Presentazione

Il Sistema SAS è probabilmente il software per condurre analisi statistiche più famoso al mondo. Questa popolarità è dovuta alla sua enorme varietà e flessibilità, sia in termini di programmazione che in termini di procedure già implementate.
Come sarà già noto ad alcuni, SAS contiene una vasta libreria di procedure statistiche (o PROC) usate per condurre analisi che vanno dalle semplici statistiche descrittive alle più complesse tecniche di analisi multivariata.
Ciò che è meno conosciuto è il fatto che SAS integra un linguaggio di programmazione di alto livello; virtualmente, qualsiasi procedura statistica può essere programmata con questo linguaggio.

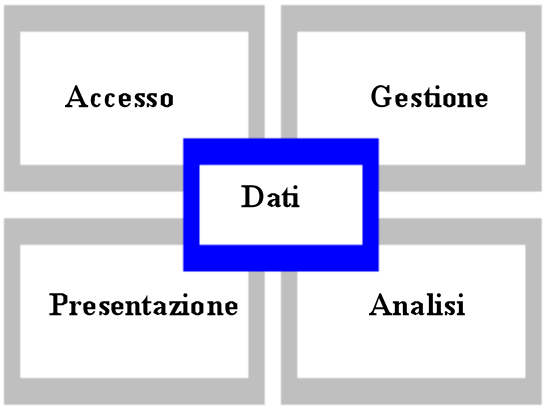
Scopo del seguente corso è fornire gli elementi di base per comprendere le potenzialità del sistema (pacchetto statistico e linguaggio di programmazione.

1. Introduzione

1.1 Uno sguardo al Sistema SAS.

SAS è un sistema software che integra diverse applicazioni; un'applicazione è ciò che trasforma dei semplici dati in qualcosa di più utile ed informativo.
Per ottenere tale trasformazione, sui dati a nostra disposizione devono essere eseguiti quattro passaggi fondamentali:

- accesso;
- gestione e organizzazione;
- analisi;
- presentazione.



Il primo di questi passaggi è l'accesso ai dati, i quali possono presentarsi ed essere memorizzati in forme diverse come:

- dati su supporto cartaceo da inserire ex-novo;
- dati memorizzati in un database esterno;
- dataset già presente nel sistema.

Una volta avuto accesso ai dati, è necessario sapere come gestirli e organizzarli, in modo da configurarli ai nostri particolari bisogni, come ad esempio:

- selezionare solo specifici campi o record da leggere;
- ordinare i record secondo determinati criteri;
- creare nuove variabili;
- ridefinire valori già esistenti.

Al fine di ottenere informazioni utili e informative, i dati precedentemente organizzati devono essere analizzati. Gli strumenti di analisi del sistema SAS spaziano,come abbiamo già visto, dalle più semplici statistiche descrittive a procedure più avanzate o finalizzate all'applicazio-ne in diversi settori come l'economia, le scienze sociali, la matematica o la medicina.

Dopo l’analisi, è necessario presentare i dati in un formato chiaro e comprensibile; anche in quest’ultimo passaggio SAS fornisce un’ampia scelta di applicazioni, che possono variare da un semplice elenco o tabella a grafici multidimensionali a colori in alta risoluzione.

1.2 Le finestre SAS: PROGRAM EDITOR, LOG, OUTPUT.

Tutti i programmi SAS devono essere scritti nel PROGRAM EDITOR, che permette, come un qualsiasi word processor, di copiare, tagliare, cancellare e muovere linee di codice, cercare e sostituire termini, salvare il lavoro fatto.

Una volta finito, il programma dovrà essere ‘lanciato’: i risultati di questa operazione saranno visibili in due differenti finestre: LOG e OUTPUT.

Il LOG contiene le informazioni su come il sistema ha elaborato il nostro programma; qui vengono segnalati, fra le altre cose, gli errori commessi, le note al programma e il tempo di elaborazione.

Nell’OUTPUT ovviamente compariranno i risultati dell’elaborazione. Se la finestra rimane vuota significa che qualcosa nel programma non ha funzionato; questo non vuol dire che ottenere un OUTPUT implichi automaticamente ottenere ciò che si voleva.

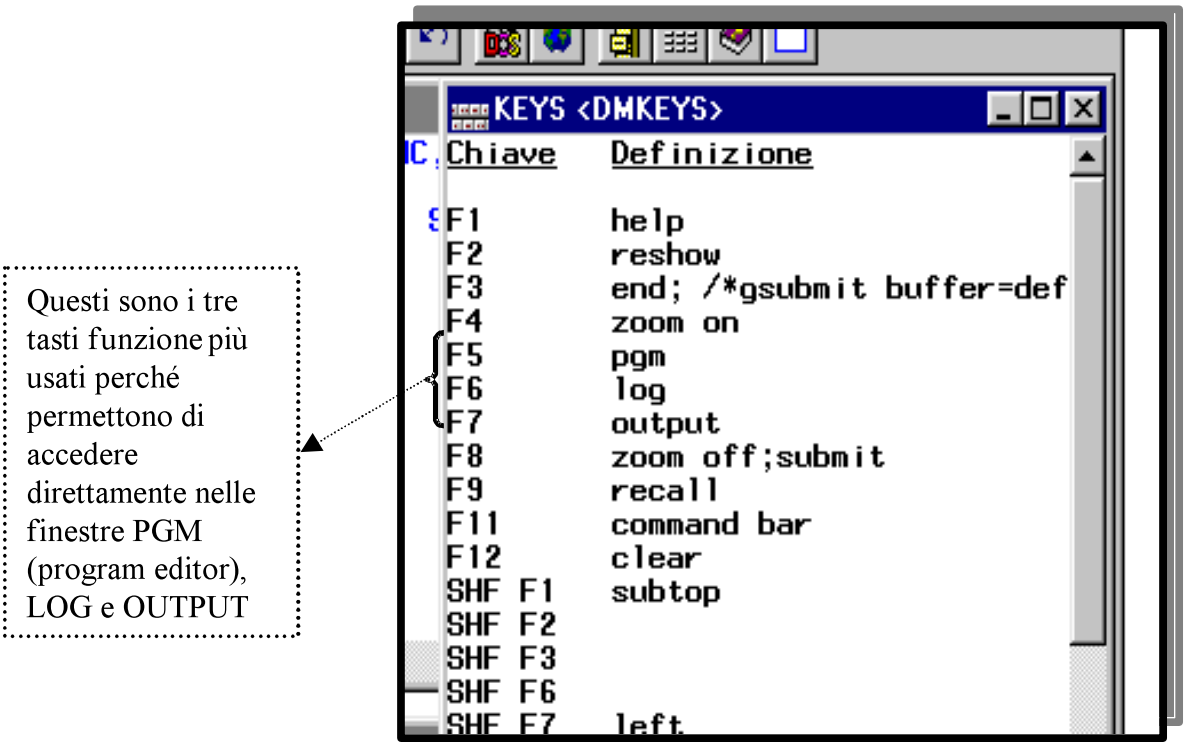
Solo dopo un’attenta analisi sia dei messaggi del LOG che dei risultati possiamo avere la certezza di non aver commesso errori.

Per accedere alle varie finestre è necesserio selezionarle dall'elenco delle finestre nel menù 'Finestre', oppure utilizzando uno dei tasti chiave, come vedremo nel seguente paragrafo.

1.3 I tasti chiave.

Nel sistema SAS è possibile attribuire ai tasti funzione o a combinazioni particolari di tasti dei comandi specifici. Generalmente vengono associate le funzioni che si usano più frequentemente, come ad esempio passare dalla finestra PROGRAM EDITOR a quella di OUTPUT, cancellare il contenuto di una finestra (CLEAR), richiamare un programma lanciato in precedenza (RECALL). L'elenco dei tasti chiave è disponibile nel menù 'Aiuto | Tasti Funzionali' ed è modificabile secondo le proprie esigenze.

FIGURA 1.1. ELENCO, MODIFICABILE, DEI TASTI CHIAVE.



2. I Fondamenti (A)

Come visto nell'introduzione, SAS è un sistema software che permette di leggere e analizzare dati in modo completo. A questo scopo è necessario conoscere e capire i comandi e le parole chiave che compongono i programmi SAS.
In questa sezione si imparerà a scrivere un programma che legga e stampi su video un *set* di dati.

2.1 DATA STEPS e PROC STEPS.

Il SAS è organizzato in STEPS (passaggi), che possono essere considerati come i paragrafi del capitolo di un libro. Esistono due differenti tipi di STEPS:

- DATA STEPS → gruppo di comandi in cui i dati vengono letti, manipolati e organizzati (è il passaggio in cui SAS è usato come un linguaggio di programmazione);

- PROC STEPS → procedure già implementate che agiscono sui dati [es.: procedure di ordinamento, analisi, stampa].

Ogni STEP deve iniziare con DATA o con PROC, in modo che il sistema riconosca il paragrafo che si troverà di fronte; tuttavia non vi è un ordine preciso per questi STEPS; si può iniziare un programma con un DATA STEP, seguito da due PROC, quindi ancora da un DATA, e così via.

Gli STEPS sono formati da comandi, che, riprendendo l'esempio del capitolo di un libro, sono come le frasi di un paragrafo.

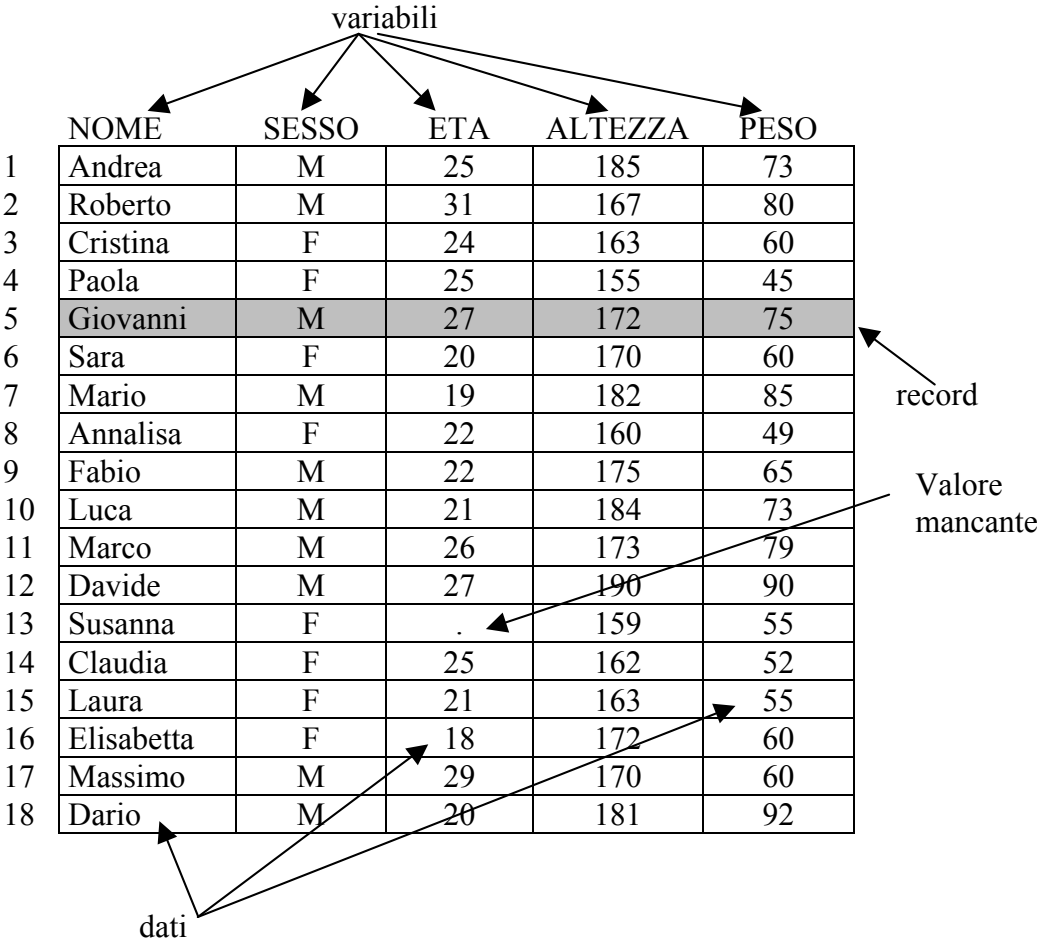
Un punto e virgola [;] segna la fine di un comando, così come il punto la fine di una frase.

I comandi sono *parole chiave* che hanno un preciso significato per il sistema e la loro sintassi richiama la lingua inglese, oppure sono altre parole che il programmatore aggiunge, come i nomi assegnati alle variabili.

2.2 DATA STEP

La Tabella 2.1 mostra un tipico *set* di dati SAS e si riferisce alla descrizione dei partecipanti ad uno studio sulla salute della popolazione.

Tabella 2.1 Set tipo di dati SAS.



	NOME	SESSO	ETA	ALTEZZA	PESO
1	Andrea	M	25	185	73
2	Roberto	M	31	167	80
3	Cristina	F	24	163	60
4	Paola	F	25	155	45
5	Giovanni	M	27	172	75
6	Sara	F	20	170	60
7	Mario	M	19	182	85
8	Annalisa	F	22	160	49
9	Fabio	M	22	175	65
10	Luca	M	21	184	73
11	Marco	M	26	173	79
12	Davide	M	27	190	90
13	Susanna	F	.	159	55
14	Claudia	F	25	162	52
15	Laura	F	21	163	55
16	Elisabetta	F	18	172	60
17	Massimo	M	29	170	60
18	Dario	M	20	181	92

DATO → singola unità di informazione, come l'altezza di una persona o il suo nome.

VARIABILE (o CAMPO) → set di dati che descrivono una specifica caratteristica, per esempio le altezze di tutti gli individui del gruppo. Le variabili possono essere classificate come numeriche o alfanumeriche (combinazioni di lettere, numeri o altri caratteri speciali).
SAS riconosce una variabile dal nome che le si assegna; questo può essere costituito al massimo da 8 caratteri e deve iniziare con una lettera o un trattino [_ (*underscore*)]. Non è possibile usare caratteri speciali come accenti e apostrofi (ETÀ→ETA)

RECORD → set di dati associati alla stessa unità di osservazione. La tabella 2.1 contiene ad esempio 18 record; ogni record è una riga di informazioni che contiene il nome, il sesso, l'altezza e il peso di ogni persona.

VALORI MANCANTI → informano il sistema che il dato non è disponibile. Sono rappresentati da un punto o da uno spazio bianco [questo dipenderà, come si vedrà più avanti, dalla forma con cui i dati saranno inseriti].

Dopo aver definito gli elementi di un dataset, vediamo in che modo inserirli nel computer.

- *Il comando* ***DATA***

Il comando DATA dichiara il nome che vogliamo assegnare all'intero dataset (valgono le stesse regole viste per nominare le variabili).

- *Il comando* ***INPUT***

Il comando INPUT dichiara i nomi che abbiamo scelto per le variabili del dataset.

- *Il comando* ***CARDS***

CARDS segnala che inizieranno le linee dei dati; nella riga successiva l'ultima linea di dati deve essere inserito un **;** .

- *Il comando* ***RUN***

RUN è l'istruzione che indica la fine di sia di un DATA STEP che di una PROC STEP.

Forma generale di un DATA STEP elementare:

```
data  nome dataset  ;
input nomi variabili ;
cards;
linee di dati
;
run;
```

2.3 PROC PRINT

Questa procedura dichiara al sistema di stampare (su video) le variabili del dataset.
La parola chiave VAR indica quali variabili stampare e in che ordine; se omessa SAS stamperà tutte le variabili contenute nel dataset, nell'ordine in cui sono state lette.

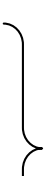
Forma generale di una PROC PRINT:

```
proc print data=  nome dataset  ;
var nomi variabili ;
run;
```

Esempio 2.1:

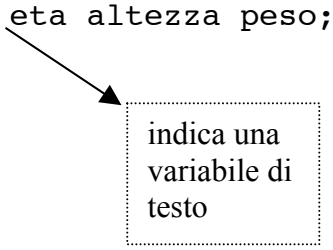
Riprendendo i dati della tabella 2.1, vogliamo far leggere al sistema le prime 3 osservazioni, per poi vederle stampate su video; il programma SAS per ottenere ciò è così strutturato:

```
data uno;
```



```
input nome $ sesso $ eta altezza peso;
cards;
Andrea M 25 185 90
Roberto M 31 167 65
Cristina F 24 163 55
;
run;

proc print data=uno;
var nome sesso altezza peso eta;
run;
```



DATA STEP

PROC

2.4 Il comando INPUT

Il comando INPUT nomina le variabili del dataset e indica a SAS dove i valori delle variabili possono essere trovati.
Questo comando può assumere varie forme, al fine di poter leggere dati di diverso formato (date, caratteri, numeri in codice binario, ecc..).
In questa sezione vedremo due diverse modalità per far leggere al sistema dati in formato numerico o carattere.

- Formato in lista.

Nell'esempio 2.1 i dati sono stati letti uno dopo l'altro, come in una lista; i valori sono separati da uno o più spazi bianchi.

SAS assocerà al primo gruppo di lettere o numeri il nome della prima variabile dichiarata nell'istruzione INPUT, il seguente gruppo di lettere o numeri alla variabile seguente, e così via.

Gli spazi bianchi sono quindi gli indicatori della fine di una variabile.

Per designare una variabile di tipo carattere si pone un segno di dollaro [**$**], preceduto e seguito da uno spazio, dopo il nome della variabile.

Utilizzando questo tipo di formato possiamo lavorare con variabili lunghe al massimo 8 caratteri.

La presenza di un valore mancante per qualche variabile deve essere indicata con un punto, altrimenti SAS assegnerà erroneamente alla variabile il valore seguente, che in realtà corrisponde ad un'altra variabile.

- Formato in colonna.

Per utilizzare questo formato, il contenuto di una variabile deve essere posto sulla stessa colonna per ogni linea di dati.

Dopo il nome della variabile deve essere specificato la posizione delle colonne (al massimo 80) in cui potrà essere trovata; se si tratta di una variabile carattere, il segno $ deve essere posizionato prima del numero di colonna.

Se i dati corrispondenti ad una medesima osservazione sono disposti su più linee di dati (perché superano l'80^{a} colonna o per una maggior comodità nell'inserimento dei dati), basterà indicare al sistema quale sia la prima variabile della riga successiva, facendola precedere dal simbolo # seguito dal numero di riga.

Questo secondo metodo, sicuramente più scomodo e macchinoso, è utile perché ci permette di saltare alcune colonne per omettere dei dati, di leggere le variabili in qualsiasi ordine, di leggere solo una parte di una variabile e di poter avere spazi bianchi all'interno della stessa.

Esempio 2.2:

Abbiamo ottenuto altre informazioni sui tre soggetti dell'esempio 2.1: giorno, mese e anno di nascita, sport praticato.
Per comodità numeriamo le colonne:

```
----+----1----+----2----+----3----+----4----+
Andrea    12  Ott 72     Pesca
Roberto    5  Gen 67     Calcio
Cristina   6  Mag 73     Tennis
```

Con il formato colonna ci sono varie possibilità di far leggere a SAS questi dati:

```
input nome $ 1-8 gionasc  $ 11-12 mesenasc $ 15-18 annonasc $ 19-
20 sport  $ 26-31;

input sport $ 26-31 annonasc  $ 19-20 nome  $ 1-8;

input nome $ 1-8 datanasc  $ 11-20 sport  $ 26-31;
```

Come visto, è stato possibile definire una variabile (datanasc) contenente spazi bianchi e lunga più di 8 caratteri, cosa invece impossibile con il formato di lista.

2.5 Le librerie di dati

Una libreria di dati è una raccolta di uno o più dataset.
Le modalità in cui SAS memorizza i dati sono di due tipi: temporaneo e permanente.

I dataset temporanei sono quelli creati durante una sessione di lavoro e, se non archiviati, vengono cancellati quando questa si conclude.
La libreria denominata 'Work' raccoglie, di default, tutti questi dataset, e viene svuotata ad ogni chiusura del programma.

I dateset permanenti vengono invece archiviati e resi diponibili in successive sessioni.
Per poter archiviare un dataset, l'utente deve definire una libreria di riferimento, contenuta in una normale cartella Windows (la libreria, per comodità d'uso, può avere lo stesso nome della cartella).
Anche per i nomi delle librerie valgono le regole viste per i nomi delle variabili e dei dataset.

Esempio 2.3: come creare una libreria:

1) Selezionare l'icona 'librerie' dalla barra degli strumenti:

2) Selezionare 'Nuova libreria';
3) Definire il nome della libreria e la cartella Windows di riferimento;
4) Scegliere l'opzione 'Assegna automaticamente all'avvio';
5) Chiudere.

Suppongo di aver creato la libreria 'esempi'.

Richiamando il programma dell'esempio 2.1, è possibile archiviare il dataset 'uno' nella libreria 'esempi', facendo precedere al nome del dataset quello della libreria di riferimento:

```
data esempi.uno;
......
```

3. I Fondamenti (B)

SAS è un programma di gestione e trattamento dei dati molto potente, in cui è possibile, sulla base di alcuni criteri, modificare le variabili del nostro dataset o crearne di nuove.
Supponiamo ad esempio di avere informazioni sui consumi di diversi modelli di automobile espressi in miglia percorse per gallone di benzina: ci potrebbe interessare cambiare l'unità di misura in km. per litro, creare una nuova variabile qualitativa basata sui consumi di ogni auto (buono, medio, elevato), ordinare i dati secondo la variabile precedentemente creata, selezionare un dataset ridotto che consideri solo un determinato sottogruppo (ad esempio solo le macchine giapponesi).
Supponiamo inoltre che i dati originali risiedano in un file già esistente: come inserirli in un programma SAS?
In questa sezione impareremo a creare nuove variabili, ordinare i dati, leggerli da file esterni, lavorare con dataset selezionati rispetto quello di partenza.

3.1 Il comando IF come selezionatore

Il comando IF (usato come selezionatore) serve ad escludere alcune osservazioni dal dataset; SAS lavorerà solo su i record che rispondono ai criteri del comando IF.
Alcuni esempi:

```
if dim_auto = 'piccola';      - lavora solo sulle auto di piccole dimensioni –
if 11 <= kpl < 18;            - lavora solo sulle auto con consumi medi     –
if paese ne 'giappone';       - esclude dall'analisi le auto giapponesi –
if (paese = 'usa') and        - lavora solo sulle auto americane di grosse dimensioni –
   (dim_auto = 'grande');
```

il simbolo di disuguaglianza ≠ è sostituito in SAS da NE (not equal).

3.2 I comandi IF…THEN…ELSE

I comandi IF…THEN…ELSE vengono usati per creare nuove variabili, nel caso in cui il valore di queste dipende dal valore di una variabile già esistente.
IF e THEN vanno considerati insieme come un unico comando, mentre ELSE viene usato di seguito, in un'altra 'frase', per migliorare l'efficienza del programma.

Nelle seguenti righe di codice il comando ELSE non viene usato. SAS controlla, per ogni comando, se la condizione IF è verificata.
Assumiamo che il primo record sia un'auto con un consumo di 15 km. per litro (variabile kpl); SAS valuterà la prima condizione e la troverà falsa, quindi passerà alla seconda e, essendo vera, assegnerà alla nuova variabile 'consumi' il valore 'medio'; andrà poi alla condizione successiva, che naturalmente sarà falsa, e così via.

```
if 18 <= kpl        then consumi = 'buono  ';
if 11 <= kpl < 18   then consumi = 'medio  ';
if       kpl < 11   then consumi = 'elevato';
```

Per eseguire in un miglior e più efficiente modo la precedente operazione è utile usare il comando ELSE: a differenza di quanto visto prima SAS prosegue a valutare le condizioni fino a che vengono verificate, saltando poi le successive.

```
     if 18 <= kpl        then consumi = 'buono  ';
else if 11 <= kpl < 18   then consumi = 'medio  ';
else if       kpl < 11   then consumi = 'elevato';
```

3.3 Come usare i dati provenienti da file esterni

Le informazioni di molti dataset risiedono su file esterni al sistema; sarebbe quindi un inutile e faticoso lavoro doverli riscrivere all'interno di un programma SAS con il comando CARDS. Fortunatamente questo non è necessario, poiché è possibile specificare dove andare a recuperare il dataset di riferimento.
I due comandi che permettono di leggere i dati da un file esterno sono:
il comando FILENAME e il comando INFILE.

Il comando **FILENAME**

FILENAME viene usato per collegare al programma SAS un file esterno, assegnandogli un nome di riferimento utilizzabile, quando necessario, nei vari STEPS.
Il comando deve precedere il DATA STEP che legge i file.
Supponiamo che i record relativi ai vari modelli di automobili stiano su di un floppy disk *a:* o nella directory 'esempi' del disco rigido *c:* e che questo abbia nome 'auto.dat':
Per assegnare a questo file un nome di riferimento (es.'datain') la sintassi del comando è:

```
filename datain 'a:\auto.dat';
filename datain 'c:\esempi\auto.dat';
```

Il comando **INFILE**

Il comando INFILE è usato all'interno del DATA STEP, prima del comando INPUT, e informa il sistema che i dati da usare, provenendo da un file esterno, non saranno quelli che seguono il comando CARDS.

Esempio 3.1:

```
data auto;
infile datain;
input modello $ kpl dim_auto $ paese $;
```

Import Wizard

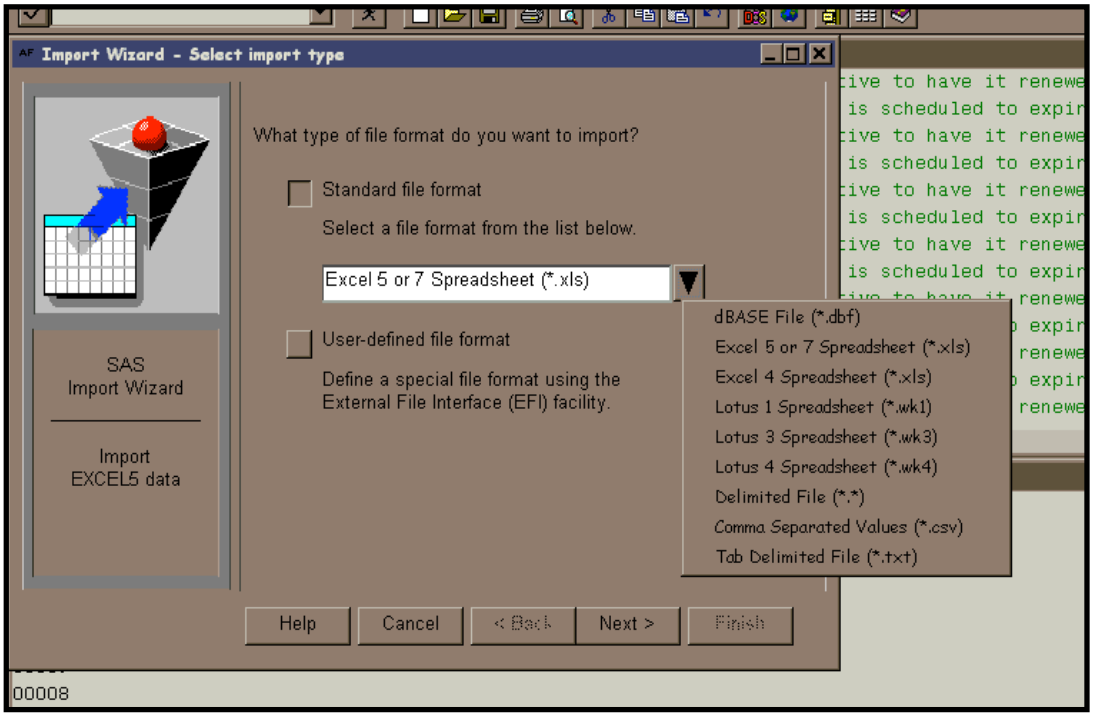
I dati dell'esempio 3.1 sono in semplice formato testo; capita spesso di dover importare dati in un formato specifico (DBF, XLS, CSV o altro). In questo caso il semplice INFILE non è sufficiente e bisognerebbe utilizzare procedure più lunghe e complicate; fortunatamente, dalla versione 6.12 (quella che noi utilizziamo) è stato aggiunto il l'"Import Wizard" (richiamato con il comando 'Import' del menù 'File'), un'applicazione che rende immediata e intuitiva l'importazione dei dati. Lo stesso discorso rimane valido per l'esportazione dei dati dal formato SAS ad altri formati.

Esempio 3.2:

Nella directory 'c:\epidemiologia\esempi' è stato salvato in formato Excel il dataset 'Nomi.xls', contenente i dati completi della tabella 2.1. Utilizzando l'Import Wizard, importiamo i dati in un dataset SAS.

1) Avviare l'Import Wizard;
2) Selezionare il formato (Excel 5 or 7 Spreadsheet);
3) Sfogliare le cartelle e selezionare quella che contiene il dataset;
4) Selezionare il file 'Nomi.xls';
5) Selezionare negli appositi campi la libreria di riferimento e il nome SAS del dataset;
6) Selezionare 'Finish'

FIGURA 3.1: IMPORT WIZARD: LA FASE DI SCELTA DEL FORMATO DA IMPORTARE



Importante: quando si vuole importare un file dal formato .xls, ci si assicuri che questo sia stato salvato come 'Cartella di Lavoro Microsoft Excel 5.0/95'.

3.4 PROC SORT

La procedura SORT serve a cambiare l'ordine dei record in un dataset; il criterio di ordinamento dipende dal comando BY.
Se volessimo ordinare i dati sui modelli di auto in base alla variabile kpl, dovremmo scrivere:

```
proc sort data = auto;
by kpl;
```

E' possibile disporre i record per più di un criterio; le osservazioni saranno ordinate in primo luogo per la prima variabile elencata, poi, all'interno di questa, l'ordine dipenderà dalla seconda variabile, e così via.
Per esempio, potrebbe interessarci vedere i dati ordinati, all'interno di ogni paese di produzione, per consumi:

Esempio 3.3:

```
proc sort data = auto;
by paese kpl;
```

Forma generale di una PROC SORT:

```
proc sort data = dataset;
by variabili < opzioni >;
```

Opzioni del comando BY: tutti i comandi SAS hanno delle opzioni facoltative che, fra le altre cose, permettono di cambiare i parametri di default. In questo caso l'ordinamento di partenza è sempre dal più piccolo al più grande o dalla A alla Z (ascendente); per poterlo invertire è necessario specificare DESCENDING prima del nome della variabile.

Esempio 3.4:

Abbiamo i dati di uno studio svolto su pazienti, provenienti da differenti cliniche, affetti da sclerodermia, una malattia della pelle caratterizzata da un ispessimento e da una elevata tensione dell'epidermide, con possibile coinvolgimento dei vasi sanguigni e degli organi interni. Parte di questi pazienti è stata trattata con un nuovo farmaco, parte con un placebo* (farmaco: 1=nuovo 2=placebo).
Prima e dopo il trattamento si sono misurati i seguenti parametri per ogni paziente: spessore della pelle (spess1 / spess2; maggiore il valore, peggiore la situazione), mobilità della pelle (mobi1 / mobi2; maggiore il valore, migliore la situazione), gravità delle condizioni generali del paziente (condiz1 / condiz2; maggiore il valore, peggiore la situazione).

Vogliamo vedere, per i pazienti dalla clinica 45 fino alla 50, in quali pazienti si sono riscontrati alcuni miglioramenti nella variabile condizione generale, raggruppando i soggetti secondo il farmaco assunto e quantificando il miglioramento.
I dati , in formato libero, sono nel file 'c:\epidemiologia\esempi\sclero.dat'.

Programma SAS:

```
filename datain 'c:\epidemiologia\esempi\sclero.dat';

data uno;
 infile datain;
 input clinica paziente farmaco spess1 spess2 mobi1 mobi2 condiz1
       condiz2 ;
 migliora = condiz1 - condiz2;
 if migliora > 0;
```

* Placebo: sostanza chimicamente inerte somministrata al posto di un farmaco. Un placebo può risultare efficace perché chi lo assume è convinto che avrà un effetto positivo.

```
 if 45 <= clinica <= 50;
run;


proc sort;
by farmaco;
run;

proc print;
by farmaco;
run;
```

OUTPUT 3.1

SAS System 8
15:17 Sunday, April 19, 1998

FARMACO=1

OBS	CLINICA	PAZIENTE	SPESS1	SPESS2	MOBI1	MOBI2	CONDIZ1	CONDIZ2	MIGLIORA
1	45	34	38	.	184	195	7	6	1
2	45	36	26	.	255	211	5	2	3
3	46	43	7	7	440	444	4	3	1
4	46	44	17	18	191	192	7	5	2
5	47	51	28	28	191	.	9	8	1
6	48	59	26	30	211	310	9	7	2
7	49	69	6	6	531	521	2	1	1
8	50	75	19	9	432	398	5	1	4

FARMACO=2

OBS	CLINICA	PAZIENTE	SPESS1	SPESS2	MOBI1	MOBI2	CONDIZ1	CONDIZ2	MIGLIORA
9	45	35	24	.	472	537	2	1	1
10	46	40	9	9	347	309	5	2	3
11	46	47	12	15	313	373	9	5	4
12	46	48	9	11	359	395	5	3	2
13	47	52	7	7	383	.	5	2	3
14	48	53	18	7	444	563	6	4	2
15	49	63	18	18	378	373	3	2	1
16	49	66	7	7	675	636	3	1	2
17	50	76	17	9	510	448	3	1	2

3.5 *Il comando SET*

In un unico programma SAS è possibile utilizzare più dataset, usandone ad esempio uno come base per crearne un altro.
Per fare riferimento ad un dataset viene usato il comando SET.

Esempio 3.5:

Sempre riferendosi ai dati dell'esempio 3.4, supponiamo di voler procedere ad un'ulteriore analisi per i pazienti della clinica 49. Potremmo creare un nuovo dataset contenente solo le osservazioni rilevate in quella clinica, utilizzando il comando IF come selezionatore.

Programma SAS:

```
data clinic49;
 set uno;
 if clinic = 49;

proc print data=clinic49;
run;
```

Nuovi comandi:

`set uno;` i dati utilizzati provengono dal dataset UNO, già esistente nel sistema, piuttosto che da un comando CARDS o da un file esterno. Essendo disponibili ora due dataset nello stesso programma, nelle procedure sarà necessario specificare quale dataset usare. Se non specificato SAS usa per default quello creato più di recente.

4. La gestione dei dati

Per piccoli set di dati, abbiamo visto come sia semplice creare un dataset SAS e applicare una delle procedure. Tuttavia, ci sono occasioni in cui si ha a che fare con una grossa quantità di dati, magari non organizzati in modo da poter essere immediatamente analizzati.
SAS possiede una notevole quantità di utili tecniche di gestione e trattamento di dati e dataset, e alcune verranno illustrate in questa lezione.
Sarebbe impossibile descrivere tutti i tools di trattamento dei dati che SAS mette a disposizione, e comunque questo non rientra nelle finalità di questo corso; lo scopo è invece dare delle indicazioni sulle potenzialità del sistema, in modo che ognuno, se interessato ad approfondire determinati aspetti, sia in possesso degli elementi per consultare e capire i manuali SAS.

4.1 La funzione LAG

SAS tratta i dati un record alla volta; tuttavia potrebbe essere utile usare un valore preso dall'osservazione precedente. La funzione LAG assegna all'osservazione corrente una nuova variabile il cui valore è quello di una determinata variabile del record precedente.
La forma della funzione è:

```
nuova variabile = lag(variabile)  ;
```

Supponiamo di avere un dataset con le temperature massime giorno per giorno: vogliamo calcolare la relativa differenza giornaliera.

```
jan1 2
jan2 7
jan3 5
```

Per poter saper la differenza dal primo al due di gennaio, è necessario che entrambe le informazioni siano disponibili sullo stesso record: la seguente istruzione aggiunge un nuova variabile PRE_TEMP alle variabili GIORNO e TEMP.

```
pre_temp = lag(temp);
```

Nuovo dataset:

```
jan 1 2 .
jan 2 7 2
jan 3 5 7
```

Per calcolare la differenza:

```
diff = temp - pre_temp;
```

4.2 I comandi DROP e KEEP

Questi due comandi possono essere usati per escludere determinate variabili dal dataset SAS o per essere sicuri che ne siano incluse solo alcune. Se il numero delle variabili da escludere è maggiore di quelle da tenere, si usa il comando KEEP; al contrario si usa il comando DROP.
La loro forma è:

```
DROP lista di variabili;
KEEP lista di variabili;
```

Perché non si dovrebbero utilizzare tutte le variabili del dataset?
Se lavoriamo con una grossa quantità di variabili, e sappiamo che alcune di esse non ci serviranno più nell'analisi dei dati, l'uso di uno dei due comandi può alleggerire il lavoro del sistema, con il vantaggio di ottenere i risultati in un tempo più breve.

4.3 Come unire i dataset.

Esistono diverse modalità per unire i dataset tra cui:

concatenazione;
unione a chiave;

Sebbene sia possibile unire due o più dataset, in questa sezione ci si concentrerà sull'unione di solo due dataset; poiché l'utilizzo di una delle due modalità sopraelencate porta a dataset finali diversi tra loro, gli esempi useranno sempre, come dataset di partenza, 'Anno1992' e 'Anno1993', con le temperature dei primi sei giorni di gennaio.

ANNO1992

giorno	temperatura
jan 1	2
jan 2	7
jan 3	5
jan 4	2
jan 5	0
jan 6	0

ANNO1993

giorno	temperatura
jan 1	3
jan 2	5
jan 3	2
jan 4	3
jan 5	0
jan 6	2

Concatenazione di dataset (SET).

Per concatenare i dataset, si usa il comando SET. Il numero totale di osservazioni del dataset finale equivale alla somma delle osservazioni dei due dataset che sono stati combinati; i record del primo dataset sono letti per primi, quindi quelli del secondo, e così via.
Il numero delle variabili nel dataset concatenato sarà uguale al numero delle differenti variabili dei due dataset di origine. Definiamo, in ANNO1992 e in ANNO1993, le variabili GIORNO e TEMP; le istruzioni per concatenare i due dataset sono:

```
data combined;
  set anno1992 anno1993;
```

COMBINED ha dodici osservazioni:

COMBINED

giorno	temp
jan 1	2
jan 2	7
jan 3	5
jan 4	2
jan 5	0
jan 6	0
jan 1	3
jan 2	5
jan 3	2
jan 4	3
jan 5	0
jan 6	2

Unione a chiave dei dataset (MERGE) e il comando RENAME.

Unire a chiave significa concatenare due o più dataset secondo determinati criteri.

Per unire i dataset , si usa il comando MERGE. Il numero di record del dataset risultante è pari al numero di record del dataset di partenza con più osservazioni.

Al comando MERGE si associa il comando BY; le osservazioni dei dataset di partenza sono unite secondo il valore assunto da una o più variabili comuni, specificate dopo BY.

I dataset di partenza **devono** essere stati in precedenza ordinati (SORT) per le variabili specificate dopo il BY.

Il dataset finale avrà il numero di record uguale alla somma delle diverse osservazioni relative alla variabile comune specificata in BY.
Se nei due dataset esistono altre variabili con lo stesso nome, i valori assegnati a quelle variabili saranno quelli dell'ultimo dataset elencato nel comando.

Per questo motivo è utile, per non perdere queste informazioni, rinominare con nomi differenti le variabili in esame.

Il comando RENAME specifica un nuovo nome da assegnare ad una variabile; la sua forma è:

```
RENAME vecchio nome = nuovo nome;
```

Assumendo che il dataset _1992 abbia le variabili GIORNO e TEMP92 (precedentemente chiamata TEMP e poi rinominata) e il dataset _1993 le variabili GIORNO e TEMP93 (precedentemente chiamata TEMP e poi rinominata), le istruzioni per unire, secondo la chiave GIORNO, i due dataset sono:

```
data merged;
 merge _1992 _1993;
 by giorno;
```

MERGED ha sei osservazioni e tre variabili:

	MERGED	
giorno	temp92	temp93
jan 1	2	3
jan 2	7	5
jan 3	5	2
jan 5	2	3
jan 6	0	0
jan 8	0	2

4.4 Le righe di commento

Includere in un programma SAS delle linee di commento è di grosso aiuto, specialmente quando il programma è lungo o quando ci si deve ricordare la funzione di determinate righe di codice.

Esistono due diversi modi di inserire commenti. Il primo è iniziare un'istruzione con un asterisco (*), e finirla con il solito punto e virgola; l'altro è iniziare il commento con /* (come se aprissimo una parentesi) e finirlo con */.

L'uso di questi simboli diviene utile anche quando, per determinate ragioni, vogliamo far eseguire solo una porzione del programma; la parte in quel momento superflua, piuttosto che essere cancellata, può venire messa 'tra parentesi'.

5. Le funzioni

Nel sistema SAS ci sono molte funzioni che permettono di lavorare sul valore dei dati. Alcune eseguono trasformazioni aritmetiche e trigonometriche; altre lavorano sulle variabili di testo ed altre ancora calcolano la probabilità da alcune differenti distribuzioni.
In questa breve sezione saranno descritte alcune delle innumerevoli funzioni disponibili in SAS. (per un elenco più esauriente si può fare riferimento ai manuali di SAS Base o all'help in linea del programma stesso).

5.1 Funzioni numeriche.

`abs (argomento)`	fornisce il valore assoluto di un argomento[1]
`exp (argomento)`	eleva *e* alla potenza indicata nell'argomento
`int (argomento)`	fornisce il valore intero dell'argomento
`log (argomento)`	fornisce il logaritmo naturale dell'argomento
`round(argomento, decimale)`	arrotonda il valore dell'argomento al decimale specificato. es: x=33.2217; y=round(x,.001) → y=33.222
`sqrt (argomento)`	fornisce la radice quadrata positiva dell'argomento

Esempio 5.1:

Da un semplice elenco di valori numerici, si vuole estrarre il valore assoluto, l'esponenziale, l'intero, il logaritmo naturale, l'arrotondamento alla seconda cifra decimale e la radice quadrata. Il programma SAS è così strutturato:

```
data funzioni;

input num;

assoluto=abs (num);
esponenz=exp (num);
intero  =int (num);
loga    =log (num);
arrot   =round(num,.01 );
radice  =sqrt(num);

cards;
10.229
-22.333
12.559
0.12
9.41
;

run;
```

le funzioni vanno inserite nel DATA STEP dopo il comando INPUT e prima del comando CARDS.

[1] l'argomento può essere sia un valore che il nome di una variabile.

```
proc print;
run;
```

5.2 Funzioni di testo.

`substr( argomento, posizione, n)`	Estrae una sottostringa dell'argomento, che inizia dal carattere nella posizione specificata e di lunghezza n.

Esempio 5.2:

Nella variabile carattere CLI_CIT, gli ultimi tre caratteri indicano la sede della clinica.
Vogliamo creare una nuova variabile CITY, che contenga solo quest'ultima informazione:

```
data citta;
input cli_cit $;
city=substr(cli_cit,3,2);
cards;
01mi
02mi
03mi
04pv
05pv
06pd
07pd
;
run;

proc print;
run;
```

OUTPUT 5.1:

SAS System 14
15:14 Sunday, April 26, 1998

OBS	CLI_CIT	CITY
1	01mi	mi
2	02mi	mi
3	03mi	mi
4	04pv	pv
5	05pv	pv
6	06pd	pd
7	07pd	pd

5.3 Funzioni di probabilità.

`poisson (m,n)`	fornisce P(X ≤ n), dove X è una variabile casuale distribuita come una Poisson e m ≥ 0 è la media.
`probbnml (p,n,m)`	fornisce P(X ≤ m), dove X è una variabile casuale distribuita come una binomiale, 0 ≤ p ≤ 1, e n è il numero di prove.
`probchi (x,df)`	fornisce P(X ≤ x), dove X è una variabile casuale distribuita come un chi-quadro, df è il numero dei gradi di libertà, e x ≥ 0.
`probf(x,ndf,ddf)`	fornisce P(X ≤ x), dove X è una variabile casuale distribuita come una F, ndf è il numero dei gradi di libertà al numeratore, ddf il numero di gradi di libertà al denominatore, e x ≥ 0.
`probhyper(N,K,n,x)`	fornisce P(X ≤ x), dove X è una variabile casuale distribuita come una ipergeometrica, N > 1 è la dimensione della popolazione, 0 ≤ K ≤ N è il numero di coloro che nella popolazione hanno una speciale caratteristica, 0 ≤ n ≤ N è la numerosità del campione, e max(0, K+n-N) ≤ x ≤ min(K,n).
`probnorm(x)`	fornisce P(X ≤ x), dove X è una variabile casuale distribuita come una normale con $\mu = 0$ e $\sigma = 1$.
`probt(x,df)`	fornisce P(X ≤ m), dove X è una variabile casuale distribuita come una t con df gradi di libertà.

Esempio 5.3:

Vogliamo verificare alcune probabilità binomiali, con n=5 e p=0.4.

```
data bino;

input m;
 n=5;
 p=0.4;
 cdf=probbnml(p,n,x);

cards;
0
1
2
3
4
5
;

run;

proc print;
 var m cdf;
run;
```

OUTPUT 5.2.

OBS	M	CDF
1	0	0. 07776
2	1	0. 33696
3	2	0. 68256
4	3	0. 91296
5	4	0. 98976
6	5	1. 00000

distribuzione cumulativa per una variabile casuale binomiale con n=5 e p=0.4

6. Statistiche descrittive.

Per descrivere dei dati è necessario calcolare medie, deviazioni standard e percentili, disegnare grafici e organizzare i dati in tabelle.
In SAS esistono diverse procedure pensate a questo scopo; alcune volte però un output potrebbe risultare difficile da leggere, specialmente perché i nomi assegnati alle variabili possono avere al massimo una lunghezza di otto caratteri. Per ovviare a tale inconveniente impareremo a rendere i risultati più comprensibili mediante l'utilizzo di etichette e titoli.
In questa lezione s'insegnerà l'uso della PROC UNIVARIATE e della PROC MEANS per il calcolo di statistiche descrittive.

6.1 I comandi LABEL e TITLE.

Il comando LABEL assegna un'etichetta ad una variabile, che apparirà nell'output accanto al nome originale. La lunghezza massima è di 40 caratteri; il comando è valido sia nel DATA STEP che nel PROC STEP.
Ecco un esempio:

```
label tempo  = 'tempo necessario a finire un esame'
      docente= 'nome del docente'
      voto   = 'punteggio ottenuto all''esame';
```

(quando, come in questo caso, c'è un apostrofo all'interno dell'etichetta, è necessario usare due apostrofi consecutivi).

Il comando TITLE assegna un titolo in testa ad ogni pagina di output; si possono creare più titoli in una stessa pagina numerandoli (TITLE1, TITLE2, …), o cambiarli all'interno del programma.

6.2 PROC UNIVARIATE

La PROC UNIVARIATE genera statistiche descrittive: media, deviazione standard, percentili o quantili (1%, 5%, 10%, 25%, 50%, 75%, 90%, 99%), valore minimo e massimo. Inoltre stampa i cinque valori più grandi e i cinque più piccoli del dataset, oltre ad altre statistiche qui non discusse. L'opzione NORMAL calcola una statistica che verifica se i dati possono considerarsi distribuiti secondo una normale.
Tramite il comando ID si specifica il nome della variabile che si vuole stampare vicino alla più piccola e alla più grande delle osservazioni (ad esempio il codice identificativo di un paziente); questa opzione può rendere più facile e veloce controllare a chi sono associati dei valori anomali o estremi.

Forma generale di una PROC UNIVARIATE:

```
proc univariate data = dataset < opzioni > ;
 var variabili;
```

Esempio 6.1:

Proseguendo l'esempio 3.1 sui dati di uno studio sulla sclerodermia, generiamo alcune statistiche descrittive. In particolare vogliamo:
- assegnare al nome di alcune variabili una descrizione più lunga;
- descrivere le diverse statistiche riguardanti il miglioramento delle condizioni generali nel gruppo che ha assunto il nuovo farmaco e in quello che ha assunto il placebo.

Programma SAS.

```
filename datain 'c:\epidemiologia\esempi\sclero.dat';

data one;
 infile datain;
 input clinica paziente farmaco spess1 spess2 mobi1 mobi2 condiz1
       condiz2 ;
 migliora = condiz1 - condiz2;

label
     farmaco  = 'farmaco (1) o placebo (2)'
     migliora = 'miglioramento delle condizioni'
;
run;

proc sort;
 by farmaco;
run;

proc univariate;
 by farmaco;
 var migliora;
 id paziente;
title 'Studio sulla Sclerodermia';
run;
```

L’Output 6.1 mostra il risultato relativo alla variabile ‘miglioramento delle condizioni’ nel gruppo che ha assunto il farmaco.
Nella prima parte dell’output troviamo il numero di osservazioni (N)=31, la media campionaria (Mean)=0.645, la deviazione standard campionaria (Std Dev)=2.026, la somma aritmetica dei valori delle osservazioni (Sum)=20, e la varianza campionaria (Variance)=4.103; quindi i quantili, la mediana (Med)=0, il range =9 e la moda (Mode)=0.
Vengono poi presentati i valori più bassi e quelli più alti del dataset: il paziente 42 ha avuto il miglioramento peggiore (-4), il paziente 15 il migliore (+5).
Le ultime tre righe ci informano che il numero dei valori mancanti è 1, pari al 3.13% dei dati.

Importante: se vogliamo generare statistiche separate per variabile (nel nostro caso 'farmaco') usiamo il comando BY, dopo esserci assicurati che il dataset sia stato ordinato (SORT) per quella variabile.

OUTPUT 6.1 OUTPUT SAS PER L'ESEMPIO 6.1

```
                Studio sulla Sclerodermia                              45
                                                 21:21 Sunday, March 29,1998

------------------------ farmaco (1) o placebo (2)=1 ------------------------

                            Univariate Procedure

Variable=MIGLIORA      miglioramento delle condizioni

                                 Moments

                N              31  Sum Wgts
                Mean     0.645161  Sum
                Std Dev  2.025642  Variance   4.1032
                Skewness 0.186157  Kurtosis   0.3589
                USS             136  CSS        123.0968
                CV       313.9745  Std Mean   0.363816
                T:Mean=0 1.773317  Pr>|T|       0.0863
                Num ^= 0        19  Num > 0          13
                M(Sign)        3.5  Pr>=|M|      0.1671
                Sgn Rank        42  Pr>=|S|      0.0919
                W:Normal 0.932542  Pr<W         0.0611


                             Quantiles(Def=5)

                100% Max         5       99%          5
                 75% Q3          2       95%          4
                 50% Med         0       90%          4
                 25% Q1          0       10%         -1
                  0% Min        -4        5%         -3
                                          1%         -4
                Range            9
                Q3-Q1            2
                Mode             0


                               Estremes

                  Lowest    ID       Highest    ID
                      -4(      42)         3(      36)
                      -3(      26)         4(       5)
                      -2(      50)         4(      29)
                      -1(      73)         4(      75)
                      -1(      30)         5(      15)


                     Valore missing        .
                     Count                 1
                     % Count/Nobs       3.13
```

indica che le statistiche presentate sono riferite ai pazienti che hanno assunto il farmaco

6.3 PROC MEANS

La PROC MEANS è la più semplice e immediata procedura per calcolare statistiche descrittive univariate di base. Per default la procedura genera la dimensione del campione, la media, la deviazione standard, i valori massimi e minimi.

Forma generale di una PROC MEANS:

```
proc means data = dataset < opzioni > ;
 by variabili;
 var variabili;
```

OUTPUT 6.2. RISULTATI DELLA PROC MEANS APPLICATA ALLA VARIABILE 'MIGLIORAMENTO DELLE CONDIZIONI' DELL'ESEMPIO PRECEDENTE:

```
                          Studio sulla Sclerodermia                    88
                                                  17:34 Sunday, May 3, 1998

          Analysis Variable : MIGLIORA miglioramento delle condizioni

------------------------- farmaco (1) o placebo (2)=1 ------------------------

           N           Mean        Std Dev        Minimum        Maximum
          ------------------------------------------------------------
          31      0.6451613      2.0256421     -4.0000000      5.0000000
          ------------------------------------------------------------

------------------------- farmaco (1) o placebo (2)=2 ------------------------

           N           Mean        Std Dev        Minimum        Maximum
          ------------------------------------------------------------
          44      0.5000000      1.6210820     -3.0000000      4.0000000
          ------------------------------------------------------------
```

7. Le tabelle di Frequenza.

Il modo più semplice e immediato per descrivere dei dati è organizzarli in tabelle.
In questa parte analizzeremo la procedura che genera tabelle (PROC FREQ); questo argomento conclude la parte introduttiva al Sistema SAS, introducendo le basi per la sezione successiva, riguardante l'analisi di uno studio di tipo Caso-Controllo.

7.1 PROC FREQ

La PROC FREQ fornisce, per il dataset in esame, le frequenze relative al valore di una variabile o alla combinazione dei valori relativi a due o più variabili.

Se la variabile in esame è continua o ha molti valori ad essa assegnati, le tabelle che otterremo riferite a quella variabile saranno poco informative.
Spesso i dati, prima di essere analizzati e presentati, devono essere organizzati in categorie (e, come vedremo, questo succederà quasi sempre nel caso dell'analisi di uno studio Caso-Controllo).

Nella sua forma più semplice, la procedura, quando riferita ad una variabile, genera una tabella ad un'entrata riassuntiva, con informazioni sulla frequenza di ogni valore, la sua percentuale sul totale e quella cumulativa.
Per due variabili si ottiene una tabella a doppia entrata contenente le frequenze di ogni cella, la loro percentuale sul totale generale, sul totale di riga e sul totale di colonna.

Nella PROC FREQ, per specificare per quali variabili costruire le tabelle, si usa il comando TABLES.

Forma generale di una PROC FREQ:

```
proc freq data = dataset ;
 by variabili;
 tables var1*var2 / < opzioni >;
```

Esempio 7.1

Alla fine di un corso si hanno i risultati delle diverse prove sostenute dagli studenti durante l'anno. Il voto finale deve essere calcolato facendo pesare per il 10% i voti dei quiz (punteggio massimo 50), per il 20% ognuno degli esami intermedi (punteggio massimo 100), per il 10% il voto di laboratorio (punteggio massimo 100) e per il 40% la prova finale (punteggio massimo 200). Si vuole poi cambiare il voto in un giudizio (ottimo, buono, discreto, sufficiente e insufficiente), valutando poi quanti studenti hanno ottenuto ottimo, quanti buono, e così via.
Si è poi curiosi di sapere se ci sia differenza nella distribuzione dei voti rispetto all'anno di corso degli studenti e al sesso.

Programma SAS.

```
filename indata 'c:epidemiologia\esempi\grades.dat';

data one;
 infile indata;
 input cod $ sesso $ anno quiz exam1 exam2 lab finexam;

 quiz = quiz*2;
 finexam = finexam/2;
 vcorso = .1*quiz + .2*(exam1+exam2) + .1*lab + .4*finexam;
 vcorso = vcorso/100;

      if        vcorso >=.90 then giud = 'a. ottimo
 else if .80 <= vcorso < .90 then giud = 'b. buono
 else if .70 <= vcorso < .80 then giud = 'c. discret
 else if .60 <= vcorso < .70 then giud = 'd. suff
 else if        vcorso < .60 then giud = 'e. insuff

run;


proc means;
 var vcorso quiz exam1 exam2 lab finexam;
 title 'Statistiche riassuntive della classe';
run;


proc freq ;
 tables giud giud*anno giud*sesso / nocol norow nopct;
 title 'Voti della classe';
run;
```

vengono lasciati degli spazi bianchi, perché SAS, definendo il valore della lunghezza della variabile carattere '`giud`' in base alla lunghezza del primo valore che incontra, tronca le eventuali altre stringhe più lunghe a

con questa opzione si indica di non mostrare i valori percentuali di colonna, di riga e totali.

OUTPUT 7.1. ESEMPIO DI TABELLA, RELATIVA AI VOTI SUDDIVISI PER SESSO.

```
                        Voti della classe                        80
                                          17:34 Sunday, May 3, 1998

                    TABELLA DI GIUD PER SESSO

                 GIUD          SESSO

                 Frequenza   ,f        ,m        ,  Totale
                 ffffffffffff^ffffffff^ffffffff^
                 a. ottimo   ,       4 ,       3 ,       7
                 ffffffffffff^ffffffff^ffffffff^
                 b. buono    ,       8 ,      15 ,      23
                 ffffffffffff^ffffffff^ffffffff^
                 c. discreto ,       3 ,       8 ,      11
                 ffffffffffff^ffffffff^ffffffff^
                 d. suff     ,       1 ,       3 ,       4
                 ffffffffffff^ffffffff^ffffffff^
                 e. insuff   ,       1 ,       3 ,       4
                 ffffffffffff^ffffffff^ffffffff^
                 Totale             17        32        49
```

7.2 Come inserire dei dati già presentati in forma tabulare.

Nell'esempio precedente abbiamo organizzato in tabella dei dati riferiti a record unici.
Può capitare invece di dover lavorare su dei dati che ci vengono forniti direttamente in forma tabulare; in questo caso i record, al posto dei singoli soggetti, dovranno essere le singole celle della tabella.

Facciamo un esempio:

Tabella 7.1 Tabella generale riferita ad uno studi Caso-Controllo.

	Esposizione		
Stato	Esposto	Non esposto	Totale
Controllo	16	48	64
Caso	40	20	60

Per inserire i dati della tabella 7.1 dovremo procedere nel seguente modo:

```
data cas_con;
   input stato $ espos $ count ;
   cards;
   control es 16
   control ne 48
   caso es 40
   caso ne 20
   ;
```

```
proc freq;
   weight count;
   tables stato*espos;
run;
```

Il data set ‘cas_con' contiene tre variabili: STATO è la variabile che indica lo stato del paziente, ESPOS è la variabile riferita all'esposizione ad un determinato fattore di rischio, COUNT il numero delle osservazioni con i rispettivi valori di stato e esposizione, in pratica corrispondenti ai valori nelle celle della tabella 7.1.

Il comando

```
weight ‘variabile’;
```

specifica al sistema che i dati della variabile corrispondente fanno riferimento a frequenze.