6.

PROC 단계의 SAS 문장들

제 5장에서의 DATA 단계에서는 자료의 입력과 자료 구조의 변형만 가능하며, DATA 단계를 거쳐 SAS 자료를 만드는 목적은 수집된 자료를 잘 분석하기 위한 적절한 자료 구조를 만들기 위함이다. 이렇게 생성된 SAS 자료를 기초로 자료분석 등 각종 통계적 자료 처리는 모두 PROC 단계에서 이루어진다.

DATA 단계에서는

: 수집된 자료를 입력하여 SAS 시스템에서 분석 가능한 상태로 정리

PROC 단계에서는

: DATA 단계에서 만들어진 SAS 자료를 분석하거나 처리할 SAS 절차(procedure)를 호출하여 지정된 SAS 자료에 대한 통계 분석을 수행

PROC 단계는 PROC 문장으로 시작하는 일련의 SAS 문장들의 모임이다. DATA 단계와 마찬가지로 PROC 단계의 끝은 대체로 **RUN 문장**을 사용하여 마무리짓는다. 그러나 SAS 절차 중에는 대화식(interactive) 절차들이 있어 이런 절차들을 마감할 때는 **QUIT 문장**을 사용해야 할 경우도 있다.

PROC 단계의 문법은 각 SAS 절차마다 고유한 문법 체계가 존재하여 **절차마다 문법 체계가 상이**하더라도 많은 SAS 절차에서 **공통적으로 나타나는 문장들도 있으며** 이러한 문장들을 6장에서 다룬다(일부는 DATA 단계의 문장들과 겹침). 이 문장들은 대체로 **분석과 출력의 제어**에 사용되며 어느 SAS 절차에서든 같은 기능을 지닌다.

6.1 PROC 문장

PROC 단계의 시작은 반드시 PROC 문장으로 시작한다. 약자 PROC 대신 PROCEDURE를 사용해도 된다.

PROC *program* DATA=*SASdataset* [*options*];

program 은 호출하려는 SAS 절차의 이름
SASdataset 은 처리하려는 SAS 자료의 이름
DATA=*SASdatset* 사용하지 않으면 가장 최근에 만들어진 SAS 자료를 처리
options 는 사용자 선택사항

(예) **PROC** PRINT DATA=class2;
RUN; /* PRINT 절차를 호출하여 자료 class2를 인쇄 */
PROC MEANS DATA=class1 N MEAN STD;
RUN; /* 자료 class1에 MEANS 절차를 적용하여 자료에 포함된 관측 수(N), 자료에 포함된 모든 숫자변수들의 산술평균(MEAN), 표준편차(STD) 계산 */
⇨ 2개의 PROC 단계
(여기에서 사용된 자료 class2와 class1은 PROC 단계에서 사용되기 전에 이미 만들어져 있어야 함)

[참고] MEANS 절차에서 사용된 N, MEAN, STD 등은 사용자 선택 사항(options)에 해당되는데, **option은 SAS 절차마다 고유**하다.

[참고] PROC PRINT 절차에서의 FORMAT 문장 ([노트08] p.15 에서.....)

(예) DATA one;
INPUT x total @@;
* FORMAT total DOLLAR10.2; * (cf) PROC 단계에서 FORMAT문;
DATALINES;
3 28982 5 2849
RUN;
PROC PRINT DATA=one;
FORMAT total **DOLLAR**10.2; ***출력될 때에만 이 포맷 적용;**
RUN;

⇒

OBS	x	total
1	3	$28,982.00
2	5	$2,849.00

FORMAT *variable(s) format ;*

변수 값에 대한 출력형태를 지정하기 위한 것으로,
DATA 단계 및 **PROC 단계**에서 모두 사용 가능.
variables(s) 는 format 지정하고 싶은 변수(들) 지정
format 은 포맷 형식 지정 (**교재 5.2.8** 및 **주제별 추가노트 참고**)

w.d	: 숫자 변수에 소수점 표시 w는 소수점 포함 전체 자리수 d는 소수점 자리수
COMMAw.d	: 숫자변수에 콤마(,) 추가 w는 소수점과 콤마 포함한 전체 자리수
DOLLARw.d	: 달러 기호와 콤마(,) 추가
YENw.d	: ₩와 콤마(,) 추가 **(Korean Ver. SAS)** (￥은 Japanese Ver. SAS 설치시)

6.2 VAR 문장

SAS 절차로 분석하거나 처리할 변수 이름들을 지정하는 문장으로, 대부분의 SAS 절차에서 이 문장이 생략되면 지정된 SAS 자료에 포함된 분석 가능한 모든 변수들에 대해서 처리한다. 약자 VAR 대신 VARIABLE을 사용해도 된다.

VAR *variable* ;

variable 은 분석 변수 이름들

(예) 자료 class 에 name, sex, id_num, credit 라는 변수들이 순서대로 있다면,

```
PROC PRINT DATA=class;
RUN;
```

⇨ name, sex, id_num, credit 모든 변수들에 관한 관측들 인쇄

```
PROC PRINT DATA=class;
  VAR id_num name credit;
RUN;
```

⇨ id_num, name, credit 변수들에 대한 관측들만 인쇄
원 자료에는 name, id_num, credit 순으로 저정되어 있더라도 VAR 문장에서 지정한 순서인 id_num, name, credit 순으로 인쇄됨.

(예) 자료 class 에 credit 만 숫자변수라면,

```
PROC MEANS DATA=class;
  * VAR 문장 추가하면 VAR 문장에서 지정한 변수들에 대한 proc means ;
RUN;
```

⇨ 숫자변수 credit 에 대한 평균, 표준편차 등 인쇄

(예) PROC 단계에서 사용할 수 있는 **단축 용법**

NUMERIC	모든 숫자변수들 선택 시 지정
CHARACTER	모든 문자변수들 선택 시 지정
ALL	데이터에 존재하는 모든 변수들 지정

```
DATA one;
  input a b $ c $ d;
  DATALINES;
        .....
RUN;
PROC PRINT DATA=one;
  VAR _NUMERIC_;
RUN;
```

⇨ 변수 a 와 d 만 출력됨

6.3 BY 문장

자료분석 시 **변수의 값에 따라** 독립적으로 **분석**이 필요한 경우가 있다.
(예) 전국에서 임의 표집된 2,000명에게 통일에 대한 찬반 견해를 물었는데,
성별에 따라 반응이 어느 정도 다른지 확인하려면?
남/녀 두 그룹으로 분류한 후 각 자료에 대하여 독립적으로 찬반 성향을 조사한다.

SAS 에서는 SORT 절차와 BY 문장의 조합으로 작업이 이루어진다.

SORT 절차는 자료를 지정된 변수의 값에 따라 순서화(sorting)하는 절차로, 순서화의 기준이 되는 변수(들)을 BY 문장에서 지정한다. BY 문장 다음에 지정되는 변수들을 BY 변수라고 한다.

SORT 절차 이외의 모든 절차에서 BY 문장이 나타나면 BY 변수의 값에 따라 그룹별로 독립적으로 분석되는데, 이런 경우에는 **SORT 절차가 반드시 선행**되어야 한다. **(descending/ascending 도 맞춰서)**

① PROC SORT;
BY *variables*;
RUN;

② PROC *program*;
BY *variables*;
RUN; /* 지정된 변수 값에 따라 분할된 그룹별로 적용 */

— 일반적으로 동일

BY [DESCENDING] *variable* ... ;

variable 은 순서화의 기준이 되는 변수의 이름
DESCENDING 지정하면 내림차순으로 순서화 (디폴트는 오름차순)
[참고] 기타 사항은 DATA 단계에서의 BY 문장 참고

[참고] 숫자변수의 오름차순의 위계는,
. (결측값) → 음수 → 0 → 양수
문자변수의 오름차순의 위계는,
␣(빈칸) ! " # $ % & ' ( ) * + , - . / 0 1 2 3 4 5 6 7 8 9 : ; < = > ? @
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z [₩] ^ _ `
a b c d e f g h i j k l m n o p q r s t u v w x y z { | } ~ 자음 모음

(예) DATA accident;

INPUT country $ city $ month number;

DATALINES;

Korea France Korea France Korea Korea Korea France Korea Korea France

Seoul Paris Seoul Paris Busan Seoul Busan Paris Seoul Busan Paris

1 2 2 3 3 4 4 4 5 5 5

5024 1200 4214 2354 1347 6635 987 3308 3375 334 893

RUN;

PROC PRINT DATA=accident;

RUN; * <---------- 인쇄절차 <1>;

PROC SORT DATA=accident;

BY country city; /* country 변수로 정렬한 후 그 안에서 다시 city 변수로 정렬 */

RUN;

PROC PRINT DATA=accident;

BY country; * <---------- 인쇄절차 <2>;

RUN;

PROC PRINT DATA=accident;

BY country city;

RUN; * <---------- 인쇄절차 <3>;

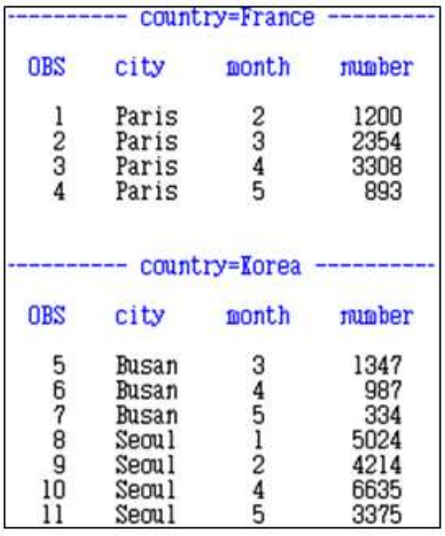
인쇄절차 <1>

⇒

OBS	country	city	month	number
1	Korea	Seoul	1	5024
2	France	Paris	2	1200
3	Korea	Seoul	2	4214
4	France	Paris	3	2354
5	Korea	Busan	3	1347
6	Korea	Seoul	4	6635
7	Korea	Busan	4	987
8	France	Paris	4	3308
9	Korea	Seoul	5	3375
10	Korea	Busan	5	334
11	France	Paris	5	893

인쇄절차 <2>

⇒



---------- country=France ----------

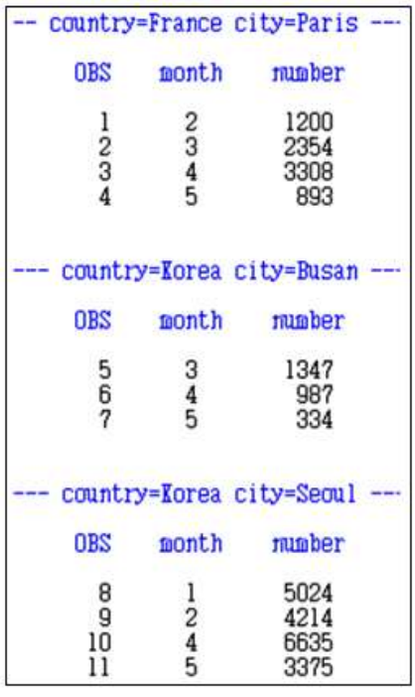
OBS	city	month	number
1	Paris	2	1200
2	Paris	3	2354
3	Paris	4	3308
4	Paris	5	893

---------- country=Korea ----------

OBS	city	month	number
5	Busan	3	1347
6	Busan	4	987
7	Busan	5	334
8	Seoul	1	5024
9	Seoul	2	4214
10	Seoul	4	6635
11	Seoul	5	3375

인쇄절차 <3>

⇒



-- country=France city=Paris ---

OBS	month	number
1	2	1200
2	3	2354
3	4	3308
4	5	893

--- country=Korea city=Busan ---

OBS	month	number
5	3	1347
6	4	987
7	5	334

--- country=Korea city=Seoul ---

OBS	month	number
8	1	5024
9	2	4214
10	4	6635
11	5	3375

[참고] PROC PRINT DATA=accident;

BY city;

RUN;

⇨ 이 경우, BY 변수로 city 하나만을 지정하는 것은 오류

(∵ city는 순서화의 최우선 기준 변수가 아니기 때문)

(PROC SORT DATA=accident; BY city; RUN; 선행 수행 필요)

(예) DATA accident;

INPUT country $ city $ month number;

DATALINES;

```
Korea   Seoul   1   5024
France  Paris   2   1200
Korea   Seoul   2   4214
France  Paris   3   2354
Korea   Busan   3   1347
Korea   Seoul   4   6635
Korea   Busan   4    987
France  Paris   4   3308
Korea   Seoul   5   3375
Korea   Busan   5    334
France  Paris   5    893
```

RUN;

PROC SORT DATA=accident OUT=sorted_acc;

/* 자료 accident를 정렬하여 sorted_acc 라는 이름으로 저장 */

BY country DESCENDING city;

/* country 변수에 대해서는 오름차순(ascending),
city 변수에 대해서는 내림차순(descending)으로 정렬 */

RUN;

PROC PRINT DATA=sorted_acc;

RUN;

⇒

OBS	country	city	month	number
1	France	Paris	2	1200
2	France	Paris	3	2354
3	France	Paris	4	3308
4	France	Paris	5	893
5	Korea	Seoul	1	5024
6	Korea	Seoul	2	4214
7	Korea	Seoul	4	6635
8	Korea	Seoul	5	3375
9	Korea	Busan	3	1347
10	Korea	Busan	4	987
11	Korea	Busan	5	334

<자료 sorted_acc>

6.4 CLASS 문장 <<<<<------- Skip

분류변수(classification variable)를 지정할 때 사용한다. CLASS 또는 CLASSES 를 사용해도 된다. 주로 분산분석 절차들에서 사용되며 MODEL 문장과 함께 사용되는 경우에는 언제나 CLASS 문장이 MODEL 문장보다 앞서 나와야 한다.

CLASS *variables* ;

분류변수란 관심있는 데이터 값을 얻을 때 범주(category) 혹은 조건에 해당하는 변수들

(예) 도시에 따라 자동차 사고 횟수에 유의적인 차이가 존재하는지 통계적으로 검증하기 위한 분산분석(ANOVA) 절차를 호출하여 분석하는 예제이다.

```
PROC ANOVA DATA=accident;  /* 자료 accident 에는 country, city, month
                              분류변수가 포함되어 있다 */
   CLASS city;  /* 분석에 필요한 분류변수만 지정 */
                /* MODEL 문장보다 앞서 나와야 함 */
   MODEL number=city;
RUN;
```

6.5 ID 문장

SAS 자료에 수록된 관측에는 내부적으로 관측 번호가 자동적으로 부여되기 때문에 관측번호를 이용하여 관측들을 구분할 수 있다. ID 문장은 내부적으로 결정된 관측번호를 무시하고 **사용자가 지정하는 변수 값으로 관측을 구분하려는 목적에서 사용**된다.

[참고] PRINT 절차(procedure)를 비롯한 대부분의 SAS 절차에서 수록된 관측 전부 또는 일부가 출력되는 경우에 출력 결과를 보면 Obs(observation)라는 변수 이름 밑에 관측 번호가 자동적으로 붙어 나온다. Obs 의 값은 1부터 순차적으로 SAS 자료에 수록된 순서대로 각 관측에 붙는데 이 **Obs 변수의 출력 대신** 다른 변수로 대체하고자 할때 **ID 문장**을 사용한다. 즉, ID(Identification)는 각 **관측 확인용 변수 이름을 지정**할 때 사용한다.

ID *variables* ;

ID 문장을 사용하면 변수 Obs는 출력되지 않음

(예) PROC PRINT DATA=accident;
ID country;
* VAR _ALL_; /* ID 문장에서 지정된 변수 이름이 VAR 문장에 또 다시 포함되면 그 변수는 두 번 출력 */
RUN;

⇒

country	city	month	number
France	Paris	2	1200
France	Paris	3	2354
France	Paris	4	3308
France	Paris	5	893
Korea	Busan	3	1347
Korea	Busan	4	987
Korea	Busan	5	334
Korea	Seoul	1	5024
Korea	Seoul	2	4214
Korea	Seoul	4	6635
Korea	Seoul	5	3375

6.6 LABEL 문장

PROC 단계에서 지정하는 LABEL 문장은 DATA 단계에서와 마찬가지 기능을 갖는다. 즉, 변수가 출력될 때 변수 이름 대신 변수에 대응하는 label 이 출력된다. 단, **DATA 단계에서 LABEL 문장**으로 label을 붙인 것은 데이터셋에 영구적으로 label 이 저장되지만, **PROC 단계에서의 label 은 일시적으로 출력을 해줄 뿐**이라는 차이가 있다.
LABEL 문장은 어떠한 SAS 절차에서도 사용 가능하며 사용 형식도 동일하다.

LABEL *variable='label'* ... ;

label 은 **256자** 까지 문자열 사용 가능(한글 128자), 홀따옴표 또는 겹따옴표 사용 가능
LABEL 문장에 지정한 변수들이 출력될 때는 원래 변수 이름 대신 대응되는 레이블이 출력됨

(예) PROC PRINT DATA=accident LABEL; /* PRINT 절차에서 변수명 대신 레이블로 출력하려면 LABEL 옵션 추가해야 */
ID country;
VAR city number;
LABEL number="# of Car Accidents" country='Country';
LABEL city="Big City";
/* LABEL 문장은 개수 무관하며 하나의 LABEL 문장에 변수와 레이블을 여러 개 지정하는 것도 가능 */
RUN;

⇒

Country	Big City	# of Car Accidents
France	Paris	1200
France	Paris	2354
France	Paris	3308
France	Paris	893
Korea	Busan	1347
Korea	Busan	987
Korea	Busan	334
Korea	Seoul	5024
Korea	Seoul	4214
Korea	Seoul	6635
Korea	Seoul	3375

(예) PROC PRINT DATA=accident SPLIT='*';
/* LABEL 출력 시 SPLIT에서 지정한 *를 기준으로 line 분리 */
/* SPLIT 이 사용되면 PRINT 절차에서 LABEL 옵션 생략 가능 */
ID country;

```
VAR city number;
LABEL number="# of * Car * Accidents" country='Country';
LABEL city="Big City";
RUN;
```

⇒

Country	Big City	# of Car Accidents
Korea	Seoul	5024
France	Paris	1200
Korea	Seoul	4214
France	Paris	2354
Korea	Busan	1347
Korea	Seoul	6635
Korea	Busan	987
France	Paris	3308
Korea	Seoul	3375
Korea	Busan	334
France	Paris	893

6.7 MODEL 문장 <<<<<------- Skip

모형(model)을 가정하여 자료분석하는 고급 통계 절차에서 설정된 **모형을 지정**할 때 사용한다. MODEL 문장의 사용 형식은 각 통계 절차마다 다소 차이가 있으나 일반적으로 다음 형태를 가진다.

MODEL *dependents*=*independents* / [*options*];

dependents 는 모형의 종속변수(dependent variable)/반응변수
independents 는 모형의 독립변수(independent variable)/설명변수
options 는 사용자 선택사항으로, 각 절차마다 고유한 지정 방법

구체적인 용법은 각 절차(procedure) 참고

(예) 회귀분석에서

```
PROC REG;
        MODEL y = x1 x2;
RUN;
QUIT;
```

(예) 분산분석에서

```
PROC ANOVA DATA=accident;
        CLASS city;
        MODEL number=city;
RUN;
QUIT;
```

6.8 OUTPUT 문장 <<<<<------- Skip

OUTPUT 문장은 여러 SAS 절차에서 사용 가능하다. 이 문장이 PROC 단계에 포함되면 호출된 SAS **절차에서 산출된 통계량들이 SAS 자료로 출력**된다. 사용 형식은 SAS 절차마다 차이가 많으나 일반적으로 다음과 같은 형식으로 쓰인다.

```
OUTPUT [OUT=SASdataset] [keyword=name] ... ;
```

SASdataset 은 SAS 절차에서 산출된 통계량들이 출력될 SAS 자료 이름
OUT=*SASdataset* 지정하지 않으면 시스템 내부에서 자료 이름 자동으로 부여
keyword 는 SAS 절차에서 산출되는 통계량에 대한 SAS 고유의 키워드
name 은 SAS 고유의 *keryword*를 대신하여 사용자가 지정한 새 이름

*keyword*를 새로운 이름으로 바꾸지 않고 그대로 사용하려면 *name*을 *keyword* 와 동일하게 지정

(예)
```
PROC MEANS DATA=accident;
   VAR number;
   OUTPUT OUT=auto MEAN=average STD=std;
        /* MEAN 은 평균, STD 는 표준편차 구하는 SAS 키워드 */
RUN;
```

⇒

MEANS 프로시저

분석 변수 : number

N	평균값	표준편차	최소값	최대값
11	2697.36	2000.01	334.0000000	6635.00

```
PROC PRINT DATA=auto;
RUN;
```

⇒

OBS	_TYPE_	_FREQ_	average	std
1	0	11	2697.36	2000.01

[참고] _TYPE_ 과 _FREQ_ 는 자동변수들로,
TYPE 은 OUTPUT 문장에 의하여 결과되는 SAS 자료의 유형,
FREQ 는 OUTPUT 문장에 의해 새로운 변수로 출력된 keyword

(예)
```
PROC REG;
   MODEL y=x;
   OUTPUT OUT=out1    .....      ;
```

6.9 QUIT 문장

대화식 절차(interactive procedure)들을 끝낼 때 사용한다.

QUIT;

SAS 절차(procedure) 중 **PLOT, ANOVA, GLM, REG** 등은 대화식 절차로, 이 절차들은 일단 호출되면 RUN 문장을 만나도 절차가 끝나지 않음 (RUN 문장을 만나면 그때까지 입력된 문장들을 실행하고 다시 새로운 지침의 입력을 대기하는 상태가 됨)
그러므로 대화식 절차에서 완전히 벗어나려면 QUIT 문장 필요

(예) PROC *procedure*;
...
... (SAS 문장들)
RUN;
QUIT;

6.10 WHERE 문장

특별한 조건을 만족하는 관측들만 골라 PROC 단계를 실행하고자 할 때 사용한다.

WHERE *whereexpression*;

whereexpression 은 일반 수식과 마찬가지로 연산자와 피연산자로 구성
(산술연산자(+, -, *, /), 비교연산자(=, ^=, <, <=, >, >=),
논리연산자(&, |, ^))
IN 같은 특수 비교연산자 사용도 가능

(예) PROC PRINT DATA=accident;
WHERE city="Seoul"; /* city 가 Seoul 인 관측만 출력 */
RUN; /* IF 문장 사용 불가 (DATA 단계에서만 사용 가능) */

⇒

OBS	country	city	month	number
1	Korea	Seoul	1	5024
3	Korea	Seoul	2	4214
6	Korea	Seoul	4	6635
9	Korea	Seoul	5	3375

(예) PROC PRINT DATA=accident;
WHERE month **IN** (1, 3, 5); /* month 가 1, 3, 5 인 관측만 출력 */
RUN; /* IF 문장 사용 불가 (DATA 단계에서만 사용 가능) */

⇒

OBS	country	city	month	number
1	Korea	Seoul	1	5024
4	France	Paris	3	2354
5	Korea	Busan	3	1347
9	Korea	Seoul	5	3375
10	Korea	Busan	5	334
11	France	Paris	5	893

(예) PROC PRINT DATA=accident;
WHERE number>3000 **&** city^="Busan";
/* number 가 3,000 이상이고 city 가 Busan 아닌 관측만 출력 */
RUN;

⇒

OBS	country	city	month	number
1	Korea	Seoul	1	5024
3	Korea	Seoul	2	4214
6	Korea	Seoul	4	6635
8	France	Paris	4	3308
9	Korea	Seoul	5	3375

(예) PROC PRINT LABEL;
VAR jobcode salary;
LABEL salary='annual salary';
WHERE jobcode='pilot2';
/* jobcode 값이 pilot2 인 것만 출력 */
RUN;

(예) WHERE lastname IN ('Smith', 'Brown');

(예) WHERE month IN (1, 2, 5);

(예) **WHERE** lastname='Smith' **AND** Salary NOT GE 50000;

(예) **PROC MEANS;**
VAR salary;
WHERE lastname='Smith';
WHERE ALSO month=1;
/* WHERE 문장 2개는 안됨 (WHERE ALSO 문장 사용) */
RUN;
⇔ PROC MEANS;
VAR salary;
WHERE lastname='Smith' **AND** month=1;
RUN;