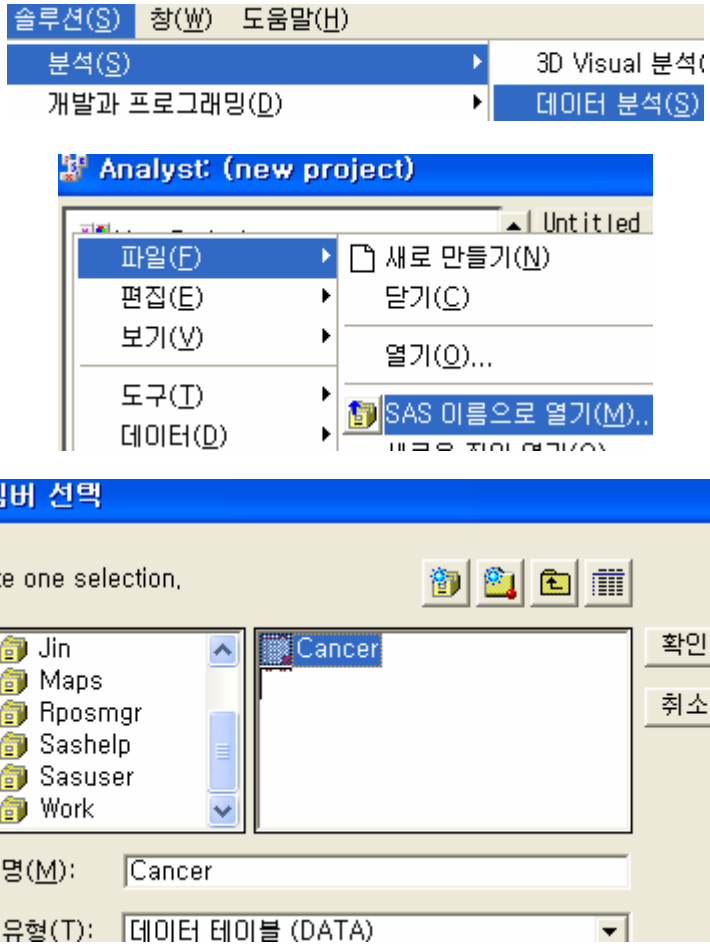
CHAPTER 3.

데이터 변환하기

2장에서는 SAS 데이터 만드는 방법(확장 편집기 이용, 메뉴 이용 외부 데이터 불러오기, 확장 편집기에서 INFILE 문이나 DATAFILE 옵션 사용하기)에 대해 살펴 보았다. 3장에서는 이미 만들어진 SAS 데이터를 연산자, 함수, 데이터 문 옵션, 제어 명령어 이용하여 변환하는 방법을 살펴보기로 하자. 통계학에서 주로 사용되는 함수에 대해서는 4장에서 다루기로 한다.

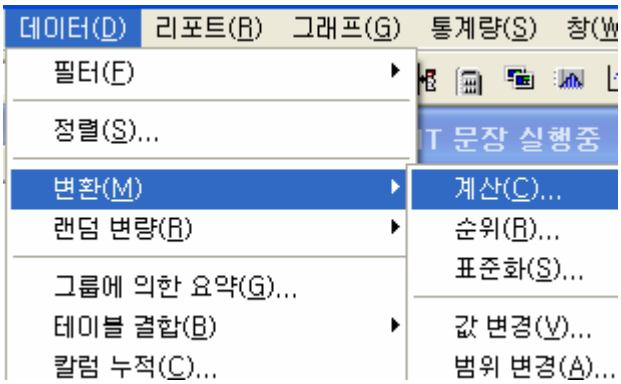
3.1 Solution(솔루션) 이용하기

데이터 변환도 SAS/ASSIST 이용하여 메뉴에서 가능하나 절차가 길고 복잡하다. 예를 들어 설명하기로 하자. 기존의 CANCER 데이터에서 시간 변수를 LOG 취하는 과정을 보자. 우선 변환하기 원하는 SAS 데이터를 아래 절차에 따라 설정한다.

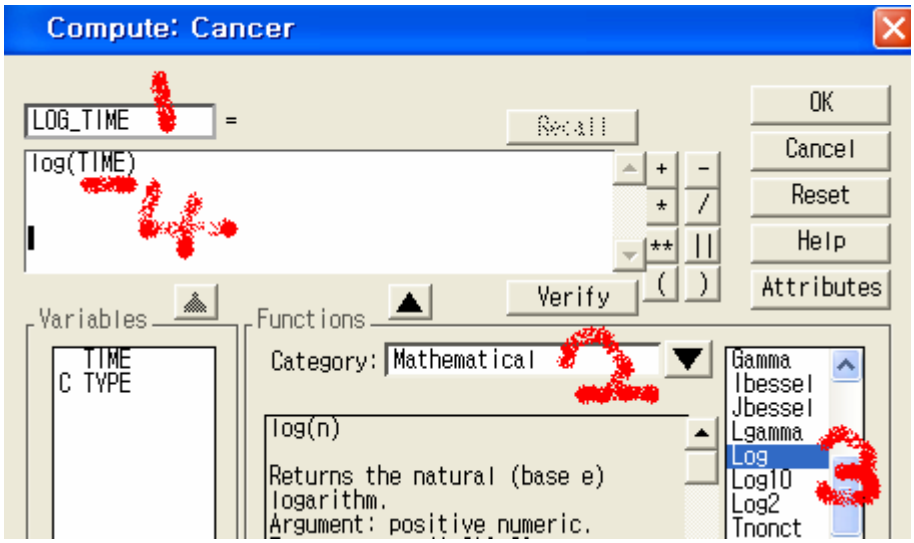


오른쪽 화면에 열린 SAS 데이터는 스프레드시트 형식으로 나타난다. 처음 열린 데이터는 보기 모드로 되어 있어 데이터 편집이 가능한 상태가 아니다. 그러므로 우선 편집이 가능한 모드로 바꾸어야 한다. 방법은 다음과 같다.

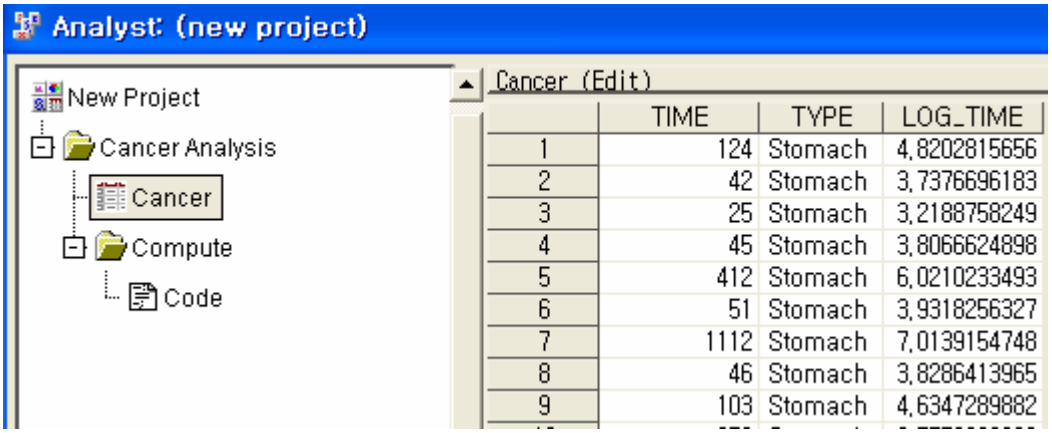
SAS 데이터가 편집 모드로 바뀌어야 변환 메뉴 안의 계산(C), 순위(R) 등의 기능이 사용가능(enabled)하게 된다. 보기 모드에서는 Disabled 되어 있어 사용이 불가능하다.



아래 화면에서 1. 새로운 변수 이름 지정, 만약 TIME을 사용하면 계산된 결과가 원 TIME 변수에 저장된다. 가능하면 다른 이름 사용하는 것이 편리하다. 2. 원하는 작업이 있는 category를 선택한다. 3. 적절한 함수를 선택한다. 4. 변환하려는 변수를 지정한다.



설정이 끝나면 OK 버튼을 눌러 작업을 완료한다. CANCER 데이터 마지막 열에 새로운 변수 LOG_TIME이 계산되어 있다. 오른쪽 화면 위에 데이터 이름 옆에 Cancer (Edit) 편집 모드를 알리는 "EDIT"가 나타나 있다. 보기 모드인 경우에는 "VIEW"가 표시된다.



	TIME	TYPE	LOG_TIME
1	124	Stomach	4.8202815656
2	42	Stomach	3.7376696183
3	25	Stomach	3.2188758249
4	45	Stomach	3.8066624898
5	412	Stomach	6.0210233493
6	51	Stomach	3.9318256327
7	1112	Stomach	7.0139154748
8	46	Stomach	3.8286413965
9	103	Stomach	4.6347289882

확장 편집기에서 프로그램을 이용하는 경우 다음 프로그램이 위의 작업과 동일한 효과를 갖는다. CANCER 데이터의 변수TIME의 로그 값이 계산되어 LOG_TIME 변수가 만들어지고 그 데이터가 "CANCER01"에 저장된다.

```
cancer.sas *
DATA CANCER;

DATA CANCER1;
     SET CANCER;
     LOG_TIME=LOG(TIME);
RUN;
```

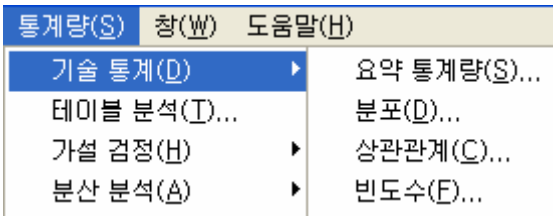
```
cancer.sas *
DATA CANCER;
DATA CANCER1;

TITLE 'CANCER1 DATA';
PROC PRINT DATA=CANCER1;
RUN;
```

▶

CANCER1 DATA

Obs	TIME	TYPE	LOG_TIME
1	124	Stomach	4.82028
2	146	Stomach	4.98361
3	64	Bronchus	4.15888
4	1112	Stomach	7.01392
5	450	Bronchus	6.10925
6	42	Stomach	3.73767

아래에 보는 것처럼 솔루션 기능에서 여러 분석이 가능하지만 데이터 변환과 같이 확장 편집기에서 적절한 PPROCEDURE(5장 참고) 단계만 작성하면 간단하게 원하는 분석 결과를 얻을 수 있다.

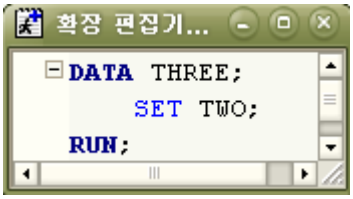


3.2 SET 문과 연산자

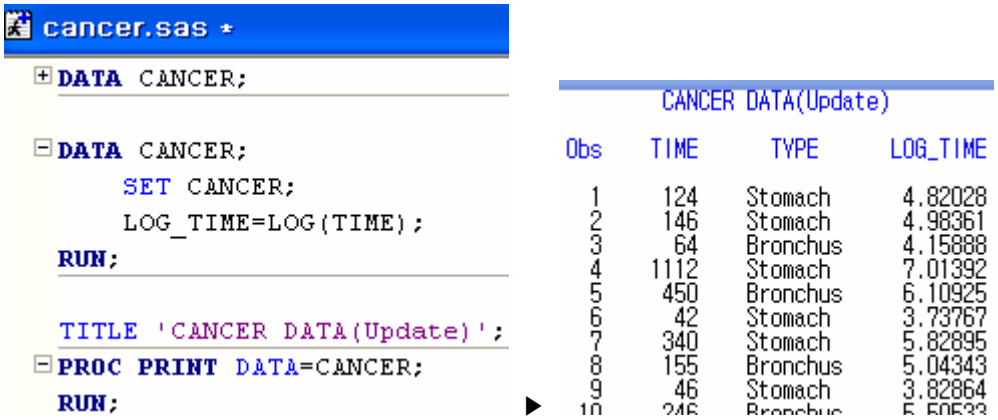
3.2.1 SET 문

SET 문은 기존의 SAS 데이터를 이용하여 새로운 SAS 데이터를 만드는 데이터 단계에서 기존의 SAS 데이터 이름을 설정하는 곳이다. 다음 프로그램은 앞의 예제 데이터 TWO와 동일한 SAS 데이터 THREE를 만드는 것이다. 이를 이용하여 이미 만들어진 SAS 데이터를 조작하게 된다. 일반적으로 SET 문과 RUN 문장 사이에 데이터 작업 프로그램을 실

행하게 된다.



다음 DATA 옆의 새로운 데이터 이름과 SET의 기존의 데이터 이름이 동일하면 데이터 변환 작업 결과 데이터가 기존의 데이터에 겹쳐 저장된다. 기존의 데이터에서 원 변수 값이 변환되지 않고 새로운 변수가 만들어지는 경우에는 기존 데이터 이름을 그대로 사용하는 것이 효율적이다.



3.2.2 연산자

사칙연산 및 Power(승)은 +, -, *, /, **을 사용하면 된다. 로그(log), 지수(exponential) 값은 SAS 함수(4장 참고)를 이용해야 한다. 다음 프로그램은 데이터 TWO를 이용하여 변수 IQ2를 IQ를 이용하여 만들고 IQ 변수를 제곱하여 IQ에 다시 저장하여 데이터 THREE를 만든다. 만약 THREE 대신 TWO를 쓰면 이전의 TWO 데이터에 새로 만들어진 데이터가 저장된다. 연산자의 우선 순위는 수학의 연산자 순위와 같다. 우선 순위를 결정하는 괄호 기호는 ()만을 사용할 수 있다.

```
cancer.sas *
DATA CANCER;

DATA CANCER2;
     SET CANCER;
     TIME1=LOG(TIME**2);
     TIME=2*TIME;
RUN;

TITLE 'CANCER2';
PROC PRINT DATA=CANCER2;
RUN;
```

CANCER2

Obs	TIME	TYPE	TIME1
1	248	Stomach	9.6406
2	292	Stomach	9.9672
3	128	Bronchus	8.3178
4	2224	Stomach	14.0278
5	900	Bronchus	12.2185
6	84	Stomach	7.4753
7	680	Stomach	11.6579
8	310	Bronchus	10.0869
9	92	Stomach	7.6573
10	492	Bronchus	11.0107
11	50	Stomach	6.4328

비교 연산자는 다음과 같다. 기호를 사용하거나 문자를 사용할 수 있다. 문자보다는 기호를 사용하는 것이 편리하다.

기호	=	^=	>	<	<=	>=	^>	^<
문자	EQ	NE	GT	LT	LE	GE	NG	NL

논리 연산자는 AND(그리고), OR(혹은)을 사용할 수 있다. 다음 프로그램은 시간에 100 이상이고(and) 병 종류가 “Stomach”인 사람만 새로운 데이터로 만드는 프로그램이다. 프로그램은 대소문자 구별이 없지만 문자 관측치의 경우에는 대소문자 구별을 해야 한다. 그러므로 아래 프로그램에서 “Stomach”로 하지 않으면 CANCER2에는 관측치가 하나도 저장되지 않는다.

```
cancer.sas *
DATA CANCER2;
     SET CANCER;
     IF (TIME>100) AND (TYPE='Stomach');
RUN;

PROC PRINT DATA=CANCER2;
RUN;
```

Obs	TIME	TYPE
1	124	Stomach
2	146	Stomach
3	1112	Stomach
4	340	Stomach
5	396	Stomach
6	103	Stomach
7	876	Stomach
8	412	Stomach

3.3 제어문

SAS의 프로그램은 한 줄씩 아래로 실행한다. 그리고 데이터 단계에서는 변환 작업이 데이터 행 단위로 차례로 실행된다. 위의 프로그램을 분해해 보자. 데이터 CANCER에는 두 개의 변수(TIME, TYPE)와 30개의 관측치가 있다. 프로그램이 실행되면 아래 절차에 의해 데이터 단계가 완성된다.

DATA CANCER2;

CANCER2 이름의 SAS 데이터를 만들 준비한다.

SET CANCER;

기존의 CANCER 데이터를 불러와 작업 준비를 한다.

TIME1=LOG(TIME2);**

TIME1 이름의 변수가 기존 데이터에 없으므로 우선 생성한다. 그리고 $LOG(Time^2)$을 계산하여 (time 값은 첫 관측치인 124가 이용) 변수 TIME1 첫 관측치에 지정한다.

TIME=2*TIME;

TIME 이름의 변수가 기존 데이터에 있으므로 2*TIME(time 값은 124) 계산되어 TIME 변수 ct 관측치가 248로 바뀐다.

RUN;

이 문장은 더 이상의 데이터 작업이 없음을 SAS에게 알리게 된다. 위의 작업이 첫 행만 끝났으므로 두 번째 행에서 동일한 작업을 반복하여 TIME1애는 9.96, TIME에는 292를 저장하고 세 번째 행으로 이동한다. 데이터 마지막 관측치까지 동일한 작업이 반복하고 결과는 데이터 CANCER2에 저장된다. PROC PRINT에 의해 출력 창에 출력하면 변환 결과를 얻게 된다.

SAS의 순차적 실행 과정을 제어하는 문장을 제어문(control statement)이라 하는데 가장 많이 사용하는 것이 IF문과 DO문이 있다.

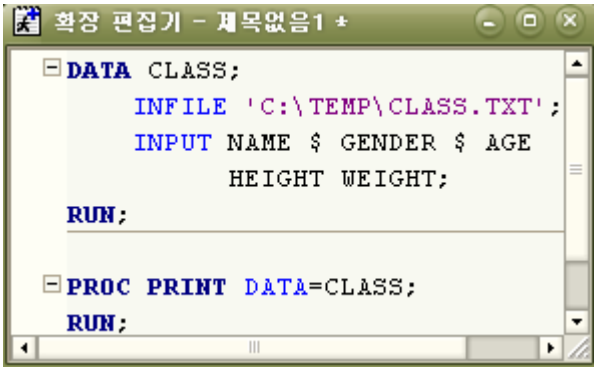
3.3.1 IF문(1)

IF (조건) THEN 문장;

조건이 성립하면 문장이 실행된다. 조건의 ( )은 사용하지 않아도 되나 프로그램을 이해하기 쉽게 하려면 사용하는 것이 좋다.

EXAMPLE: if ~ then 문 사용하기

CLASS.txt는 학생 19명의 이름(Name), 성별(Gender), 나이(Age), 키(Height, 인치), 몸무게(Weight, 파운드)를 조사한 자료이다. 이 예제 데이터는 SASHELP 라이브러리에 CLASS 라는 이름의 SAS 데이터로 존재하는 SAS 예제 데이터이다. (관측치 수 19개이다)



Obs	NAME	GENDER	AGE	HEIGHT	WEIGHT
1	Alfred	M	14	69.0	112.5
2	Alice	F	13	56.5	84.0
3	Barbara	F	13	65.3	98.0
4	Carol	F	14	62.8	102.5
5	Henry	M	14	63.5	102.5
6	James	M	12	57.3	83.0
7	Jane	F	12	59.8	84.5
8	Janet	F	15	62.5	112.5
9	Jeffrey	M	13	62.5	84.0
10	John	M	12	59.0	99.5

위에서 키를 CM(1인치=2.54cm), 몸무게를 Kg(1파운드=453g)으로 환산하고, 변수명은 HEIGHT0와 WEIGHT0로 하자. 또한 공식 (체중kg)/(신장m)^{2}을 이용하여 신체질량지수(Mass Index)를 계산하고 변수명은 MASS로 하자.

확장 편집기 - 제목없음1 *

```
DATA CLASS1;
    SET CLASS;
    HEIGHTO=HEIGHT*2.54;
    WEIGHTO=WEIGHT*0.453;
    MASS=WEIGHTO/(HEIGHTO/100)**2;
RUN;

PROC PRINT DATA=CLASS1;
    VAR HEIGHT HEIGHTO WEIGHT WEIGHTO MASS;
RUN;
```

PROC PRINT 에서 VAR 문장은 출력하기 원하는 변수만 지정해주는 것이다. 출력된 결과는 다음과 같다.

Obs	HEIGHT	HEIGHTO	WEIGHT	WEIGHTO	MASS
1	69.0	175.260	112.5	50.9625	16.5915
2	56.5	143.510	84.0	38.0520	18.4762
3	65.3	165.862	98.0	44.3940	16.1373
4	62.8	159.512	102.5	46.4325	18.2488
5	63.5	161.290	102.5	46.4325	17.8487
6	57.3	145.542	83.0	37.5990	17.7501

키가 170cm 이상인 개체(학생)에 대해 “TALL” 이라고 구별하는 변수(GROUP)를 만들자.

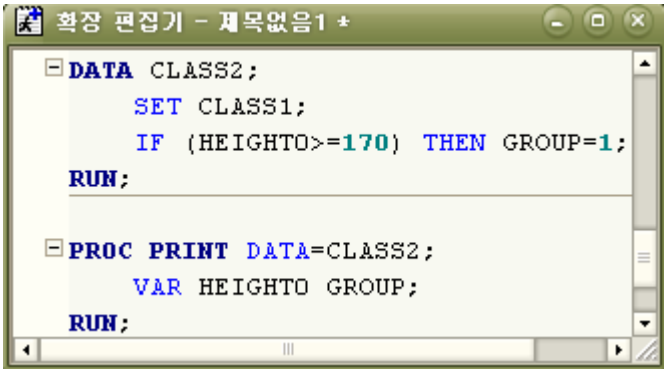
확장 편집기 - 제목없음1 *

```
DATA CLASS2;
    SET CLASS1;
    IF (HEIGHTO>=170) THEN GROUP='TALL';
RUN;

PROC PRINT DATA=CLASS2;
    VAR HEIGHTO GROUP;
RUN;
```

SAS 시스템

Obs	HEIGHTO	GROUP
1	175.260	TALL
2	143.510	
3	165.862	
4	159.512	
5	161.290	
6	145.542	
7	151.892	
8	158.750	
9	158.750	
10	149.860	
11	130.302	
12	163.322	

170cm 이상인 경우에는 GROUP 변수가 TALL이 되나 그렇지 않으면 결측치가 된다. 문자일 경우에는 빈 칸이 되나 숫자인 경우에는 .이 된다.

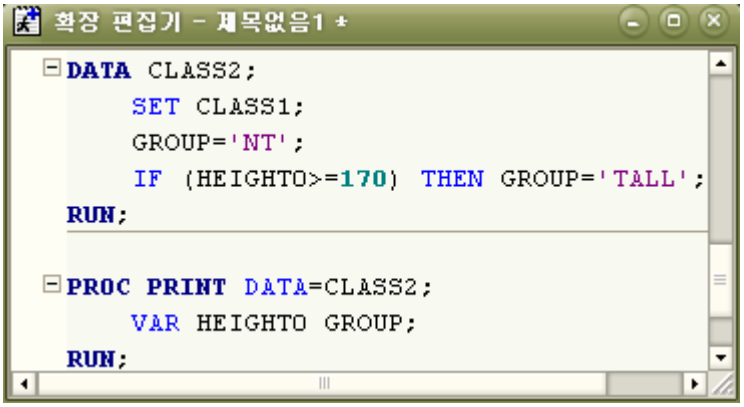


```
확장 편집기 - 제목없음1 *
DATA CLASS2;
    SET CLASS1;
    IF (HEIGHTO>=170) THEN GROUP=1;
RUN;

PROC PRINT DATA=CLASS2;
    VAR HEIGHTO GROUP;
RUN;
```

SAS 시스템

Obs	HEIGHTO	GROUP
1	175.260	1
2	143.510	.
3	165.862	.
4	159.512	.
5	161.290	.
6	145.542	.
7	151.892	.
8	158.750	.
9	158.750	.
10	149.860	.
11	130.302	.
12	163.322	.

그러므로 GROUP에 대한 기본값을 설정해 주는 것이 좋다. 다음과 같이 문자열의 크기는 처음 설정하는 곳의 크기에 따라 결정된다. 그러므로 'NT'로 하면 GROUP 변수의 문자열 크기는 2이다.



```
확장 편집기 - 제목없음1 *
DATA CLASS2;
    SET CLASS1;
    GROUP='NT';
    IF (HEIGHTO>=170) THEN GROUP='TALL';
RUN;

PROC PRINT DATA=CLASS2;
    VAR HEIGHTO GROUP;
RUN;
```

SAS 시스템

Obs	HEIGHTO	GROUP
1	175.260	TA
2	143.510	NT
3	165.862	NT
4	159.512	NT
5	161.290	NT
6	145.542	NT
7	151.892	NT
8	158.750	NT
9	158.750	NT
10	149.860	NT
11	130.302	NT
12	163.322	NT
13	143.002	NT

조건을 만족하는 데이터 일부만 얻을 때 DELETE 명령을 사용한다. 아래 왼쪽은 성별이 M(case sensitive, 대소문자 구별)인 데이터를 없애는 것이고 오른쪽은 F인 데이터만 얻는 것이므로 같다. 이를 데이터 SUBSET이라 한다.

```
DATA CLASS3;
     SET CLASS2;
     IF (GENDER='M') THEN DELETE;
RUN;

PROC PRINT DATA=CLASS3;
     VAR GENDER;
RUN;
```

```
DATA CLASS3;
     SET CLASS2;
     IF (GENDER='F');
RUN;

PROC PRINT DATA=CLASS3;
     VAR GENDER;
RUN;
```

SAS 시스템

Obs	GENDER
1	F
2	F
3	F
4	F
5	F
6	F
7	F
8	F
9	F

EXAMPLE: if ~ then 문 사용하기(2)

몸무게가 30미만이면 'WEAK', 몸무게가 50이상이면 'BIG', 그리고 30이상 50미만이면 'NORMAL'로 분류하는 변수 GROUP1을 만들어보자.

```
DATA CLASS4;
     SET CLASS2;
     IF (WEIGHTO<30) THEN GROUP1='WEAK';
     IF (WEIGHTO>=50) THEN GROUP1='BIG';
     IF (30<=WEIGHTO<50) THEN GROUP1='NORMAL';
RUN;

PROC PRINT DATA=CLASS4;
     VAR WEIGHTO GROUP1;
RUN;
```

Obs	WEIGHT0	GROUP1
1	50.9625	BIG
2	38.0520	NORM
3	44.3940	NORM
4	46.4325	NORM

EXAMPLE: if ~ then 문 사용하기(3)

다음은 신체질량지수에 따른 비만 판정 기준이다. MASS0=(MASS+10)의 값을 이용하여 각 학생의 비만 정도를 변수 MASS_G으로 표현해보자. 적정='NORMAL' 조금 비대='LITTLE' 비대='FAT'으로 설정하시오.

신체질량지수에 의한 비만 판정기준표(단위: Kg/m^2)ACSM 1998		
구분	남자	여자
적정	24 ~ 27	23 ~ 26
조금 비대	27 ~ 31	26 ~ 32
비대	31 이상	32 이상

확장 편집기 - 제목없음1 *

```
DATA CLASS5;
    SET CLASS1;
    MASS0=MASS+10;
    IF (GENDER='M' AND 24<=MASS0<=27) THEN MASS_G='NORMAL';
    IF (GENDER='M' AND 27<MASS0<=31) THEN MASS_G='LITTLE';
    IF (GENDER='M' AND MASS0>31) THEN MASS_G='FAT';
    IF (GENDER='F' AND 23<=MASS0<=26) THEN MASS_G='NORMAL';
    IF (GENDER='F' AND 26<MASS0<=32) THEN MASS_G='LITTLE';
    IF (GENDER='F' AND MASS0>32) THEN MASS_G='FAT';
RUN;

PROC PRINT DATA=CLASS5;
    VAR GENDER MASS0 MASS_G;
RUN;
```

Obs	GENDER	MASSO	MASS_G
1	M	26.5915	NORMAL
2	F	28.4762	LITTLE
3	F	26.1373	LITTLE

EXAMPLE: if 문으로 데이터 일부 얻기

CLASS.txt에서 키가 60인치 이상이거나 몸무게 100파운드 이상인 학생들만으로 이루어진 SAS 데이터를 만들어보자. IF문의 () 조건 후에 아무 것도 사용하지 않으면 조건을 만족하는 관측치만 SET 문의 데이터에 저장된다.

확장 편집기 - 제목없음1 *

```
DATA CLASS7;
     SET CLASS;
     IF (HEIGHT>=60) OR (WEIGHT>=100);
RUN;

PROC PRINT DATA=CLASS7;
     VAR NAME HEIGHT WEIGHT;
RUN;
```

Name	Height	Weight
Alfred	69.0	112.5
Barbara	65.3	98.0
Carol	62.8	102.5
Henry	63.5	102.5
Janet	62.5	112.5
Jeffrey	62.5	84.0
Judy	64.3	90.0
Mary	66.5	112.0
Philip	72.0	150.0
Robert	64.8	128.0
Ronald	67.0	133.0
William	66.5	112.0

아래 프로그램은 이전 프로그램과 동일한 결과를 얻는다.

확장 편집기 - 제목없음1 *

```
DATA CLASS7;
     SET CLASS;
     IF (HEIGHT<60) AND (WEIGHT<100);
RUN;
```

3.3.2 IF문(2)

IF (조건) THEN DO; 문장(들); END;

( )의 조건이 성립하면 DO~END 사이의 문장들을 실행한다. 3.2.1절에서 THEN 뒤에 문장이 하나인 경우와 달리 여러 문장을 실행하고자 할 때 사용된다.

EXAMPLE: if ~ then do; 사용하기

학생들의 키(HEIGHT)가 60인치 보다 큰 관측치(학생)의 경우 몸무게(파운드, WEIGHT)는 0.9배, 키는 1.1배를 하자. 변수명은 HEIGHT2와 WEIGHT2으로 하자.

확장 편집기 - 제목없음1 *

```
DATA CLASS6;
    SET CLASS;
    IF (HEIGHT>60) THEN DO;
        HEIGHT2=HEIGHT*11;
        WEIGHT2=WEIGHT*0.9;
    END;
RUN;

PROC PRINT DATA=CLASS6;
    VAR HEIGHT HEIGHT2 WEIGHT WEIGHT2;
RUN;
```

SAS 시스템

Obs	HEIGHT	HEIGHT2	WEIGHT	WEIGHT2
1	69.0	759.0	112.5	101.25
2	56.5	.	84.0	.
3	65.3	718.3	98.0	88.20
4	62.8	690.8	102.5	92.25
5	63.5	698.5	102.5	92.25
6	57.3	.	83.0	.
7	59.8	.	84.5	.
8	62.5	687.5	112.5	101.25
9	62.5	687.5	84.0	75.60

만약 키가 60보다 크지 않은 학생의 경우 새로운 변수 HEIGHT2와 WEIGHT2는 결측치이다. 만약 이 때 새로운 변수의 값을 기존 값으로 하려면 다음과 같이 하면 된다.

확장 편집기 - 제목없음1 *

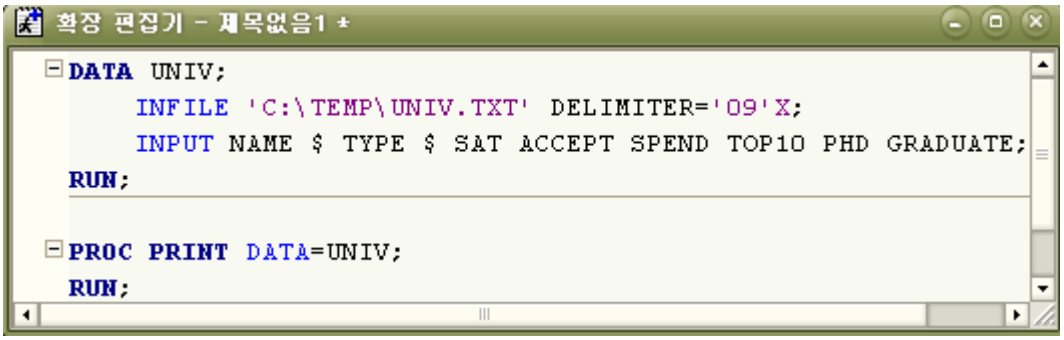
```
DATA CLASS6;
     SET CLASS;
          HEIGHT2=HEIGHT;
          WEIGHT2=WEIGHT;
     IF (HEIGHT>60)THEN DO;
          HEIGHT2=HEIGHT*11;
          WEIGHT2=WEIGHT*0.9;
     END;
RUN;
```

SAS 시스템

Obs	HEIGHT	HEIGHT2	WEIGHT	WEIGHT2
1	69.0	759.0	112.5	101.25
2	56.5	56.5	84.0	84.00
3	65.3	718.3	98.0	88.20
4	62.8	690.8	102.5	92.25
5	63.5	698.5	102.5	92.25
6	57.3	57.3	83.0	83.00
7	59.8	59.8	84.5	84.50
8	62.5	687.5	112.5	101.25
9	62.5	687.5	84.0	75.60
10	59.0	59.0	99.5	99.50
11	51.3	51.3	50.5	50.50
12	64.3	707.3	90.0	81.00

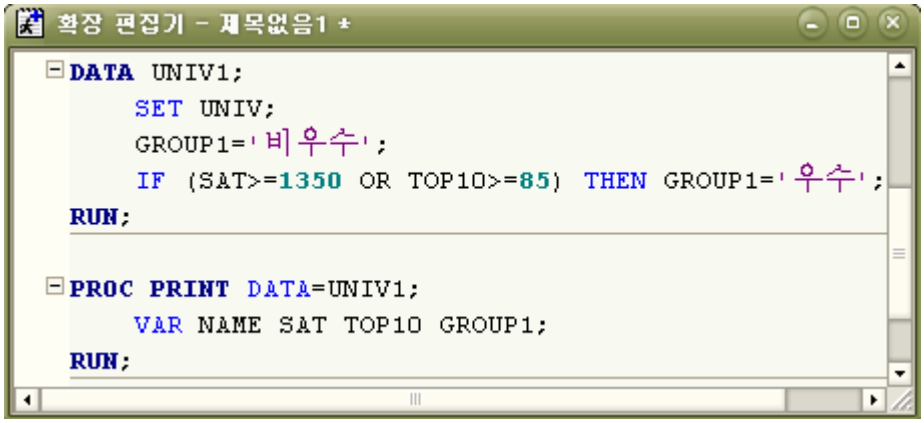
EXAMPLE: if ~ then do; 사용하기(2)

🖫UNIV.txt 자료 미국 대학 이름, 학교 종류(Liberal Arts, Univ.), SAT 점수, 입학률(acceptance rate), 일년간 학생들이 쓴 비용, 입학생 중 고등학교 때 10% 안에 든 학생 비율(%), 교직원의 박사 학위 소지 비율(%), 졸업하는 비율(%)이다.



Obs	NAME	TYPE	SAT	ACCEPT	SPEND	TOP10	PHD	GRADUATE
1	Amherst	Lib Arts	1315	22	26636	85	81	93
2	Swarthmo	Lib Arts	1310	24	27487	78	93	88
3	Williams	Lib Arts	1336	28	23772	86	90	93
4	Washingt	Lib Arts	1234	29	17998	61	89	78
5	Grinnell	Lib Arts	1244	67	22301	65	79	73
6	Mount Ho	Lib Arts	1200	61	23358	47	83	83

SAT 성적이 1350 이상이거나 입학생 중 고등학교 때 10% 안에 든 학생 비율(TOP10)이 85% 이상이면 우수 학교라 하고 그렇지 않으면 비 우수 학교라 정의한다. 우수 여부를 분류하는 변수를 하나 만들어보고, 변수명은 GROUP1 이라 하자.



Obs	NAME	SAT	TOP10	GROUP1
1	Amherst	1315	85	우수
2	Swarthmo	1310	78	비우수
3	Williams	1336	86	우수
4	Washingt	1234	61	비우수
5	Grinnell	1244	65	비우수
6	Mount Ho	1200	47	비우수
7	Colby	1200	52	비우수

변수 TOP10, PHD, GRADUATE 값의 평균을 구해보고, 변수명은 AVG1으로 하자. 만약 AVG1의 값이 85이상이면 TOP10은 90, PHD는 85로 하고, AVG1 변수가 출력될 때 정수로 출력되게 하자.

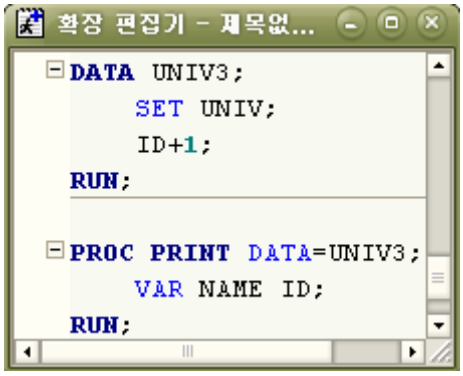
확장 편집기 - 제목없음1 *

```
DATA UNIV2;
     SET UNIV;
          AVG1=INT(MEAN(TOP10, PHD, GRADUATE));
     IF (AVG1>=85) THEN DO;
          TOP10=90;
          PHD=85;
     END;
RUN;

PROC PRINT DATA=UNIV2;
     VAR NAME TOP10 PHD GRADUATE AVG1;
RUN;
```

Obs	NAME	TOP10	PHD	GRADUATE	AVG1
1	Amherst	90	85	93	86
2	Swarthmo	90	85	88	86
3	Williams	90	85	93	89
4	Washingt	61	89	78	76
5	Grinnell	65	79	73	72
6	Mount Ho	47	83	83	71
7	Colby	52	75	84	70

자동 생성 변수 ID+1;

ID+1;의 의미는 변수 ID를 만들고 관측치마다 1씩 증가시키라는 것이다. ID+2;를 하면 어떻게 될까? 2, 4, 6, …으로 된다.



SAS 시스템

Obs	NAME	ID
1	Amherst	1
2	Swarthmo	2
3	Williams	3
4	Washingt	4
5	Grinnell	5
6	Mount Ho	6
7	Colby	7
8	Hamilton	8
9	Bates	9
10	Haverfor	10
11	Harvard	11

3.3.3 IF문 (3)IF (조건) THEN; 문장1(들); ELSE DO; 문장2(들); END;

( )의 조건이 성립하면 문장1을 그렇지 않은 경우 문장2를 실행한다. 그러나 이 문장은 다음과 같이 IF문 2번 쓰는 것으로 해결될 수 있다.

IF (조건) THEN DO; 문장1(들); END

IF (반대 조건) THEN DO; 문장2(들); END;

EXAMPLE: if ~ then; else 사용하기

🖫UNIV.txt 자료: 교직원 중 박사 학위 소지자(PHD)가 90% 이상이면 그룹을 (변수 이름: GROUP2) 'HIGH' 그렇지 않으면 'LOW'라 하자. PRINT PROCEDURE에서 원하는 변수만 출력하려면 VAR 문장을 사용하면 된다.

확장 편집기 - 제목없음1 *

```
DATA UNIV;

DATA UNIV4;
     SET UNIV;
     IF (PHD>=90) THEN GROUP2='HIGH';
     ELSE GROUP2='LOW';
RUN;
PROC PRINT DATA=UNIV4;
     VAR PHD GROUP2;
RUN;
```

Obs	PHD	GROUP2
1	81	LOW
2	93	HIGH
3	90	HIGH
4	89	LOW
5	79	LOW
6	83	LOW
7	75	LOW
8	86	LOW
9	81	LOW
10	91	HIGH
11	99	HIGH

아래 프로그램은 위와 동일한 작업이다. 그러나 결과를 보면 “HIGH”가 다 출력된 것이 아니라 “HIG”로 세 글자만 출력되었다. 이는 GROUP2 변수의 초기치가 설정될 때 “LOW”로 세 글자로 되었기 때문이다. 이런 경우 GROUP2=’LOW ’로 설정하여 주면 된다.

확장 편집기 - 제목없음1 *

```
DATA UNIV;

DATA UNIV5;
     SET UNIV;
     IF (PHD<90) THEN GROUP2='LOW';
     ELSE GROUP2='HIGH';
RUN;
```

Obs	PHD	GROUP2
1	81	LOW
2	93	HIG
3	90	HIG
4	89	LOW
5	79	LOW
6	83	LOW
7	75	LOW
8	86	LOW
9	81	LOW
10	91	HIG

IF~THEN ELSE 문을 모르는 경우 다음과 같이 프로그램 하면 된다. 물론 IF 문을 두 번 사용함으로써 실행 시간이 약간 길어지기는 하겠지만 컴퓨터 성능의 발달로 인하여 실제 사용자들은 차이를 느끼지 못할 것이다.

```
확장 편집기 - 제목없음1 *
DATA UNIV;

DATA UNIV6;
     SET UNIV;
     IF (PHD<90) THEN GROUP2='LOW ';
     IF (PHD>=90) THEN GROUP2='HIGH';
RUN;
TITLE 'UNIV6 DATA';
PROC PRINT DATA=UNIV6;
     VAR PHD GROUP2;
RUN;
```

UNIV6 DATA

Obs	PHD	GROUP2
1	81	LOW
2	93	HIGH
3	90	HIGH
4	89	LOW
5	79	LOW
6	83	LOW
7	75	LOW
8	86	LOW
9	81	LOW
10	91	HIGH
11	99	HIGH
12	96	HIGH

3.3.4 DO문

DO UNTIL, DO WHILE 문이 있으나 사용 빈도가 매우 적을 뿐 아니라 고급 프로그램이 주로 사용되므로 본 책에서는 생략하기로 하자. DO 문은 설정된 초기 값과 말기 값까지 설정된 증가분만큼 증가시키면서 문장을 실행한다.

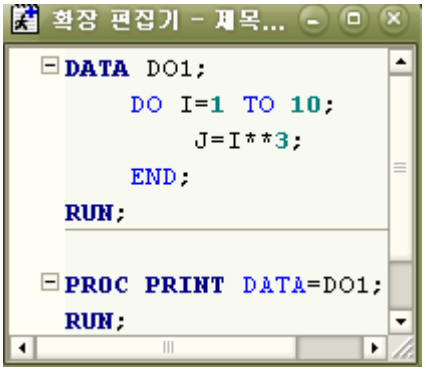
DO 변수이름=초기 값 **TO** 말기 값 **(BY** 증가분**);**

    문장(들)**;**

END;

EXAMPLE: DO문 사용하기

다음 프로그램은 1부터 10까지의 정수의 세제곱을 계산하는 프로그램이다. 변수 I를 1부터 10까지 1씩 증가시키면서 DO loop 안에 있는 문장을 반복 실행한다. 증가분(BY) 설정이 없었으므로 1씩 증가한다. I=1, 2, …, 10으로 증가되면서 "**J=I**3;**"을 반복한다. 그러므로 J=3, 8, …, 1000이 결과이다.

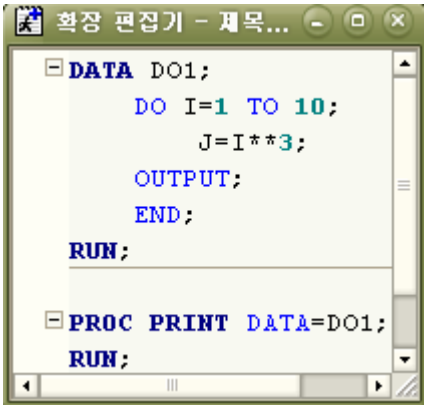


```
확장 편집기 - 제목...
DATA DO1;
    DO I=1 TO 10;
        J=I**3;
    END;
RUN;

PROC PRINT DATA=DO1;
RUN;
```

SAS 시스템

Obs	I	J
1	11	1000

그런데 위 프로그램을 실행시키면 출력 결과는 마지막 한 개만 나온다. J=1000이 맞는데 변수 I는 하나 더 증가된 11이다. 원하는 결과를 얻기 위해서는 OUTPUT을 사용하면 된다. OUTPUT은 현재 변수 값들을 데이터에 저장하라는 의미이다.



```
확장 편집기 - 제목...
DATA DO1;
    DO I=1 TO 10;
        J=I**3;
    OUTPUT;
    END;
RUN;

PROC PRINT DATA=DO1;
RUN;
```

SAS 시스템

Obs	I	J
1	1	1
2	2	8
3	3	27
4	4	64
5	5	125
6	6	216
7	7	343
8	8	512
9	9	729
10	10	1000

다음 프로그램은 증가 설정하는 BY 옵션을 사용한 경우이다. 말기 값이 10이나 정수 9 이후 값이 13이므로 9까지만 저장된다.

```
확장 편집기 - 제목없음1 *
DATA DO1;
    DO I=1 TO 10 BY 4;
        J=I**3;
        OUTPUT;
    END;
RUN;
TITLE 'DO1 DATA';
PROC PRINT DATA=DO1;
RUN;
```

DO1 DATA

Obs	I	J
1	1	1
2	5	125
3	9	729

EXAMPLE: DO문 사용하기(2)

다음은 일정한 숫자만큼 증가시키지 않는 경우 예제이다.

```
확장 편집기 - 제목...
DATA DO2;
    DO I=1, 2, 5;
        J=I**3;
    OUTPUT;
    END;
RUN;

PROC PRINT DATA=DO2;
RUN;
```

SAS 시스템

Obs	I	J
1	1	1
2	2	8
3	5	125

문자열을 사용할 수 있다. 문자열을 사용할 때는 “ ” 혹은 ‘ ’을 사용해야 한다. “^ =”의 의미는 같지 않다는 의미이다. 즉 (I^='남자')는 변수 I가 '남자'가 아닐 때의 조건이다.

확장 편집기 - 제목없음1 *

```
DATA DO3;
     DO I='남자', '여자';
          IF (I='남자') THEN K=0;
          IF (I^='남자') THEN K=1;
          OUTPUT;
     END;
RUN;

PROC PRINT DATA=DO3;
RUN;
```

SAS 시스템

Obs	I	K
1	남자	0
2	여자	1

EXAMPLE: DO문 사용하여 데이터 입력하기

다음은 세 개의 그룹(1, 2, 3)으로부터 남자 3명 여자 3명을 추출하여 IQ를 조사한 데이터이다. 분석을 위하여 SAS 데이터를 입력해 보자.

그룹1		그룹2		그룹3	
남자	여자	남자	여자	남자	여자
125, 120, 115	113, 110, 135	123, 118, 130	100, 130, 130	110, 110, 115	115, 120, 110

아래와 같이 변수 4개(그룹, 성별, 반복, IQ), 관측치 18개 입력하면 된다. 이처럼 DO 문을 사용하지 않고 SAS 데이터로 만들 수 있다.

확장 편집기 - 제목없음1 *

```
DATA DO4;
     INPUT GROUP GENDER $ R IQ @@;
     CARDS;
1 남자 1 125 1 남자 2 120 1 남자 3 115
1 여자 1 113 1 여자 2 110 1 여자 3 135
2 남자 1 123 2 남자 2 118 2 남자 3 130
2 여자 1 100 2 여자 2 130 2 여자 3 130
3 남자 1 110 3 남자 2 110 3 남자 3 115
3 여자 1 115 3 여자 2 120 3 여자 3 110
;
RUN;

PROC PRINT DATA=DO4;
RUN;
```

SAS 시스템

Obs	GROUP	GENDER	R	IQ
1	1	남자	1	125
2	1	남자	2	120
3	1	남자	3	115
4	1	여자	1	113
5	1	여자	2	110
6	1	여자	3	135
7	2	남자	1	123
8	2	남자	2	118
9	2	남자	3	130
10	2	여자	1	100
11	2	여자	2	130
12	2	여자	3	130
13	3	남자	1	110
14	3	남자	2	110
15	3	남자	3	115
16	3	여자	1	115
17	3	여자	2	120
18	3	여자	3	110

위의 데이터처럼 변수 값이 반복되는 경우 DO 문을 사용하면 간단하게 입력할 수 있다. 이런 경우는 드물고 입력해야 할 데이터가 많은 경우가 아니면 효율성이 떨어지므로 자주 사용되는 예제는 아니다.

확장 편집기 - 제목없음1 *

```
DATA DO5;
     DO GROUP=1 TO 3;
          DO GENDER='남자', '여자';
               DO R=1 TO 3;
                    INPUT IQ @@;
                    OUTPUT;
               END;
          END;
     END;
     CARDS;
125 120 115 113 110 135 123 118 130
100 130 130 110 110 115 115 120 110
;
RUN;

PROC PRINT DATA=DO5;
RUN;
```

EXAMPLE: DO문 두 번 사용하기

이단~십이단을 출력해 보자. 앞의 숫자 반복을 위하여 DO문 사용, 뒤에 숫자 반복을 위하여 DO문을 사용하면 된다. NOOBS 옵션의 의미는 출력할 때 앞에 OBS를 출력하지 말라는 옵션이다. **OUTPUT;**을 사용함으로써 그 때의 변수 관측치를 저장하도록 한다.

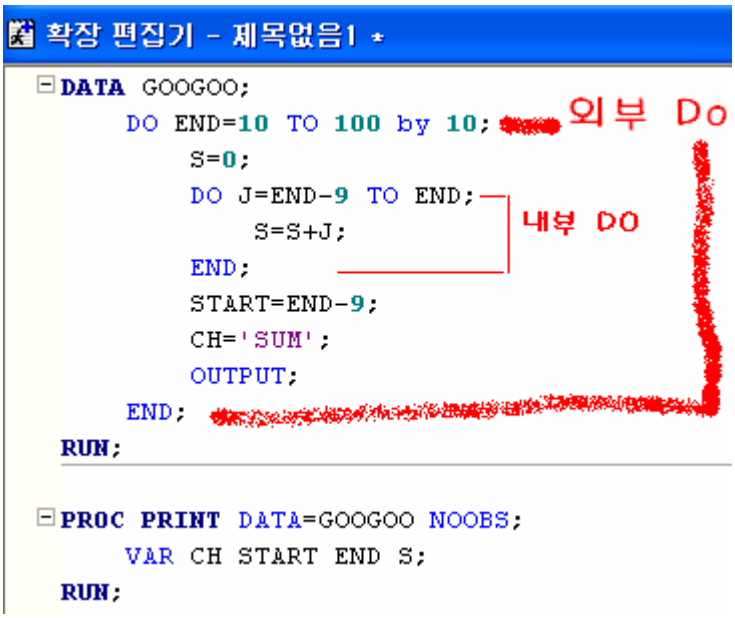
확장 편집기 - 제목없음1 *

```
DATA GOOGOO;
     CH1='*';CH2='=';
     DO I=2 TO 12;
          DO J=1 TO 12;
          R=I*J;
          OUTPUT;
          END;
     END;
RUN;

PROC PRINT DATA=GOOGOO NOOBS;
     VAR I CH1 J CH2 R;
RUN;
```

I	CH1	J	CH2	R
2	*	1	=	2
2	*	2	=	4
2	*	3	=	6
2	*	4	=	8
2	*	5	=	10
2	*	6	=	12
2	*	7	=	14
2	*	8	=	16
2	*	9	=	18
2	*	10	=	20
2	*	11	=	22
2	*	12	=	24
3	*	1	=	3
3	*	2	=	6
3	*	3	=	9

EXAMPLE: DO문 두 번 사용하기(2)

10씩 증가하면서 합을 출력하는 프로그램을 작성해 보자. 1부터 10까지의 합계, 11-20까지의 합계, ... 이렇게 100까지 출력하는 프로그램을 작성하자. S=0; 이 행이 실행될 때 변수 S의 값이 0으로 설정된다. OUTPUT;을 내부 DO 루프 밖에 사용함으로써 내부 DO 루프 실행이 끝난 후 그 결과를 저장하게 된다. 그러므로 변수 END가 반복되는 회수만큼 저장된다. 그러므로 관측치가 10개이다. OUTPUT;을 내부 DO문 안에 사용하면 관측치가 100개 출력될 것이다.




```
DATA GOOGOO;
    DO END=10 TO 100 by 10;
        S=0;
        DO J=END-9 TO END;
            S=S+J;
        END;
        START=END-9;
        CH='SUM';
        OUTPUT;
    END;
RUN;

PROC PRINT DATA=GOOGOO NOOBS;
    VAR CH START END S;
RUN;
```

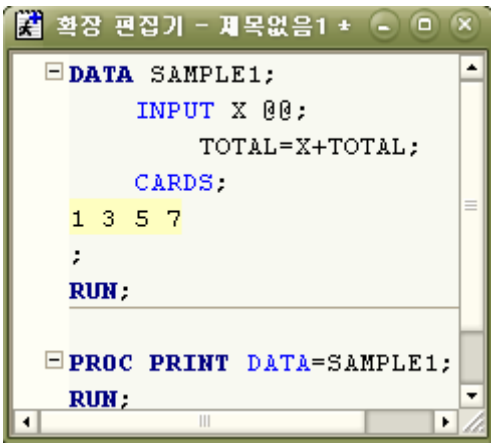
CH	START	END	S
SUM	1	10	55
SUM	11	20	155
SUM	21	30	255
SUM	31	40	355
SUM	41	50	455
SUM	51	60	555
SUM	61	70	655
SUM	71	80	755
SUM	81	90	855
SUM	91	100	955

3.3.5 RETAIN 문

RETAIN 문은 변수의 초기 값 설정과 저장된 값을 유지하기 위해 사용된다. 예제 중심으로 사용 방법을 설명하기로 하자.

EXAMPLE: RETAIN문 사용하기

SAS 데이터에서 새로운 변수에 초기치를 할당하기 위하여 사용된다. 만약 초기치를 할당하지 않으면 결측치(.)로 배정된다.

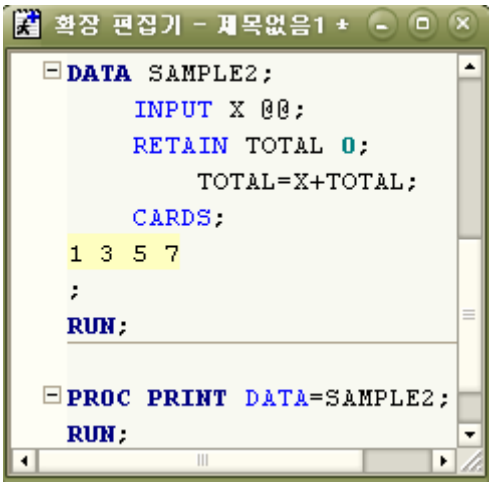


```
DATA SAMPLE1;
     INPUT X @@;
          TOTAL=X+TOTAL;
     CARDS;
1 3 5 7
;
RUN;

PROC PRINT DATA=SAMPLE1;
RUN;
```

SAS 시스템

Obs	X	TOTAL
1	1	.
2	3	.
3	5	.
4	7	.

RETAIN 문을 사용하여 TOTAL 변수의 초기 값으로 0이 할당된다. 첫 행은 X=1이고 왼쪽 TOTAL=0이므로 TOTAL에는 1이 저장된다. 두 번째 행에는 X=3이고 왼쪽 TOTAL(이전 TOTAL 값)=1이므로 TOTAL에는 4가 저장된다.



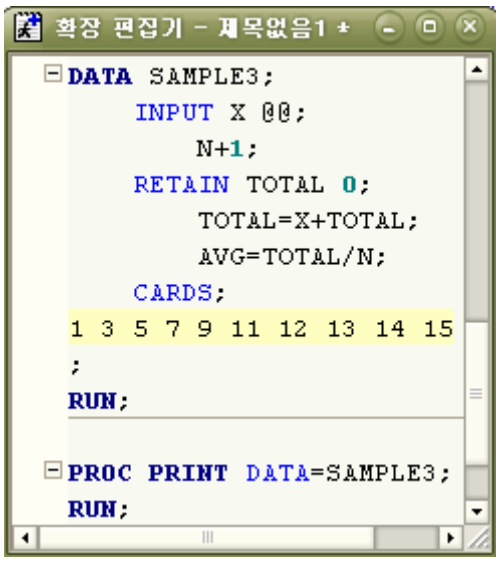
```
DATA SAMPLE2;
     INPUT X @@;
     RETAIN TOTAL 0;
          TOTAL=X+TOTAL;
     CARDS;
1 3 5 7
;
RUN;

PROC PRINT DATA=SAMPLE2;
RUN;
```

SAS 시스템

Obs	X	TOTAL
1	1	1
2	3	4
3	5	9
4	7	16

EXAMPLE: RETAIN문 사용하기(2)

자료의 평균을 구하는 프로그램을 RETAIN 문을 이용하여 작성해보자. 마지막 행의 AVG 변수의 값이 평균이다.



```
확장 편집기 - 제목없음1 *
DATA SAMPLE3;
    INPUT X @@;
        N+1;
    RETAIN TOTAL 0;
        TOTAL=X+TOTAL;
        AVG=TOTAL/N;
    CARDS;
1 3 5 7 9 11 12 13 14 15
;
RUN;

PROC PRINT DATA=SAMPLE3;
RUN;
```

SAS 시스템

Obs	X	N	TOTAL	AVG
1	1	1	1	1.00000
2	3	2	4	2.00000
3	5	3	9	3.00000
4	7	4	16	4.00000
5	9	5	25	5.00000
6	11	6	36	6.00000
7	12	7	48	6.85714
8	13	8	61	7.62500
9	14	9	75	8.33333
10	15	10	90	9.00000

3.4 데이터 합치기

3.4.1 SET과 MERGE

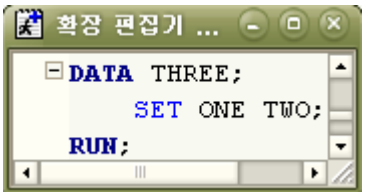
SAS 데이터를 합치는 방법으로 SET 문과 MERGE 문이 있다. SET은 두 데이터를 세로로 합치는 것이고 MERGE 문은 가로로 합치게 된다.

EXAMPLE: SET문과 MERGE문 사용하기

다음과 같이 두 개의 DTAT SET(ONE, TWO)이 있다고 가정하자. ONE에는 3개의변수 X1, X2, X3가 있고, TWO에는 3개의 변수 X1, X2, X4가 있을 때 SET과 MERGE를 이용하여 자료를 합칠 경우 어떠한 결과가 나오나 확인해보자.

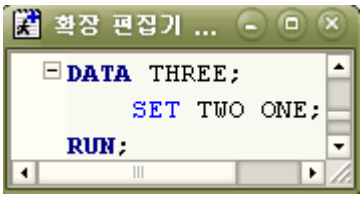
데이터 ONE	데이터 TWO
DATA ONE; INPUT X1 X2 X3; DATALINES; 1 2 3 4 5 6 RUN;	DATA TWO; INPUT X1 X2 X4; DATALINES; 7 8 9 0 11 12 13 14 15 RUN;

SET 문은 데이터를 세로로 결합한다. 데이터 ONE을 위에 TWO를 아래에 결합한다. 없는 변수에 대해서는 결측치로 저장된다.



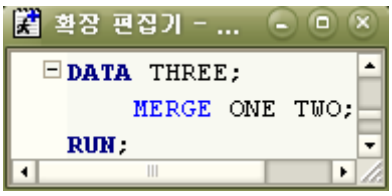
Obs	X1	X2	X3	X4
1	1	2	3	.
2	4	5	6	.
3	7	8	.	9
4	0	11	.	12
5	13	14	.	15

데이터 TWO를 위에 놓고 두 데이터를 결합한다. 결과는 위와 동일하다.



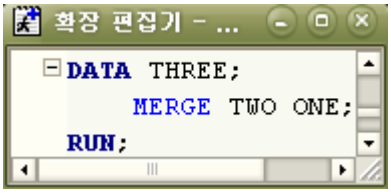
Obs	X1	X2	X4	X3
1	7	8	9	.
2	0	11	12	.
3	13	14	15	.
4	1	2	.	3
5	4	5	.	6

MERGE 문은 데이터를 가로로 결합한다. 두 데이터 동일한 변수가 있으면 뒤에 있는 데이터 변수 관측치 값이 엎어진다. 관측치 개수는 TWO의 것이 더 많으므로 3개가 된다. 동일 변수 X1, X2에 대해서는 뒤에 있는 TWO의 관측치 값이 저장된다. TWO에는 변수 X3의 관측치가 ONE의 관측치가 저장되고 3번째는 결측치로 처리된다.



Obs	X1	X2	X3	X4
1	7	8	3	9
2	0	11	6	12
3	13	14	.	15

변수 X1, X2에 대해서는 뒤에 있는 ONE의 관측치가 엎어진다. 그러나 ONE에는 관측치가 2개이므로 나머지 부분은 TWO 데이터 관측치가 그대로 있다. 변수 X3의 경우 TWO에는 없으므로 3번째 관측치는 결측치가 나타난다.



Obs	X1	X2	X4	X3
1	1	2	9	3
2	4	5	12	6
3	13	14	15	.

EXAMPLE: SET문과 MERGE문 사용하기(2)

3개의 DATA SET ONE, TWO, THREE가 각각 다음과 같을 때 SET과 MERGE문을 적절히 이용하여 데이터가 어떻게 합쳐지는가 보자.

```
DATA ONE;
    INPUT X1 X2 X3;
    CARDS;
1 2 3
4 5 6
7 8 9
;
RUN;
```

```
DATA TWO;
    INPUT X1 X4;
    CARDS;
0 1
2 3
4 5
6 7
;
RUN;
```

```
DATA THREE;
    INPUT X1 X2 X5;
    CARDS;
0 1 0
2 1 2
;
RUN;
```

SET 문을 이용한 경우

```
DATA FOUR;
    SET ONE TWO;
RUN;

PROC PRINT DATA=FOUR;
RUN;
```

SAS 시스템

Obs	X1	X2	X3	X4
1	1	2	3	.
2	4	5	6	.
3	7	8	9	.
4	10	11	.	12

```
DATA FIVE;
     SET ONE TWO THREE;
RUN;

PROC PRINT DATA=FIVE;
RUN;
```

SAS 시스템

Obs	X1	X2	X3	X4
1	1	2	3	.
2	4	5	6	.
3	7	8	9	.
4	10	11	.	12
5	1	2	3	.
6	4	5	6	.
7	7	8	9	.
8	10	11	.	12

```
DATA SIX;
     SET THREE ONE;
RUN;

PROC PRINT DATA=SIX;
RUN;
```

SAS 시스템

Obs	X1	X2	X3	X4
1	1	2	3	.
2	4	5	6	.
3	7	8	9	.
4	10	11	.	12
5	1	2	3	.
6	4	5	6	.
7	7	8	9	.

MERGE 문을 이용한 경우

```
DATA SEVEN;
     MERGE ONE TWO;
RUN;

PROC PRINT DATA=SEVEN;
RUN;
```

SAS 시스템

Obs	X1	X2	X3	X4
1	10	11	3	12
2	4	5	6	.
3	7	8	9	.

```
DATA EIGHT;
     MERGE ONE TWO THREE;
RUN;

PROC PRINT DATA=EIGHT;
RUN;
```

SAS 시스템

Obs	X1	X2	X3	X4
1	1	2	3	.
2	4	5	6	.
3	7	8	9	.
4	10	11	.	12

```
DATA NINE;
    MERGE THREE ONE;
RUN;

PROC PRINT DATA=NINE;
RUN;
```

SAS 시스템

Obs	X1	X2	X3	X4
1	1	2	3	.
2	4	5	6	.
3	7	8	9	.
4	10	11	.	12

3.4.2 UPDATE 문

가로 결합이고 뒤에 데이터가 앞의 데이터 위에 덮혀지는 점에서 MERGE 문과 유사하나 뒤의 데이터에 결측치가 있을 경우 이전 데이터 값이 저장된다. 예를 들어 보자.

확장 편집기 - 제목없음2 *

```
DATA OLD;
    INPUT ID X1 X2;
    DATALINES;
    1 12 23
    2 1 3
    3 4 5
RUN;
```

```
DATA NEW;
    INPUT ID X2;
    CARDS;
    2 3
    3 .
RUN;
```

```
DATA MG;
    MERGE OLD NEW;
    BY ID;
RUN;
```

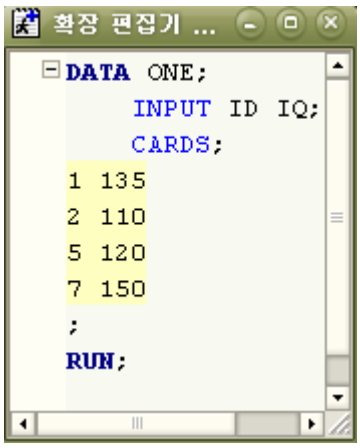
Obs	ID	X1	X2
1	1	12	23
2	2	1	3
3	3	4	.

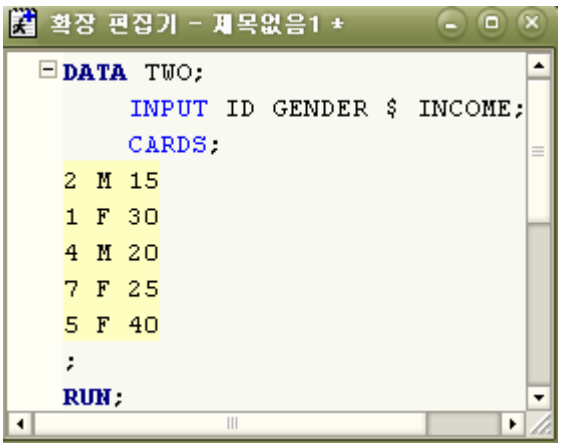
```
DATA UP;
    UPDATE OLD NEW;
    BY ID;
RUN;
```

Obs	ID	X1	X2
1	1	12	23
2	2	1	3
3	3	4	5

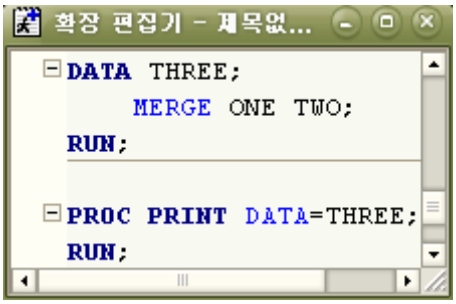
3.4.3 공통 변수 이용하여 합치기

고객에 대한 정보가 두 데이터에 나뉘어 저장되어 있다고 하자. 예를 들어 ONE이라는 데이터에는 IQ, 데이터 TWO에는 성별, 소득의 정보가 있다. 물론 고객을 식별하는 ID변수는 모두 있다. 어떻게 합칠 것인가? 가로로 합치는 MERGE 문을 생각하게 될 것이다.





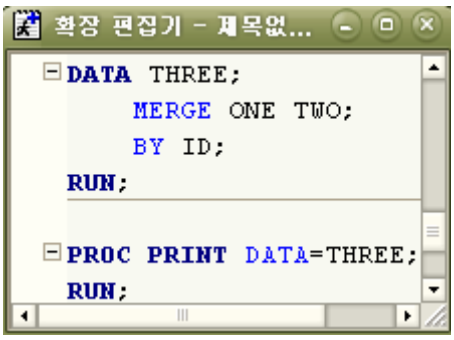
MERGE 문을 사용하면 동일 변수 이름의 ID의 경우 뒤에 데이터로 엎어져 버리므로 다음 결과를 얻는다.



SAS 시스템

Obs	ID	IQ	GENDER	INCOME
1	2	135	M	15
2	1	110	F	30
3	4	120	M	20
4	7	150	F	25
5	5	.	F	40

이런 문제를 해결하기 위한 방법으로 BY 문을 사용하게 된다. **BY ID;**의 의미는 변수 ID에 의해 MERGE 작업을 시행하라는 것이다. 즉 ID가 같은 값을 찾아 결합하게 된다.



```
DATA THREE;
    MERGE ONE TWO;
    BY ID;
RUN;

PROC PRINT DATA=THREE;
RUN;
```

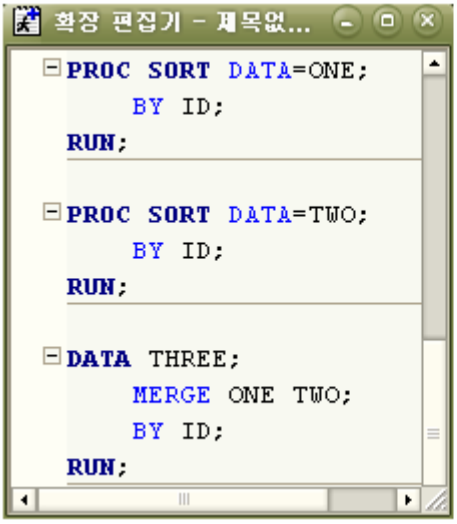
SAS 시스템

Obs	ID	IQ	GENDER	INCOME
1	2	135	M	15
2	1	110	F	30
3	4	120	M	20
4	7	150	F	25
5	5	.	F	40

그러나 출력 결과에 변화가 없다. 이상하다. 앞에서 설명하였듯이 로그 창을 살펴보는 습관을 갖자. 오류 메시지가 있다. 의미는 데이터 TWO가 정렬(sort)되어 있지 않았다는 것이다. SAS에서 BY문을 쓰려면 BY문에 지정된 변수에 의해 그 데이터가 정렬되어 있어야 오류가 발생하지 않는다. 데이터 단계 뿐 아니라 PROCEDURE 단계에서 사용되는 BY 문도 동일한 규칙을 적용 받는다.

```
1059  data three;
1060      merge one two;
1061      by id;
1062  run;

ERROR: BY variables are not properly sorted on data set WORK.TWO.
id=2 ig=110 gender=m income=15 FIRST.id=1 LAST.id=1 ERROR=1 N=2
```

이 문제를 해결하려면 데이터를 ID 변수에 의해 크기 순으로 정렬해야 한다. 이 때 사용되는 PROCEDURE는 SORT이다. 이에 대한 자세한 내용은 5장에서 다루기로 한다.



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

PROC SORT DATA=TWO;
    BY ID;
RUN;

DATA THREE;
    MERGE ONE TWO;
    BY ID;
RUN;
```

Obs	ID	IQ	GENDER	INCOME
1	1	135	F	30
2	2	110	M	15
3	4	.	M	20
4	5	120	F	40
5	7	150	F	25

이런 데이터 합치기는 새로운 관측에 의해 데이터를 갱신(update)하는 경우에도 사용된다. 데이터 OLD를 기존의 고객 정보, 데이터 NEW를 새로운 정보라 하자. 새로운 정보에 의해 기존 데이터를 갱신해 보자. 변수 ID는 학번이고 GPA는 전 학년 평점이라 하자. 학번 2의 학생의 평점이 3.42로 수정되었다고 하자. 두 데이터 모두 ID에 의해 정렬되어 있으므로 굳이 정렬할 필요는 없다.

확장 편집기 - 제목없음1 *

```
DATA OLD;
     INPUT ID GPA;
     DATALINES;
     1 3.24
     2 2.74
     3 4.01
RUN;

DATA NEW;
     INPUT ID GPA;
     DATALINES;
     2 3.42
RUN;
```

확장 편집기 - 제목없음1 *

```
DATA OLD;
DATA NEW;

DATA UP;
     MERGE OLD NEW;
     BY ID;
RUN;

PROC PRINT DATA=UP;
RUN;
```

Obs	ID	GPA
1	1	3.24
2	2	3.42
3	3	4.01

3.4.4 필요한 변수만 가져오기

두 데이터에 동일한 변수(예: ID)가 있어 이 변수에 의해 두 데이터를 합치려고 한다면 MEREGE 문을 사용하려고 할 것이다.

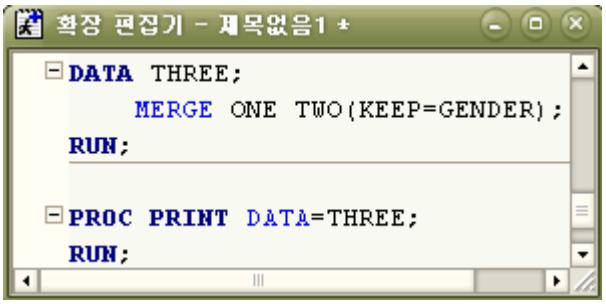
확장 편집기 - 제...

```
DATA ONE;
     INPUT NAME $ IQ;
     CARDS;
LEE 150
KIM 145
YOO 140
;
RUN;
```

확장 편집기 - 제목없음1 *

```
DATA TWO;
     INPUT NAME $ GENDER $ INCOME;
     CARDS;
LEE F 50000
KIM M 45000
YOO F 40000
;
RUN;
```

아래 프로그램은 **DATA TWO**로부터 **GENDER**라는 변수만 가져와 **ONE**에 **MERGE**한다**.** 만약 소득 변수도 가져오기 원하면 **TWO(KEEP=GENDER INCOME);** 이렇게 사용하면 된다**.** 만약 필요 없는 변수를 제외하려면 **TWO(DROP=INCOME);** 이렇게 사용하면 된다**.**

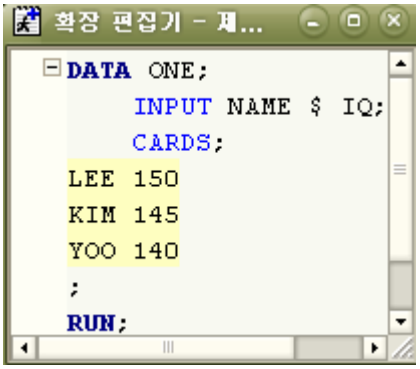


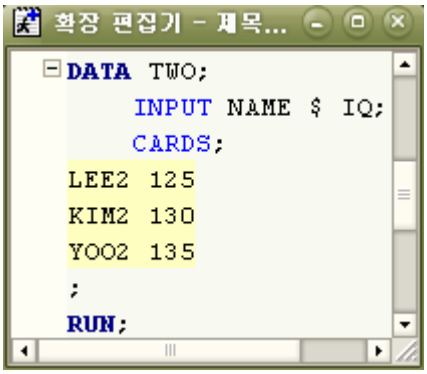
SAS 시스템

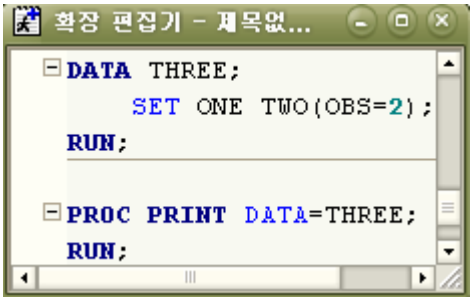
Obs	NAME	IQ	GENDER
1	LEE	150	F
2	KIM	145	M
3	YOO	140	F

3.4.5 일부 관측치만 가져오기

다음 프로그램은 두 개의 데이터를 SET 문을 사용하여 합치는데 TWO 데이터에서 처음 2개 데이터만 가져올 때 사용된다.







SAS 시스템

Obs	NAME	IQ
1	LEE	150
2	KIM	145
3	YOO	140
4	LEE2	125
5	KIM2	130

DROP과 KEEP 문장은 변수를 원하는 변수를 제거하거나 그 변수들만 저장하고자 할 때 사용한다. DELETE는 관측치를 제외할 때 사용하는 것임을 기억하기 바란다.

EXAMPLE: DROP문과 KEEP문 사용하기

다음 예에서는 두 개의 DATA SET ONE과 TWO를 SET 문장을 이용하여 합치는 프로그램이다. DATA THREE는 KEEP을 이용하여 X3, X4, X5 변수를 선택한 것이고 DATA FOUR는 DROP을 이용하여 X1, X2 변수를 제외한 것이므로 데이터 THREE, FOUR의 저장 결과는 동일하다.

확장 편집기 - 제목없음1 *

```
DATA ONE;
     INPUT X1 X2 X3 X4 @@;
     CARDS;
1  2  3  4  5  6  7  8
9 10 11 12 13 14 15 16
;
RUN;

DATA TWO;
     INPUT X1 X2 X5 @@;
     CARDS;
2 4 6 8 10 12 14 16 18
;
RUN;
```

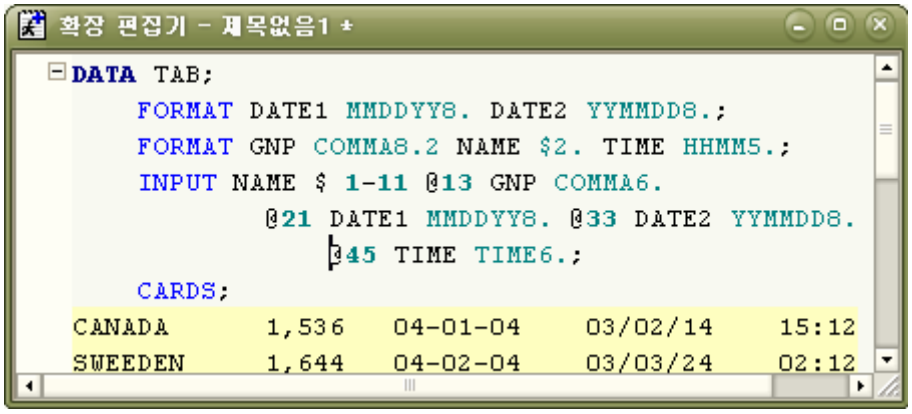
확장 편집기 - 제목없음1 *

```
DATA THREE;
     SET ONE TWO;
     KEEP X3 X4 X5;
RUN;

DATA FOUR;
     SET ONE TWO;
     DROP X1 X2;
RUN;
```

3.5 알아두면 편리한 기능

3.5.1 INFORMAT과 FORMAT

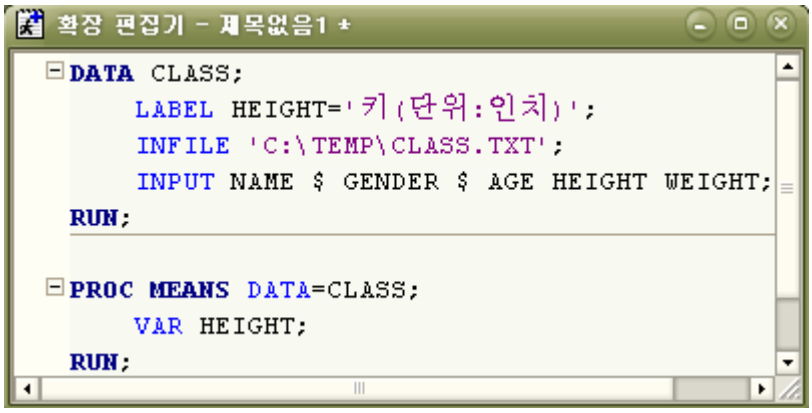
데이터를 읽어 올 때 형식을 지정하는 것은 INFORMAT(INPUT 문장에 옵션 형식으로 지정한다)이라 하고 결과 창에 출력하는 형식을 지정하는 것을 FORMAT(INPUT 문장 위에 문장 형식으로 지정한다)이라 한다.



3.5.2 Label 문

변수가 어떤 내용인지 설명을 부여하는 문장이다. 다음은 CLASS 데이터의 Height 변수에 대한 Label 문 부여 방법이다. 다른 변수에 대해서도 동일한 방법으로 부여할 수 있다. MEANS는 측정형 변수의 기초 통계량을 구하는 PROC이다.

변수 이름을 부여할 때 적절한 것을 사용한 경우에는 LABEL 문의 효용은 줄어든다. 그러나 변수가 많아 $X1, X2,..,$ 등으로 이름을 부여한 경우 LABEL 문이 사용하여 변수들을 구별하는 것이 좋다.

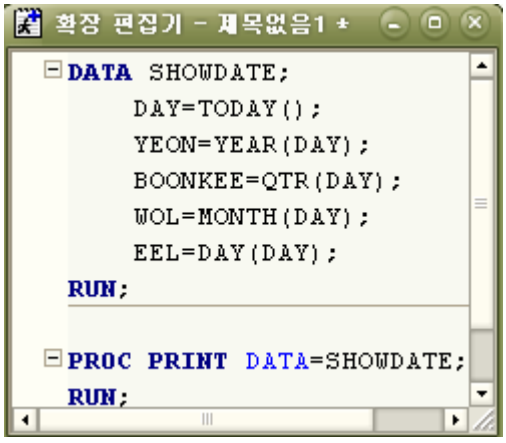


분석 변수 : HEIGHT 키(단위:인치)

N	평균값	표준편차	최소값	최대값
19	62.3368421	5.1270752	51.3000000	72.0000000

3.5.3 날짜 보기

컴퓨터 현재 시각, 날짜를 출력하는 프로그램이다. 날짜의 경우 1960년 1월 1일을 0으로 하여 매일 1씩 증가시킨다.



SAS 시스템

Obs	DAY	YEON	BOONKEE	WOL	EEL
1	16275	2004	3	7	23

3.5.4 FORMAT PROCEDURE

변수의 값에 대한 설명을 설정하는 PROC 단계이다. 사용자 정의의 FORMAT을 작성하는 PROC이다. 예를 들어 설명하는 것이 더 편리하다.

EXAMPLE: PROC FORMAT 사용하기

SASHELP 라이브러리에 있는 CLASS 데이터를 예제 데이터로 사용하자.

SAS 시스템

NAME	GENDER	AGE	HEIGHT	WEIGHT
Alfred	M	14	69.0	112.5
Alice	F	13	56.5	84.0
Barbara	F	13	65.3	98.0
Carol	F	14	62.8	102.5
Henry	M	14	63.5	102.5
James	M	12	57.3	83.0
Jane	F	12	59.8	84.5
Janet	F	15	62.5	112.5
Jeffrey	M	13	62.5	84.0
John	M	12	59.0	99.5
Joyce	F	11	51.3	50.5

FORMAT procedure를 사용하여 원하는 형식으로 출력해 보자. AGEFMT, GENDERFMT은 형식 이름이므로 사용자가 설정할 수 있으나 변수명+FMT로 하는 것이 구별하는데 유리하다. $의 의미는 변수의 관측치가 문자나 문자열일 경우 사용한다. FORMAT으로 설정되지 않은 관측치는 원래 값으로 출력된다. FORMAT procedure는 한 번 설정해 놓으면 향후 PRINT procedure에 계속 사용할 수 있다.

확장 편집기 - 제목없음1 *

```
PROC FORMAT;
    VALUE AGEFMT
          11-12='국민학생'
          13-15='중학생'
          16-18='고등학생';
    VALUE $GENDERFMT
          'M'='남자'
          'F'='여자';
RUN;

PROC PRINT DATA=CLASS;
    FORMAT AGE AGEFMT. GENDER $GENDERFMT.;
RUN;
```

Obs	NAME	GENDER	AGE	HEIGHT	WEIGHT
1	Alfred	남자	중학생	69.0	112.5
2	Alice	여자	중학생	56.5	84.0
3	Barbara	여자	중학생	65.3	98.0
4	Carol	여자	중학생	62.8	102.5
5	Henry	남자	중학생	63.5	102.5
6	James	남자	국민학생	57.3	83.0
7	Jane	여자	국민학생	59.8	84.5
8	Janet	여자	중학생	62.5	112.5
9	Jeffrey	남자	중학생	62.5	84.0
10	John	남자	국민학생	59.0	99.5
11	Joyce	여자	국민학생	51.3	50.5
12	Judy	여자	중학생	64.3	90.0
13	Louise	여자	국민학생	56.3	77.0
14	Mary	여자	중학생	66.5	112.0
15	Philip	남자	고등학생	72.0	150.0
16	Robert	남자	국민학생	64.8	128.0
17	Ronald	남자	중학생	67.0	133.0
18	Thomas	남자	국민학생	57.5	85.0
19	William	남자	중학생	66.5	112.0

이처럼 FORMAT 문은 데이터를 출력(PROC PRINT)할 때 각 변수의 값에 대한 설명이나 범주 그룹을 설정할 때 사용된다. 원 데이터의 값들은 변하는 것이 아니고 출력할 때만 설정된 형식(format)으로 출력 창에 출력되는 것이다.

EXAMPLE: PROC FORMAT 사용하기(2)

CEO.txt(나이, 연봉) 데이터를 우선 SAS 데이터 CEO를 만들어 보자.

```
DATA CEO;
     INFILE 'C:\TEMP\CEO.TXT';
     INPUT AGE SALARY;
RUN;

PROC PRINT DATA=CEO;
RUN;
```

SAS 시스템

Obs	AGE	SALARY
1	53	145
2	43	621
3	33	262
4	45	208
5	46	362
6	55	424
7	41	339
8	55	736
9	36	291
10	45	58

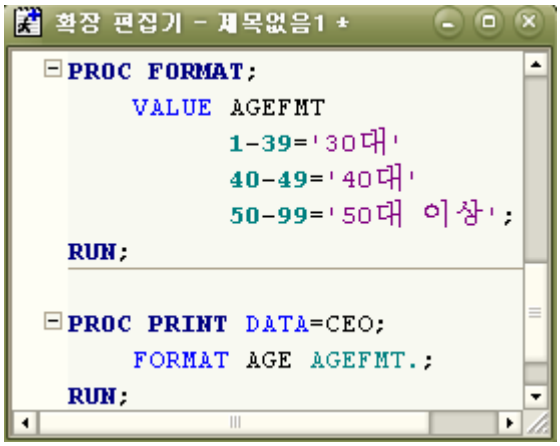
IF문 사용해 나이를 '30대', '40대', '50대 이상'으로 나누고 변수명을 AGE_G으로 하자. 그리고 LABEL 문을 사용하여 변수에 대한 설명을 붙여보자.

```
DATA CEO1;
     SET CEO;
     LABEL AGE_G='나이그룹' SALARY='연봉';
     AGE_G='30대';
     IF (AGE>=40) THEN AGE_G='40대';
     IF (AGE>=50) THEN AGE_G='50대이상';
RUN;

PROC MEANS DATA=CEO1;
     CLASS AGE_G;
     VAR SALARY;
RUN;
```

분석 변수 : SALARY 연봉

나이그룹	관측치 수	N	평균값	표준편차	최소값	최대값
30대	5	5	314.6000000	44.3260646	262.0000000	370.0000000
40대	19	19	406.2631579	220.3537162	58.0000000	862.0000000
50대	35	35	415.8285714	236.0218691	21.0000000	1103.00

FORMAT 문을 이용하여 나이 변수 AGE가 30대, 40대, 50대 이상으로 출력되게 하자.



```
확장 편집기 - 제목없음1 *
PROC FORMAT;
    VALUE AGEFMT
          1-39='30대'
          40-49='40대'
          50-99='50대 이상';
RUN;

PROC PRINT DATA=CEO;
    FORMAT AGE AGEFMT.;
RUN;
```

SAS 시스템

Obs	AGE	SALARY
1	50대 이상	145
2	40대	621
3	30대	262
4	40대	208
5	40대	362
6	50대 이상	424
7	40대	339
8	50대 이상	736
9	30대	291
10	40대	58
11	50대 이상	498
12	50대 이상	643
13	40대	390
14	40대	332
15	50대 이상	250

3.5.5 자동생성 변수

SAS 내에서 자동으로 생성되는 변수에는 "_이름_"에서 보는 것처럼 _가 변수명 양쪽에 붙는다. 가장 많이 사용되는 것이 "_N_"으로 Obs의 값과 동일하다. 그리고 SAS 분석 결과 얻어지는 변수들은 SAS가 자동으로 이름을 부여하는데 형식은 "_변수명_"이다.

EXAMPLE: _N_ 자동생성 변수

CEO 중 연봉이 큰 10명만으로 이루어진 새로운 데이터를 만들어보자. 연봉이 10번째인 CEO의 연봉이 "643"인줄 알고 있다면 IF문을 사용하면 된다.

```
확장 편집기 - 제목없음1 *
DATA CEO2;
    SET CEO;
    IF (SALARY>=643);
RUN;
PROC PRINT DATA=CEO2;
RUN;
```

Obs	AGE	SALARY
1	50	643
2	48	659
3	60	726
4	55	736
5	69	750
6	74	800
7	56	802
8	40	808
9	47	862
10	57	1103

연봉 서열 10번째에 대한 데이터가 없다면 다음 방법을 이용하면 된다. 데이터를 크기 순으로 정렬 한 후 열 번째 데이터까지만 저장하면 된다. 데이터를 크기 순으로 정렬하는 PROC는 SORT이다. BY문은 정렬하려는 변수 이름을 지정하는 것이고, DESCEDING 옵션은 크기의 역순으로 정렬하라는 의미이다.

데이터를 연봉 크기 순으로 역 정렬한 후 10번째 관측치까지만 얻으면 된다. SAS 데이터를 출력할 때 OBS라는 변수가 있는 것처럼 출력되지만 실제 데이터에는 없다. OBS와 같은 역할을 하는 변수가 자동 생성변수인 _N_이다.

확장 편집기 - 제목없음1 *

```
PROC SORT DATA=CEO;
     BY DESCENDING SALARY;
RUN;
DATA CEO2;
     SET CEO;
     IF (_N_<=10);
RUN;

PROC PRINT DATA=CEO2;
RUN;
```

Obs	AGE	SALARY
1	57	1103
2	47	862
3	40	808
4	56	802
5	74	800
6	69	750
7	55	736
8	60	726
9	48	659
10	50	643

3.5.6 SAS 데이터 나누기

3.3.1절에서는 IF문을 사용하여 일정한 조건에 따라 SAS 데이터의 일부 SAS 데이터로 만드는 방법을 알아보았다. 그 때는 데이터 단계 한 번에 하나의 SAS 데이터를 얻었으나 여기서는 두 개 이상의 SAS 데이터를 얻는 방법을 알아보자.

EXAMPLE: SAS 데이터 두 개 이상 만들기

🖫CLASS.txt 자료에서 다음 프로그램을 실행해 보자. 두 개의 SAS 데이터가 만들어지며 하나는 CLASSM(여기에는 성별이 M인 관측치), 다른 하나는 CLASSF(성별이 F인 관측치)이다. 성별 변수의 경우 범주가 2개이므로 아래 IF문장 대신 ELSE OUPUT CLASSF;을 사용해도 같은 결과를 얻는다.

확장 편집기 - 제목없음1 *

```
DATA CLASSM CLASSF;
     SET CLASS;
     IF (GENDER='M') THEN OUTPUT CLASSM;
     IF (GENDER='F') THEN OUTPUT CLASSF;
RUN;

PROC PRINT DATA=CLASSM;
RUN;
```

SAS 시스템

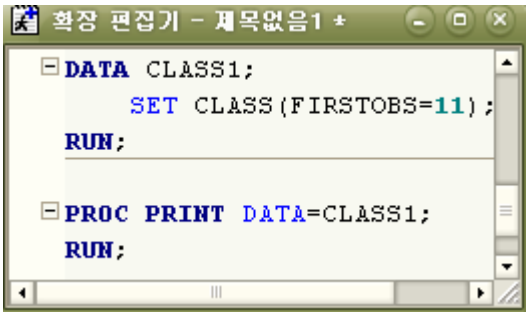
Obs	NAME	GENDER	AGE	HEIGHT	WEIGHT
1	Alfred	M	14	69.0	112.5
2	Henry	M	14	63.5	102.5
3	James	M	12	57.3	83.0
4	Jeffrey	M	13	62.5	84.0
5	John	M	12	59.0	99.5
6	Philip	M	16	72.0	150.0
7	Robert	M	12	64.8	128.0
8	Ronald	M	15	67.0	133.0
9	Thomas	M	11	57.5	85.0
10	William	M	15	66.5	112.0

3.5.7 FIRSTOBS와 OBS 옵션

SET 문장에 사용되는 옵션으로 FIRSTOBS는 시작 관측치를 지정하는 것이고 OBS는 관측치의 끝을 지정한다. FIRSTOBS가 생략되면 처음부터, OBS 옵션이 생략되면 끝까지 저장된다.

EXAMPLE: FIRST/OBS 옵션 사용하기

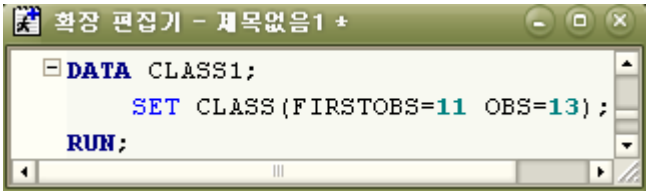
🖫CLASS.txt 자료에서 다음 프로그램을 실행해 보자. 11번째 관측치부터 CLASS3에 저장된다.



```
확장 편집기 - 제목없음1 *
DATA CLASS1;
    SET CLASS(FIRSTOBS=11);
RUN;

PROC PRINT DATA=CLASS1;
RUN;
```

NAME	GENDER	AGE	HEIGHT	WEIGHT
Joyce	F	11	51.3	50.5
Judy	F	14	64.3	90.0
Louise	F	12	56.3	77.0
Mary	F	15	66.5	112.0
Philip	M	16	72.0	150.0
Robert	M	12	64.8	128.0
Ronald	M	15	67.0	133.0
Thomas	M	11	57.5	85.0
William	M	15	66.5	112.0

아래 프로그램은 11번째부터 13번째 관측까지만 저장된다.



```
확장 편집기 - 제목없음1 *
DATA CLASS1;
    SET CLASS(FIRSTOBS=11 OBS=13);
RUN;
```

Obs	NAME	GENDER	AGE	HEIGHT	WEIGHT
1	Joyce	F	11	51.3	50.5
2	Judy	F	14	64.3	90.0
3	Louise	F	12	56.3	77.0

3.5.8 RENAME 문장

변수 이름을 바꿀 때 사용된다. 앞에서 설명하였듯이 SAS 데이터에서 변수를 제거하는 명령은 DROP, 원하는 변수만 저장하는 명령은 KEEP이다.

RENAME 이전 변수명=새로운 변수명**;**

EXAMPLE: RENAME 문장 사용하기

CLASS.txt 자료에서 변수 HEIGHT의 이름을 HEIGHT0로 변환하여 보자. 만약 RENAME 문장을 모른다면 다음 방법이 있다. HEIGHT와 동일한 HEIGHT0 만들고 DROP 문을 사용하여 HEIGHT 변수를 제거하면 된다.

3.sas *

```
DATA CLASS8;
    SET CLASS;
    HEIGHT0=HEIGHT;
    DROP HEIGHT;
RUN;
```

RENAME 문장을 사용하면 간단히 해결된다. 등호 왼쪽이 원 변수 이름임에 유의하기 바란다. 여러 변수를 다음 방법으로 동시에 이름 변경할 수 있다.

3.sas *

```
DATA CLASS8;
    SET CLASS;
    RENAME HEIGHT=HEIGHT1 WEIGHT=WEIGHT1;
RUN;
```

3.5.9 PUT 문 사용하기

PUT문은 데이터를 로그 창에 출력할 때 사용한다. SAS 데이터가 제대로 만들어졌는지 알려면 로그 창의 NOTE(노트)를 보면 된다. 그러나 만들어진 데이터 내용을 보려면 PROC PRINT을 사용하여 출력 창에 출력을 해야 한다. 이런 번거로움을 덜기 위하여 아래와 같이 PUT문을 사용하면 로그 창에 데이터가 출력된다.

cancer.sas *

```
DATA CANCER;
    INPUT TIME  TYPE $ @@;
    PUT TIME TYPE;
    CARDS;
124 Stomach 146 Stomach 64  Bronchus 1112 Stomach 450 Bronchus
42  Stomach 340 Stomach 155 Bronchus 46 Stomach 246 Bronchus
25  Stomach 396 Stomach 859 Bronchus 103 Stomach 166 Bronchus
45  Stomach 81  Bronchus 151 Bronchus 876 Stomach 63 Bronchus
412 Stomach 461 Bronchus 166 Bronchus 223 Bronchus 72 Bronchus
51  Stomach 20  Bronchus 37 Bronchus 138 Bronchus 245 Bronchus
RUN;
```

로그 - (제목없음)

```
160  DATA CANCER;
161       INPUT TIME  TYPE $ @@;
162       PUT TIME TYPE;
163       CARDS;

124 Stomach
146 Stomach
64 Bronchus
1112 Stomach
450 Bronchus
```

3.5.10 TITLE문과 FOOTNOTE문

TITLE문은 출력 결과의 제목을 붙이는 것으로 제목이 두 줄 이상이면 TITLE1, TITLE2 등으로 지정하면 된다. SAS를 새로 시작하면 디폴트 제목은 “SAS 시스템”이다. 한 번 지정된 제목은 다시 지정하지 않는 한 유효하다.

FOOTNOTE는 출력 결과 꼬리말이며 사용 방법은 TITLE과 동일하다. 다음 프로그램은 CANCER 데이터 출력 시 제목과 꼬리말을 출력한 것이다. PROC MEANS 앞에 제목 설정이 없으므로 PROC 결과 제목도 동일하다. 만약 제목을 없애고 싶다면 TITLE;(따옴표 사용하지 않음) 문장을 사용하면 된다. FOOTNOTE도 같은 방법으로 없앨 수 있다.

확장 편집기 - 제목없음2 *

```
TITLE1 '암 데이터 출력';
TITLE2 '시간/암 종류';
FOOTNOTE '2005년1월3일';
PROC PRINT DATA=CANCER;
RUN;

PROC MEANS DATA=CANCER MEAN STD;
     VAR TIME;
RUN;
```

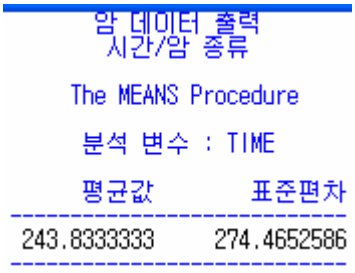
암 데이터 출력
시간/암 종류

Obs	TIME	TYPE
1	124	Stomach
2	146	Stomach
3	64	Bronchus
4	1112	Stomach
5	450	Bronchus

중간 생략

29	138	Bronchus
30	245	Bronchus

2005년1월3일


암 데이터 출력
시간/암 종류

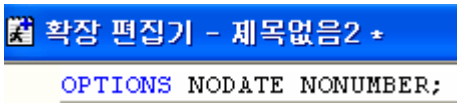
The MEANS Procedure

분석 변수 : TIME

평균값	표준편차
243.8333333	274.4652586

3.5.11 OPTIONS 문

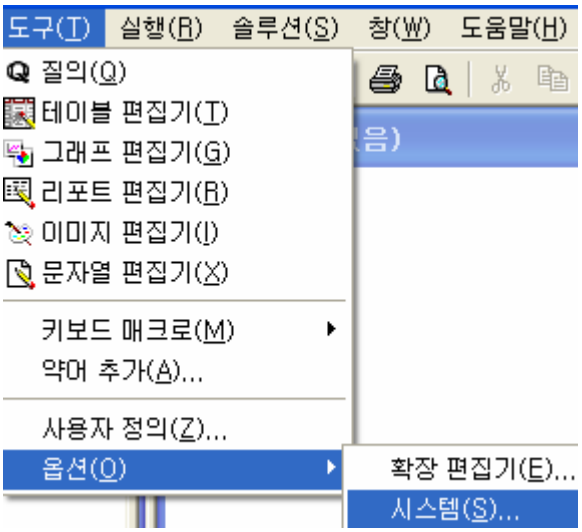
출력 결과에 대한 형식을 지정할 때 사용한다. 사용 방법은 다음과 같다. 이 문장은 프로그램 어디에서 사용 가능하며 문장 사용 이후부터 설정이 유효하다. SAS가 종료되면 설정한 옵션은 디폴트로 reset된다.

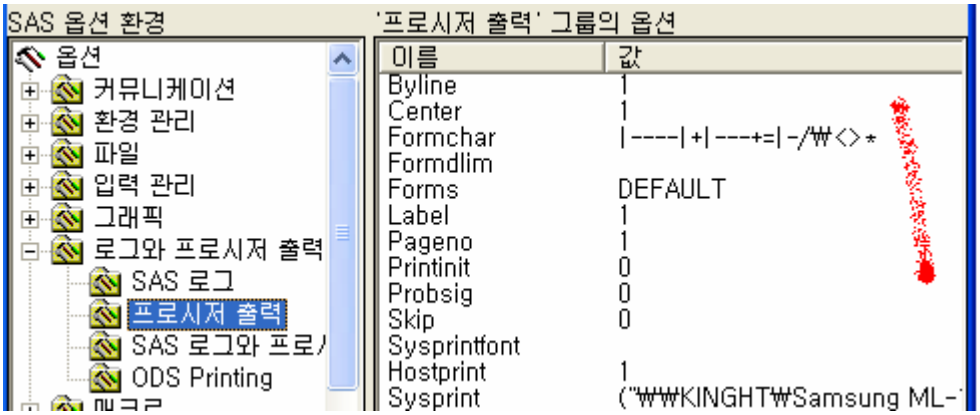

확장 편집기 - 제목없음2 *

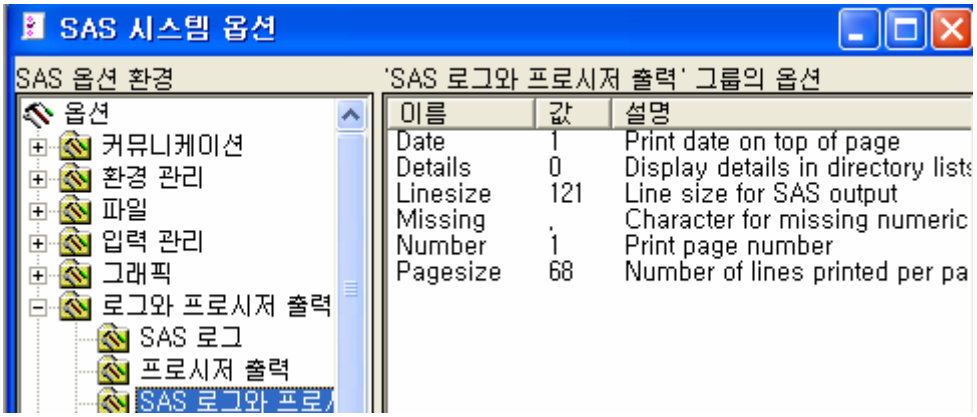
```
OPTIONS NODATE NONUMBER;
```

NODATE	출력 제목에 날짜/시간 출력하지 않음.
NONUMBER	출력 제목에 페이지 출력하지 않음.
NOCENTER	출력 결과가 왼쪽 정렬된다. 디폴트=가운데 정렬
PAGESIZE	출력 결과가 한 페이지에 몇 행인지 지정한다. 디폴트=68
LINESIZE	출력 결과 넓이를 지정한다. 디폴트=121

출력 결과에 대한 옵션 설정은 SAS 메뉴에서도 가능하다.





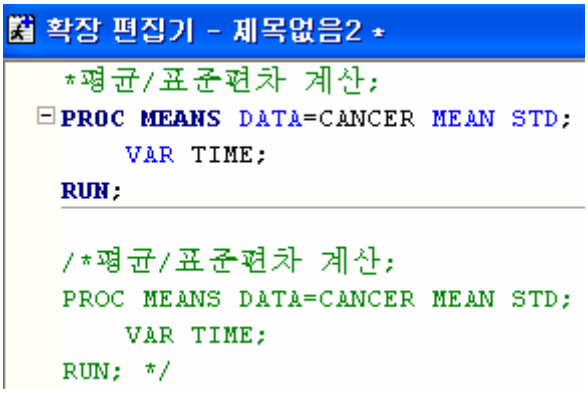


3.5.12 주석문

프로그램 내용을 설명하거나 프로그램 디버깅을 위하여 사용되는 문장으로 실행되지 않는다. 사용 방법은 다음과 같다. 주석문(comment)은 확장 편집기 내에서 자동적으로 초록색이 된다.

* (설명내용, 혹은 프로그램);	▶ 한 행일 경우
/* (설명 내용, 혹은 프로그램) */	▶ 두 행 이상인 경우

아래 첫 프로그램에서는 첫 행만 실행되지 않고 아래 프로그램에서는 프로그램 전체가 실행되지 않는다.



3.5.13 두 데이터 합칠 때 변수명으로 SAS 데이터 구별하기

두 SAS 데이터를 합칠 때 새로운 변수를 이용하여 데이터 소스를 구별해 보자.

EXAMPLE: 데이터 합칠 때 데이터 소속 구별하기

CLASS.txt 자료에서 다음 프로그램을 실행해 보자. CLASSM에서 가져온 관측치에는 DATA NAME에 남자, CLASSF에서 가져온 경우에는 여자가 저장되어 있다.

확장 편집기 - 제목없음1 *

```
DATA CLASS3;
     DATANAME='남자'; SET CLASSM; OUTPUT;
     DATANAME='여자'; SET CLASSF; OUTPUT;
RUN;

PROC PRINT DATA=CLASS3;
RUN;
```

Obs	DATANAME	NAME	GENDER	AGE	HEIGHT	WEIGHT
1	남자	Alfred	M	14	69.0	112.5
2	여자	Alice	F	13	56.5	84.0
3	남자	Henry	M	14	63.5	102.5
4	여자	Barbara	F	13	65.3	98.0
5	남자	James	M	12	57.3	83.0
6	여자	Carol	F	14	62.8	102.5
7	남자	Jeffrey	M	13	62.5	84.0
8	여자	Jane	F	12	59.8	84.5
9	남자	John	M	12	59.0	99.5

3.5.14 배열(array) 사용하기

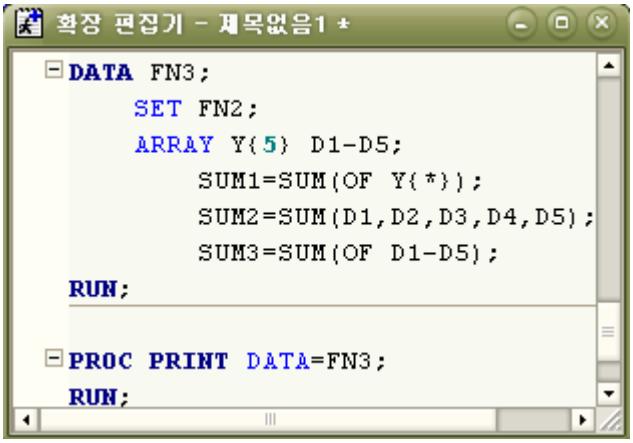
SAS에서 배열의 사용 예를 다양하지만 여기서는 변수 이름에 동일한 접두사(prefix)를 갖는 경우 변수 이름을 관리하는 예제를 살펴 보자. 다음 프로그램은 D1, D2, D3, D4, D5 변수명을 X에 의해 관리하는 프로그램이다.

확장 편집기 - 제목없...

```
DATA FN2;
     ARRAY X{5} D1-D5;
     DO I=1 TO 5;
          INPUT X{I} @@;
     END;
     DROP I;
     CARDS;
1 2 3 4 5 6 7 8 9 10
;
RUN;

PROC PRINT DATA=FN2;
RUN;
```

Obs	D1	D2	D3	D4	D5
1	1	2	3	4	5
2	6	7	8	9	10

배열은 DATA 문을 사용할 때마다 지정해 줘야 하며 이전에 배열에 대한 이름으로 사용하였던 것과 다른 이름을 사용할 수 있다.



Obs	D1	D2	D3	D4	D5	SUM1	SUM2	SUM3
1	1	2	3	4	5	15	15	15
2	6	7	8	9	10	40	40	40

$X_1 \sim Normal(1,1), X_2 \sim Normal(2,4), X_3 \sim Normal(3,9), X_4 \sim Normal(4,16), X_5 \sim Normal(5,25)$ 을 따르는 변수 10개를 생성하고, 각각의 seed는 1, 2, 3, 4, 5로 하자.

확장 편집기 - 제목없음1 *

```
DATA FN4;
     ARRAY X{5} X1-X5;
     DO I=1 TO 10;
          X1=RANNOR(1)*1+1;
          X2=RANNOR(2)*2+2;
          X3=RANNOR(3)*3+3;
          X4=RANNOR(4)*4+4;
          X5=RANNOR(5)*5+5;
     OUTPUT;
     END;
     DROP I;
RUN;

PROC PRINT DATA=FN4;
RUN;
```

SAS 시스템 15

Obs	X1	X2	X3	X4	X5
1	2.80482	1.84017	4.18973	-0.3333	16.1915
2	0.37577	3.02732	2.74017	1.6233	5.1595
3	0.26220	1.49972	5.05501	0.7834	1.2786
4	0.20450	2.68142	2.09847	-1.3994	7.1635
5	2.30572	4.85025	1.75260	10.4575	-0.2886
6	0.05167	3.90730	4.17594	3.6954	11.1028
7	0.36916	0.72848	1.97999	3.6949	9.8268
8	-0.21670	4.36898	1.96892	8.3610	4.3234
9	-0.35950	-2.66268	1.77094	6.6168	6.9963
10	0.53069	3.73266	0.22883	9.1099	-2.2630