCLE	PHOENIX	AZ	85044

SAS Data Set Perm.Members

Fname	Lname	Address	City	State	Zip
LEE	ATHNOS	1215 RAINTREE CIRCLE	PHOENIX	AZ	85044

Sequential Processing of Multiple Records in the DATA Step

Control returns to the top of the DATA step. The variable values are reinitialized to missing.

```
data perm.members;
  infile memdata;
  input Fname $ Lname $ /
        Address $ 1-20 /
        City & $10. State $ Zip $;
run;
```

```
>----+----10---+----20
LEE ATHNOS
1215 RAINTREE CIRCLE
PHOENIX  AZ 85044
HEIDIE BAKER
1751 DIEHL ROAD
VIENNA  VA 22124
```

Program Data Vector

_N_	Fname	Lname	Address	City	State	Zip
2						

Sequential Processing of Multiple Records in the DATA Step

During the second iteration, values for Fname and Lname are read beginning in column 1 of the fourth record.

```
data perm.members;
  infile memdata;
  input Fname $ Lname $ /
        Address $ 1-20 /
        City & $10. State $ Zip $;
run;
```

```
>----+----10---+----20
LEE ATHNOS
1215 RAINTREE CIRCLE
PHOENIX  AZ 85044
HEIDIE BAKER
1751 DIEHL ROAD
VIENNA  VA 22124
```

Program Data Vector

_N_	Fname	Lname	Address	City	State	Zip
2	HEIDIE	BAKER				

Sequential Processing of Multiple Records in the DATA Step

The value for Address is read. The / line pointer control advances the input pointer to the beginning of the fifth record.

```
data perm.members;
  infile memdata;
  input Fname $ Lname $ /
        Address $ 1-20 /
        City & $10. State $ Zip $;
run;
```

```
>----+----10---+----20
LEE ATHNOS
1215 RAINTREE CIRCLE
PHOENIX  AZ 85044
HEIDIE BAKER
1751 DIEHL ROAD
VIENNA  VA 22124
```

Program Data Vector

_N_	Fname	Lname	Address	City	State	Zip
2	HEIDIE	BAKER	1751 DIEHL ROAD			

Sequential Processing of Multiple Records in the DATA Step

The values for City, State, and Zip are read. The INPUT statement is complete again.

```
data perm.members;
  infile memdata;
  input Fname $ Lname $ /
        Address $ 1-20 /
        City & $10. State $ Zip $;
run;
```

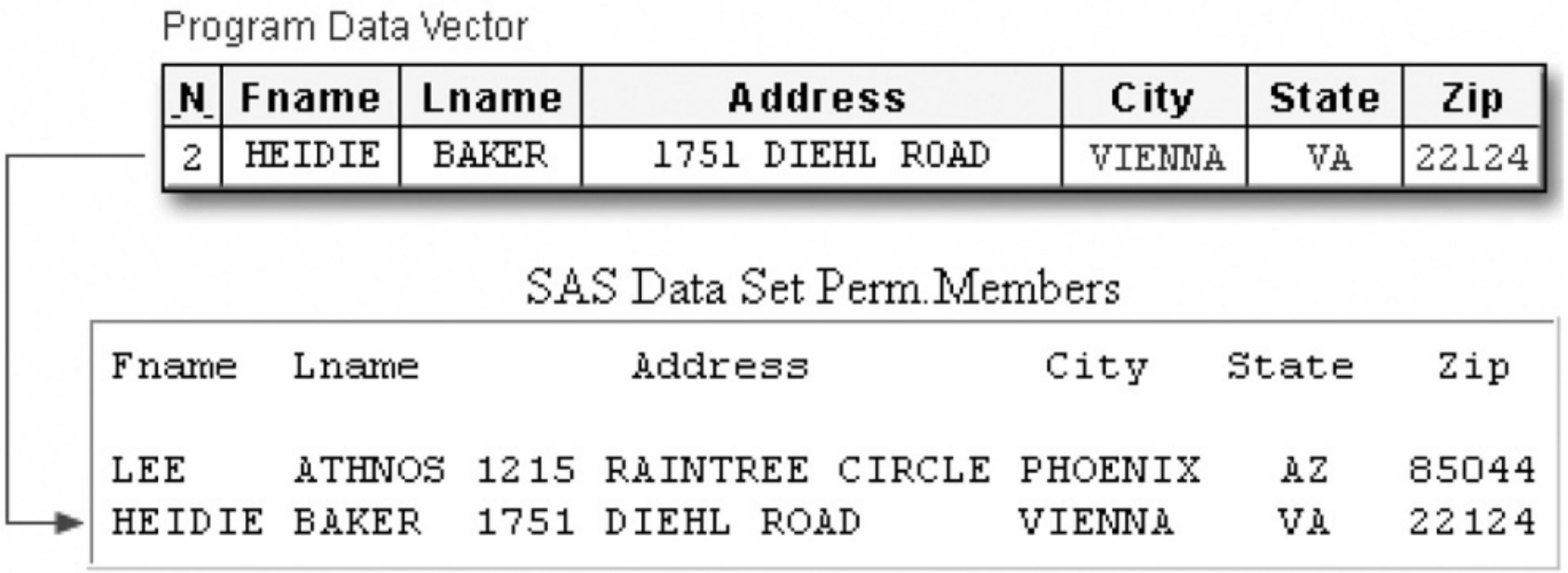
```
>----+----10---+----20
LEE ATHNOS
1215 RAINTREE CIRCLE
PHOENIX  AZ 85044
HEIDIE BAKER
1751 DIEHL ROAD
VIENNA  VA 22124
```

Program Data Vector

_N_	Fname	Lname	Address	City	State	Zip
2	HEIDIE	BAKER	1751 DIEHL ROAD	VIENNA	VA	22124

Sequential Processing of Multiple Records in the DATA Step

The values in the program data vector are written to the dataset as the second observation.



Sequential Processing of Multiple Records in the DATA Step

This process continues until the whole raw data file is read. PROC PRINT output for Perm.Members shows that a single observation contains the complete information for each member.

```
proc print data=perm.members;
run;
```

Obs	Fname	Lname	Address	City	State	Zip
1	LEE	ATHNOS	1215 RAINTREE CIRCLE	PHOENIX	AZ	85044
2	HEIDIE	BAKER	1751 DIEHL ROAD	VIENNA	VA	22124
3	MYRON	BARKER	131 DONERAIL DRIVE	ATLANTA	GA	30363
4	JOYCE	BENEFIT	85 MAPLE AVENUE	MENLO PARK	CA	94025

Reading Multiple Records Sequentially: Number of Records per Observation

In the previous example the raw data file contained **the same number** of records for each observation that was being created.

Suppose there are only two records for the second member, the name and address are read and assigned to corresponding variables. Then as directed by the INPUT statement, the third member's name is read as a value for City and the third member's address as State and Zip. So, inspect your raw data file carefully and make sure that it contains the same number of records for each observation.

```
1---+----10---+----20
LEE ATHNOS
1215 RAINTREE CIRCLE
PHOENIX  AZ 85044
HEIDIE BAKER
1751 DIEHL ROAD
MYRON BARKER
131 DONERAIL DRIVE
ATLANTA  GA 30363
JOYCE BENEFIT
85 MAPLE AVENUE
MENLO PARK  CA 94025
```

Obs	Fname	Lname	Address	City	State	Zip
1	LEE	ATHNOS	1215 RAINTREE CIRCLE	PHOENIX	AZ	85044
2	HEIDIE	BAKER	1751 DIEHL ROAD	MYRON BARK	131	DONERAIL
3	ATLANTA	GA	JOYCE BENEFIT	85 MAPLE A	MENLO	PARK

Reading Multiple Records Non-Sequentially: Using the #n Line Pointer Control

The #n pointer control can read records in any order.

```
input
    #1 Lname $ 1-8 Fname $ 10-15
    #2 Department $ 1-12 JobCode $ 15-19
    #3 Salary comma10.;

input
    #2 Department $ 1-12 JobCode $ 15-19
    #1 Lname $ 1-8 Fname $ 10-15
    #3 Salary comma10.;

input
    #2 Name $ 1-12 Age 15-16
    Gender $ 18;
```

```
1---+----10---+----
ABRAMS   THOMAS
MARKETING      SR01
$25,209.03
BARCLAY  ROBERT
EDUCATION      IN01
$24,435.71
COURTNEY MARK
PUBLICATIONS   TW01
$24,006.16
```

Raw Data File Admit

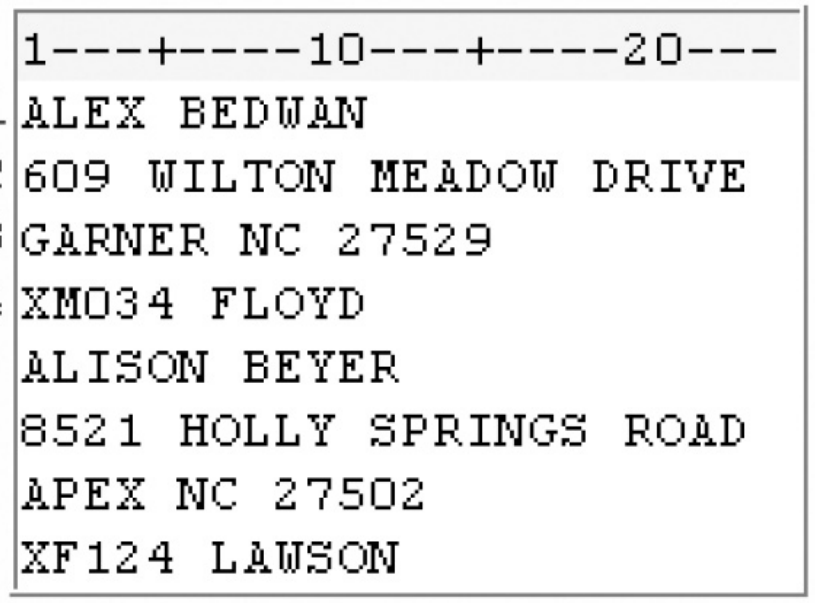
```
1---+----10---+----
S. Thompson    37 M
L. Rochester   31 F
M. Sabatello   43 M
```

Reading Multiple Records Non-Sequentially: Using the #n Line Pointer Control

In the raw data file Patdata, the first three records contain a patient's name, address, city, state, and zip code, and the fourth record contains the patient's ID number followed by the name of the primary physician.

You want to read each patient's information in the following order:

1. ID number (ID)
2. first name (Fname)
3. last name (Lname)
4. address (Address)
5. city (City)
6. state (State)
7. zip (Zip)
8. doctor (Doctor)



Reading Multiple Records Non-Sequentially: Using the #n Line Pointer Control

To read the value for ID in the fourth record, specify #4 before naming the variable and defining its attributes.

```
data perm.patients;
    infile patdata;
    input
        #4 ID $5.
```

```
1---+----10---+----20---
ALEX BEDWAN
609 WILTON MEADOW DRIVE
GARNER NC 27529
XM034 FLOYD
```

Reading Multiple Records Non-Sequentially: Using the #n Line Pointer Control

To read the values for Fname and Lname in the first record, specify #1 before naming the variables and defining their attributes.

```
data perm.patients;
    infile patdata;
    input
        #4 ID $5.
        #1 Fname $ Lname $
```

```
1---+----10---+----20---
ALEX BEDWAN
609 WILTON MEADOW DRIVE
GARNER NC 27529
XM034 FLOYD
```

Reading Multiple Records Non-Sequentially: Using the #n Line Pointer Control

Use the #n line pointer control to move the input pointer to the second record and read the value for Address.

```
data perm.patients;
    infile patdata;
    input
        #4 ID $5.
        #1 Fname $ Lname $
        #2 Address $23.
```

```
1---+----10---+----20---
ALEX BEDWAN
609 WILTON MEADOW DRIVE
GARNER NC 27529
XM034 FLOYD
```

Reading Multiple Records Non-Sequentially: Using the #n Line Pointer Control

Now move the input pointer to the third record and read the values for City, State, and Zip, in that order. In this raw data file, the values for City contain eight characters or less and do not contain embedded blanks, so you can use standard list input to read these values.

```
data perm.patients;
    infile patdata;
    input
        #4 ID $5.
        #1 Fname $ Lname $
        #2 Address $23.
        #3 City $ State $ Zip $
```

```
1---+----10---+----20---
ALEX BEDWAN
609 WILTON MEADOW DRIVE
GARNER NC 27529
XM034 FLOYD
```

Reading Multiple Records Non-Sequentially: Using the #n Line Pointer Control

Now you need to move the input pointer down to the fourth record to read the value for Doctor, which begins in column 7.

```
data perm.patients;
    infile patdata;
    input
        #4 ID $5.
        #1 Fname $ Lname $
        #2 Address $23.
        #3 City $ State $ Zip $
        #4 @7 Doctor $6.;
run;
```

```
1---+----10---+----20---
ALEX BEDWAN
609 WILTON MEADOW DRIVE
GARNER NC 27529
XM034 FLOYD
```

Formatted input

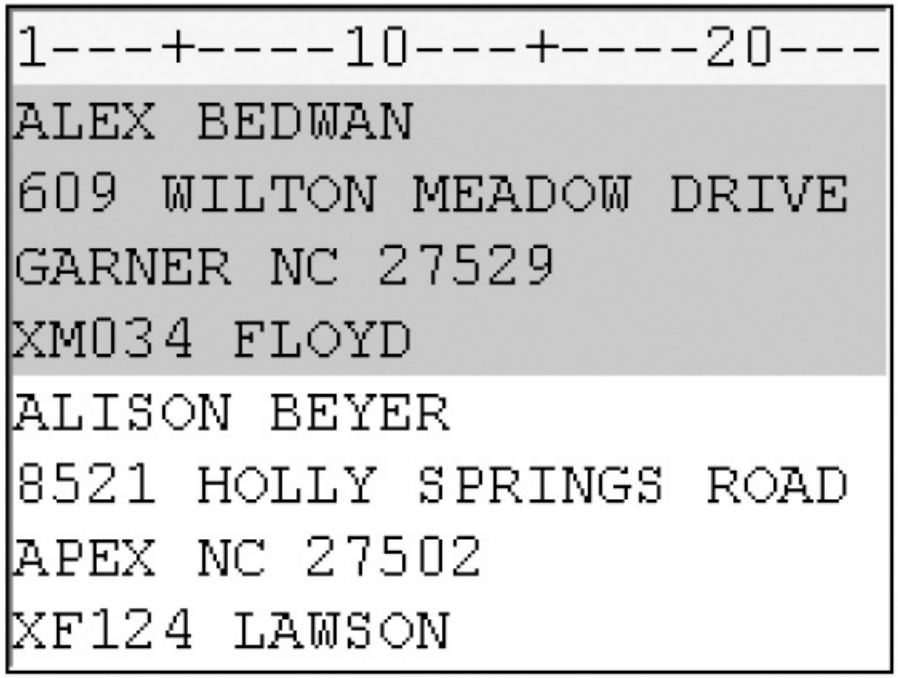
Column pointer control

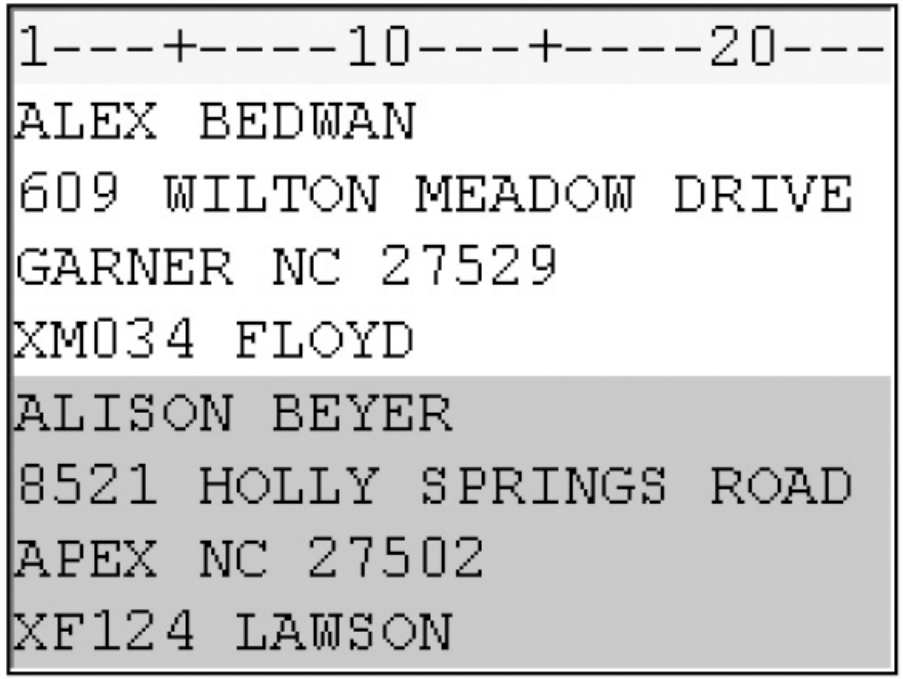
Line pointer control

Reading Multiple Records Non-Sequentially: Using the #n Line Pointer Control

The first time the DATA step executes, the first four records are read, and an observation is written to the dataset.

During the second iteration, the next four records are read, and the second observation is written to the dataset, and so on.





Reading Multiple Records Non-Sequentially: Using the #n Line Pointer Control

The PROC PRINT output of the dataset shows how the data fields that are spread over several records have been condensed into one observation.

ID	Fname	Lname	Address	City	State	Zip	Doctor
XM034	ALEX	BEDWAN	609 WILTON MEADOW DRIVE	GARNER	NC	27529	FLOYD
XF124	ALISON	BEYER	8521 HOLLY SPRINGS ROAD	APEX	NC	27502	LAWSON
XF232	LISA	BONNER	109 BRAMPTON AVENUE	CARY	NC	27511	LAWSON
XM065	GEORGE	CHESSON	3801 WOODSIDE COURT	GARNER	NC	27529	FLOYD

Combining Line Pointer Controls

The forward slash (/) line pointer control and the *#n* line pointer control can be used together in a SAS program to read multiple records both sequentially and non-sequentially.

Combine two controls:

```
data perm.patients;
  infile patdata;
  input
       #4 ID $5.
       #1 Fname $ Lname $ /
          Address $23. /
          City $ State $ Zip $ /
          @7 Doctor $6.;
run;
```

Only use #n control (alternative):

```
data perm.patients;
    infile patdata;
    input
       #4 ID $5.
       #1 Fname $ Lname $
       #2 Address $23.
       #3 City $ State $ Zip $
       #4 @7 Doctor $6.;
run;
```