SAS

STATISTICAL ANALYSIS SYSTEM

Uso di SAS per le analisi statistiche

A cura di

Laura Neri

Dip. di Economia Politica e Statistica

Università degli Studi di Siena

PARTE I

IL SISTEMA SAS

Sistema integrato di prodotti software

- data entry, manipolazione archivi;
- stesura di report e grafici;
- analisi statistiche e matematiche;
- previsioni e supporto alle decisioni;
- ricerca operativa e project management;
- sviluppo di applicazioni.

Il fulcro del sistema SAS è il modulo SAS Base

- linguaggio SAS;
- procedure per analisi dei dati e stesura di report;
- macro-linguaggio.

ORGANIZZAZIONE DEL SISTEMA SAS

L’analisi dei dati si svolge seguendo due passi fondamentali:

1) L’organizzazione dei dati
2) L’analisi dei dati

Nel SAS System

Data Step

Inizia con un *data statement* e consente di leggere/creare e/o modificare archivi di dati

Proc Step

Inizia con un *proc statement*, ovvero richiama una procedura SAS, la esegue su un SAS dataset e produce dei risultati

NB la suddetta suddivisione è una semplificazione perché in realtà vedremo che anche il *proc statement* può creare un data set

Un programma SAS può essere costituito da

- Un solo Data Step
- Un solo Proc Step
- Più Data Step e/o più Proc Step

AMBIENTE DI LAVORO A FINESTRE

Le finestre base del SAS per Windows sono 5:

- **SAS Explorer** contenuto delle LIBRARIES e dei SAS datasets;
- **SAS Results** visualizzazione dei risultati di ogni procedura eseguita;

e le finestre di programmazione

- **SAS Enhanced Editor e Program Editor** scrittura istruzioni (programmi);
- **SAS Log** esito operazioni eseguite. Una volta eseguito il programma, SAS scrive dei messaggi nel SAS LOG, ignorare tali messaggi può essere pericoloso perché talvolta si ottengono dei risultati ma tali risultati potrebbero essere non corretti per qualche problema intercorso nelle istruzioni digitate;
- **SAS Output** risultati delle esecuzioni.

Inoltre:

SAS SYSTEM HELP per la consultazione della sintassi e delle opzioni del linguaggio e delle procedure

Visualizzazione dei risultati nella finestra di OUTPUT

Una volta ‘sottomesso’ il programma SAS (SAS windowing environment), i risultati appariranno nella finestra OUTPUT e nella finestra Results window.

SAS DATA SET

Data Step e Proc Step si applicano ai file in formato SAS - SAS DATA SET -

I SAS DATA SET sono organizzati in forma rettangolare

V_1	V_2	....	....	V_k

- ogni riga rappresenta un'osservazione;
- ogni colonna rappresenta una variabile
- tutte le osservazioni possiedono le stesse variabili➔ le variabili non osservate per una data osservazione sono registrate come mancanti (*missing*)

Ogni SAS dataset è autodescrittivo, infatti esplicita:

- Il nome del *SAS data set*
- Il nome delle variabili, le *label*, ed il formato
- Il contenuto delle variabili per ogni osservazione

LIBRERIE: DEFINIZIONE E GESTIONE

Per definire una libreria permanente si possono seguire due strade:

- fare click sul comando New Library della barra Menu ed inserire le informazioni richieste, tra cui il nome della library ed il percorso fisico associato alla libreria stessa
- Scrivere nella finestra Program Editor l'istruzione

LIBNAME *mylibname 'path'*;

Dove:

mylibname	è il nome attribuito alla libreria
path	è il percorso fisico dove vengono memorizzati i dati

IL LINGUAGGIO SAS: concetti introduttivi

In SAS vengono utilizzati SAS statement per scrivere una serie di istruzioni (*statement*) che vanno a costituire il SAS PROGRAM.

Naturalmente per scrivere un programma SAS è necessario utilizzare il linguaggio appropriato il SAS LANGUAGE.

Il SAS PROGRAM è costituito da una serie di istruzioni ordinate.

La prima regola da seguire nella stesura di un programma è:

every SAS statement ends with a semicolon

Istruzioni SAS (*SAS statement)*

- Iniziano con una parola chiave
- Terminano sempre con il carattere punto e virgola
- Possono essere scritte con lettere maiuscole e minuscole, il linguaggio SAS non è *case sensitive*
- Possono iniziare in una qualsiasi colonna della riga e proseguire su più righe

- Più istruzioni possono essere scritte sulla stessa riga ma devono essere separate da punto e virgola

Per **salvare** un programma SAS si fa uso del Menu principale. Il programma viene salvato con estensione .SAS

REGOLE PER NOMI SAS DEFINITI DALL'UTENTE

- Numero massimo di caratteri dipende dal tipo di nomi (32, 8)
- Il primo carattere deve essere una lettera o un underscore "_"
- I caratteri successivi al primo possono essere lettere, cifre, "_"
- Indipendentemente da maiuscolo o minuscolo, SAS converte tutto in maiuscolo
- Nei nomi non possono comparire spazi bianchi
- Non sono ammessi caratteri speciali ($,£,*…)
- Non si possono utilizzare nomi di variabili automatiche del sistema (_N_, *ERROR*_)
- Non si possono utilizzare nomi che il SAS riserva a librerie speciali (LIBRARY, SASHELP, WORK, USER, MAPS..)

ESECUZIONE PROGRAMMI SAS

Esecuzione in modo semi-interattivo

- Stesura del programma nella finestra Enhanced/Program editor
- Visualizzazione risultati nella finestra Output
- Segnalazioni inviate dal sistema (messaggi di errore, warning, altro) nella finestra Log

Il programma SAS viene automaticamente compilato dal sistema prima di essere eseguito:

- ❖compilazione ed esecuzione avvengono a blocchi (data step, proc step);
- ❖i blocchi si chiudono con la parola chiave RUN;

INTRODUZIONE AL DATA STEP

La struttura del Data Step è la seguente:

DATA *nomefile (opzioni)***;**

…..

RUN;

L'ISTRUZIONE **DATA** e L'ISTRUZIONE **SET**

DATA *nomedataset <opzioni>;*

SET *nomedataset;*

RUN;

Dove *nomedataset* è il nome di un Sas dataset. Tale nome può essere scritto a due livelli:

nome1.nome2

- *nome1*: è il nome di primo livello ed indica la libreria in cui memorizzare il Sas data set (per default la libreria è WORK)
- *nome2*: è il nome del Sas data set che viene memorizzato nel percorso fisico associato alla libreria.

Come opera il suddetto blocco di istruzioni?

- ✓ legge il file indicato all'istruzione **SET**
- ✓ scrive sul file indicato all'istruzione **DATA**

PARTE II

LETTURA DI DATI DI TIPO ASCII

- Istruzione **INFILE**
- Istruzione **INPUT**
- Istruzione **DATALINES** (**CARDS)**

Istruzione **INFILE**

Indica al sistema dove leggere i dati

INFILE *'nomefile' [opzioni];*

nomefile

nome, con eventuale percorso, del file ASCII da leggere o parola chiave CARDS se i dati sono inseriti da programma

Istruzione **INPUT**

Definisce nome, tipo e modo di lettura delle variabili

I modi di lettura sono: a lista, a colonna, con formato

LETTURA A LISTA

Possibile quando i dati sono registrati in formato libero, con almeno uno spazio bianco tra un campo ed il successivo

INPUT *var1 var2 var3;*

INPUT *var1-var10;*

INPUT *var1 $ var2;*

Lettura a lista da file esterno

Input dataset: INPUT_LISTA.TXT

100 1 34
200 1 65
300 2 29
400 1 31
500 1 45
600 2 40
700 2 68
800 1 51
900 1 48

```
data pippo;
infile 'F:\written\didattica\CorsoSAS\input_lista.txt';
input codice genere eta;
run;
```

Lettura a lista e scrittura del file da programma

```
data pluto;
input x1-x3;
datalines;
1 5 7
9 3
2
6 9 8
13 5 8
;
run;
```

equivalentemente

```
input x1-x3;
cards;
1 5 7
9 3
2
6 9 8
13 5 8
;
run;
```

Output SAS data set PLUTO

x1	X2	X3
1	5	7
9	3	2
6	9	8
13	5	8

LETTURA A COLONNA

Input dataset: INPUT_COLONNA.TXT

100m34
200m65
300f29
400m31
500m45
600f40
700f68
800m51
900m48
999m36

```
data pippo;
infile
'F:\written\didattica\CorsoSAS\input_colonna.txt';
input codice 1-3 genere $ 4 eta 5-6;
run;
```

LETTURA CON FORMATO

Non si specificano le colonne ma la lunghezza di ogni campo

INPUT *var informat.* ;

INPUT (*varlist) (informat list*) ;

INPUT (*varlist) ([n*] informat.*) ;

Sintassi generale *informat*

w. legge numeri interi o decimali con punto decimale codificato nel campo

$w. Legge stringhe di caratteri ASCII

```
*LETTURA A FORMATO: i campi non sono separati;
data pippo;
infile
'F:\written\didattica\CorsoSAS\input_colonna.txt';
input codice 3.  genere $1.  eta 2.;
run;
```

In questo caso i dati sono separati da uno spazio quindi devo incrementare il puntatore

```
*LETTURA A FORMATO: i campi sono separati da uno
spazio;
data pippo;
infile
'F:\written\didattica\CorsoSAS\input_lista.txt';
input codice 3. +1 genere $1. +1 eta 2.;
run;
```

alternativamente il simbolo @i indica al puntatore di spostarsi alla colonna i

```
data pippo;
infile
'F:\written\didattica\CorsoSAS\input_lista.txt';
input codice 3. @5  genere $1. @7 eta 2.;
run;
```

Si tenga presente che la lettura con formato è indispensabile nel caso in cui ci siano da leggere dei campi contenenti una data.

Esempio: Supponiamo di avere dei dati provenienti dalla degustazione di 4 vini di marche diverse. I degustatori sono 3. I dati inseriti nel file sono: la marca del vino, l'anno di

produzione, la data di degustazione, i punteggi dei tre degustatori.

```
*Esempio di lettura con formato ;
data punti_vino;
infile cards;
input marca $1 +1 anno 2. +1 data ddmmyy10. +1
(punti1-punti3) (3*4.);
cards;
A 93 20-11-1994 7.8 7.1 6.5
B 95 23-12-1998 7.9 7.6 7.5
C 99 10-11-2000 6.5 7.0 6.8
D 00 30-10-2002 5.9 6.4 7.2
;
run;

proc print data=punti_vino;
var marca anno punti1-punti3 data;
format data date8.;
run;

proc print data=punti_vino;
var marca anno punti1-punti3 data;
format data MONyy.;
run;


proc print data=punti_vino;
var marca anno punti1-punti3 data;
format data DDMMyy.;
run;
```

LETTURA DI PIU' OSSERVAZIONI DA UNO STESSO RECORD

@@ indica la reale fine del record, quindi nell'esempio che segue i dati vengono letti a coppie

```
/*LETTURA DI PIU' OSSERVAZIONI DA UNO STESSO
RECORD*/
data unrecord;
input genere $ peso @@;
cards;
m 60 f 50 m 68
m 82 f 55 f 56
;
run;
```

LETTURA DI UNA OSSERVAZIONE SU PIU' RECORDS

/ indica di andare al record successivo

#n indica di andare al record n

Nell'esempio l'osservazione relativa a ciascun vino è registrata su 3 record:

- nel 1° la marca del vino, l'anno di produzione,
- nel 2° la zona di provenienza del vino,
- nel 3° il punteggio attribuito al vino.

```
/*LETTURA DI una  OSSERVAZIONE SU PIU' RECORD*/
*ESEMPIO VINO, ELIMINATA LA DATA;
data punti_vino_zona;
infile cards;
input marca $ anno  / ZONA $  #3 punti;
cards;
A 1993
ZONA1
7
B 1995
ZONA2
7.9
C 1999
ZONA3
6.5
D 2000
ZONA4
5.9
;
RUN;
```

DATI NEI RECORD E VARIABILI IN INPUT

(a) le variabili specificate esauriscono completamente i dati presenti nel record;
(b) le variabili specificate richiedono un numero di dati minore rispetto a quelli esistenti;
(c) le variabili specificate richiedono un numero di dati superiore rispetto a quello esistente.

(c) SAS prosegue nell'input del primo record leggendo dalla riga seguente; su tale riga salta i successivi campi e va a capo a leggere nel record successivo partendo dalla prima colonna e scrive sulla finestra LOG il messaggio:

```
NOTE: SAS went to a new line when INPUT
statement reached past the end of a
line.
```

```
data uno;
input var1-var6;
cards;
18 66 24 23
19 45 34 20
```

```
40 97 56 43
23 56 73 90
;
run;

NOTA: SAS è passato a una nuova riga quando l'istruzione
INPUT ha superato la fine di una riga.
NOTA: Il data set WORK.UNO ha 2 osservazioni e 6
variabili.
```

Oss	var1	var2	var3	var4	var5	var6
1	18	66	24	23	19	45
2	40	97	56	43	23	56

CONTROLLO ESAURIMENTO DATI

INFILE *nome_file* **MISSOVER**;

Fa sì che la lettura non vada oltre la fine del record specificato nell'istruzione cards, assegna valore mancante a quelle variabili per cui non vi sono dati.

```
Data tre;
infile cards missover;
cards;
```

```
18 66 24 23
19 71 26 21
20 68 23 29
;
run;
proc print data=tre;
run;
```

```
NOTE: The data set WORK.TRE has 3 observations and
6 variables.
NOTE: The DATA statement used 0.0 seconds.
```

OBS	VAR1	VAR2	VAR3	VAR4	VAR5	VAR6
1	18	66	24	23	.	.
2	19	71	26	21	.	.
3	20	68	23	29	.	.

IMPORTAZIONE/ESPORTAZIONE AUTOMATICA DI DATI

Dal Menu principale:

- ❖File
- ❖Import Data/Export Data
- ❖….

*** tracciato record indagine consumi ISTAT**

*esempio di importazione da EXCEL dati Istat “indagine sui Consumi delle famiglie italiane”, il file CONSUMO_TOSCANA.XLS è una selezione di variabili per la sola Regione Toscana

Se l’operazione di importazione ha funzionato nella finestra di Log apparirà:

```
NOTE: WORK.CONSUMO_TOSCANA was successfully created
```