UNIVERSITAT DE BARCELONA

Introducció al SAS

Ramon Alemany

Dept. Econometria, Estadística i Economia Aplicada

Universitat de Barcelona

PROGRAMA

1. **GESTIÓ D'ARXIUS DE DADES**
    1. Importar i Exportar dades des d'arxius externs
    2. Especificació de l'estructura d'arxius externs i control de lectura.
2. **PROGRAMACIÓ EN EL DATA STEP**
    1. Tractament específic de variables segons els seus atributs. Funcions.
    2. Manipulació d'observacions.
    3. Gestió de valors perduts (missing values)
    4. Diagnòstic i depuració d'errors.
3. **COMBINACIÓ DE CONJUNTS DE DADES SAS**
    1. Concatenació de conjunts de dades SAS.
    2. Fusió de conjunts de dades SAS.
    3. Actualització de conjunts de dades SAS.
4. **PROCEDIMENTS BÀSICS**
    1. Introducció al PROC STEP. Comandes comunes
    2. Creació d'informes tipus i taules.
    3. Procediments estadístics i gràfics.
5. **LLENGUATGE MATRICIAL AMB SAS**
    1. Procediment IML i aplicacions.
    2. Operant amb matrius i Funcions IML.
    3. Programant amb IML. Mòduls: programa i funció.

Bibliografia:

Kleinman, K. i Horton, N.J. (2010) ***SAS and R. Data Management, Statistical Analysis and Graphics***. Champan and Hall. London, New York.

PROGRAMES SAS

- Extensió de l'arxiu: .sas
- La finestra de l'Editor té quatre utilitats:
  - Accés i edició de programes SAS existents
  - Escriure nous programes SAS
  - Executar programes SAS (Submit)
  - Salvar programes SAS

- Programa SAS – seqüència de passes que l'usuari envia a executar

- Execució de programes SAS
  - Programa sencer
  - Selecció de parts del programa SAS

PROGRAMES SAS (2)

- Regles de Sintaxi per les comandes de SAS
  - Format lliure – podem usar minúscules o majúscules
  - Normalment s'inicien amb una paraula clau identificadora de l'etapa
  - Poden ocupar múltiples línies
  - Les comandes sempre finalitzen amb un punt-i-coma ;
  - Pot haver-hi més d'una comanda per línia (no és recomanable)

- Errors habituals
  - Paraules clau mal escrites
  - Oblit dels punts-i-comes
  - Opcions invàlides
  - Els errors apareixen en la finestra LOG

PROGRAMES SAS (3)

- Dues etapes bàsiques en tots els programes SAS:
  - Data Step
    - S'utilitza per crear bases de dades SAS i manipular les dades,
    - Comença amb la paraula clau DATA
  - Proc's Step
    - Usats típicament per processar bases de dades SAS
    - Comencen amb la paraula clau PROC

- El final d'una etapa DATA o una etapa PROC s'indiquen per:
  - RUN – majoria de les etapes
  - QUIT – algunes etapes
  - Començant una altra etapa (DATA o PROC)

PROGRAMES SAS (4)

- Els resultats de l'execució dels programes SAS apareixen en dues finestres
  - SAS log
    - Informació de com s'ha processat el programa SAS
    - Inclou missatges d'avis i d'error
    - La informació va apareixent a mesura que es van executant les etapes del programa SAS
  - SAS output
    - Informes o resultats generats pels PROC's de SAS
    - Els resultats van apareixent a mesura que es van executant les etapes del programa SAS

BASES DE DADES DE SAS I LLIBRERIES

- Bases de dades SAS
  - Arxiu específicament estructurat que conté dades.
  - Extensió de l'arxiu - .sas7bdat
  - Format de files I columnes – semblant a Excel
    - Columnes – variables
    - Files – observacions
  - Dos tipus de variables (bàsicament)
    - Caràcter – conté qualsevol valor (lletres, números, símbols, etc.)
    - Numèric – números amb coma flotant
  - Es localitzen en les llibreries de dades de SAS

BASES DE DADES DE SAS I LLIBRERIES (2)

- Llibreries de dades de SAS
  - Contenen bases de dades de SAS
  - S'identifiquen assignant un nom de referència a la llibreria (libref)
  - **Temporals**
    - Llibreria Work
    - Les bases de dades SAS d'aquesta llibreria s'esborren en tancar la sessió de SAS
    - No necessiten un nom de referència
  - **Permanents**
    - Les bases de dades SAS no es perden en finalitzar la sessió
    - Llibreria SASUSER (per defecte)
    - Es poden crear i accedir a les pròpies llibreries

BASES DE DADES DE SAS I LLIBRERIES (3)

- Llibreries de dades SAS
  - Assignació de noms de referència a les llibreries

```
LIBNAME libref 'SAS-data-library';
```

  - Regles de sintaxi pels noms de les llibreries
    - 8 caràcters o menys
    - Han de començar per una lletra o un guió baix (_)
    - La resta de caràcters poden ser lletres, números o guions baixos (_)

BASES DE DADES DE SAS I LLIBRERIES (4)

- Llibreries de dades SAS
  - Identificant bases de dades de SAS amb Llibreries
    `libref.filename`

  - Accedint a bases de dades SAS ubicades a llibreries
    Exemple:
    ```
    DATA noves_dades;
    SET libref.filename;
    RUN;
    ```

  - Creant bases de dades SAS dins de llibreries
    Exemple:
    ```
    DATA libref.filename;
    SET velles_dades;
    RUN;
    ```

CREACIÓ DE BASES DE DADES SAS

- Creant una base de dades SAS a partir de dades en brut

Quatre mètodes:

1. Lectura de dades en el propi programa SAS

2. Lectura de dades des d'arxius en format ASCII

3. Importar dades existents usant les opcions del menú Import

4. Entrada manual de dades usant l'Editor de Taules

1. Lectura de dades en el propi programa de SAS

```
DATA nombase ;
  INPUT noms variables a llegir i formats ;
  CARDS;
   . . .
  dades
   . . .
  ;
RUN;
```

2. Lectura de dades des de arxius ASCII

```
DATA nombase ;
  INFILE arxiu extern en format ASCII  ;
  INPUT noms variables a llegir i formats ;
RUN;
```

Si la longitud de l'arxiu és superior a 80 columnes, aleshores hem de fer ús de l'opció **lrecl = n** en la comanda **INFILE**.

Comanda **DATA**

Assenyala l'inici de l'etapa DATA i dóna nom a la base de dades SAS que s'està creant. També pot incloure una etiqueta per la base de dades.

```
DATA test  (LABEL='Base de dades de prova');
```

Comanda **INFILE**

Identifica l'origen dels registres de dades que volem llegir. Es pot fer de dues formes:

- Especificant el nom i ubicació de l'arxiu extern

```
DATA test;
INFILE 'F:\TESTING.DAT';
```

- Especificant una referència d'un arxiu extern creada prèviament amb la comanda FILENAME.

```
FILENAME testfile 'F:\TESTING.DAT';
DATA test;
INFILE testfile;
```

Comanda **INPUT**

Descriu les variables a llegir, donant-hi noms a cadascuna i identificant la seva localització. Aquesta localització es pot indicar de dues formes:

- Columna
- Llista

INPUT en columnes

El nom de la variable va seguida pels números de les columnes on es troben els valors de la variable corresponent. Els valors de les variables han d'estar en les mateixes columnes per a tots els registres, i poden ser variables numèriques o caràcter (en aquest cas cal posar el signe $ entre el nom de la variable i els números de les columnes)

Per exemple:

```
INPUT nom $ 1-8 edat 11-12;
```

llegirà els següents registres:

```
----+----1----+----2----+----3
PEREZ      12
LOPEZ      23
GARCIA     30
```

Amb l'**INPUT** en columnes:

- Les variables caràcter poden incloure blancs

```
INPUT nom $ 1-13 edat 17-18;

----+----1----+----2----+----3
PEREZ PEREZ     12
LOPEZ LOPEZ     23
GARCIA GARCIA   30
```

- Els valors caràcter poden tenir una longitud de màxim 200 caràcters

- Un camp en blanc o amb un punt (.) es considerat 'missing' i no provoca errades de lectura

```
INPUT nom $ 1-13 edat 17-18;
    ----+----1----+----2----+----3
    PEREZ PEREZ     12
    LOPEZ LOPEZ     .
    GARCIA GARCIA   30
```

- Els valors de les variables poden ser llegits en qualsevol ordre, sigui quina sigui la seva posició en el registre.

- Alguns valors o parts de valors poden ser tornats a llegir.

```
INPUT id 10-15 grup 13;
```

Útils amb l'**INPUT** en columnes i també en Llista:

- #n mou el punter a la línea n

```
INPUT nom $ 1-13 #2 edat 1-2;

----+----1----+----2----+----3
PEREZ PEREZ
12
LOPEZ LOPEZ
23
GARCIA GARCIA
30
```

- / avança el punter fins a la columna 1 de la següent línea

```
INPUT nom $ 1-13 edat 17-18 / grup;
```

- +n mou el punter n columnes endavant

```
INPUT nom $ 1-13 +4 edat;
```

- @n mou el punter fins a la columna n

```
INPUT nom $ 1-13 @17 edat;
```

INPUT en llista

En aquesta forma es posa la llista de noms de les variables (també cal posar el signe $ després del nom d'una variable caràcter).

Es poden llegir valors de variables separats per blancs o un altre caràcter prefixat que caldrà indicar amb l'opció **DELIMITER=** de la comanda **INFILE.**

El INPUT en llista per defecte presenta moltes limitacions:

- Els valors 'missing' cal indicar-los amb un punt (.)
- Els valors caràcter no poden contenir blancs i han de tenir com a màxim 8 dígits
- Les variables s'han de llegir en l'ordre en què es troben

Per exemple:

```
INPUT nom $ edat ;
```

llegirà els següents registres:

```
----+----1----+----2----+----3
PEREZ  12
LOPEZ  23
GARCIA 30
```

Exemple DELIMITER=

```
DATA nums;
  INFILE datalines  DELIMITER='&'  DSD;
  INPUT X Y Z;
  datalines;
  1&2&3
  4&5&6
  7&8&9
  ;
 RUN;
```

Per a arxius tabulats [TAB]:

```
DELIMITER='09'x
```

Opció DSD

L'opció DSD canvia el tractament dels delimitadors que fa SAS i agafa com a delimitador per defecte la coma. Quan s'especifica DSD el SAS tracta dos delimitadors consecutius com a 'missings' i elimina les cometes de les variables caràcter llargues o que inclouen blancs.

```
DATA scores;
  INFILE datalines DELIMITER=',';
  INPUT test1 test2 test3;
  datalines;
  91,87,95
  97,,92
  ,1,1
  ;
```

Output:

Obs	test1	test2	test3
1	91	87	95
2	97	92	1

```
DATA scores;
 INFILE datalines DELIMITER=','  DSD ;
 INPUT test1 test2 test3;
 datalines;
 91,87,95
 97,,92
 ,1,1
 ;
```

Output:

Obs	test1	test2	test3
1	91	87	95
2	97	.	92
3	.	1	1

Útils amb l'**INPUT** en llista:

- : permet especificar un format d'entrada de més de 8 dígits per les variables caràcter, eliminant les cometes.
- ~ manté les cometes de les variables caràcter llargues
- Informats $w. w.d DDMMYY10.
- @ una línea de dades serà llegida per més d'una comanda INPUT
- @@ una línea de dades té valors per més d'una observació

Lectura de dades separades per comes i que poden contenir comes com a part de valors caràcter:

```
DATA scores;
INFILE datalines DSD;
INPUT Name : $9. Score Team : $25. Div $;
datalines;
Joseph,76,"Red Racers, Washington",AAA
Mitchel,82,"Blue Bunnies, Richmond",AAA
Sue Ellen,74,"Green Gazelles, Atlanta",AA
;
RUN;
```

Output:

Obs	Name	Score	Team	Div
1	Joseph	76	Red Racers, Washington	AAA
2	Mitchel	82	Blue Bunnies, Richmond	AAA
3	Sue Ellen	74	Green Gazelles, Atlanta	AA

Proveu ara:

```
DATA scores;
INFILE datalines;
INPUT Name : $9. Score Team : $25. Div $;
datalines;
Joseph,76,"Red Racers, Washington",AAA
Mitchel,82,"Blue Bunnies, Richmond",AAA
Sue Ellen,74,"Green Gazelles, Atlanta",AA
;
RUN;
```

Proveu ara:

```
DATA scores;
INFILE datalines DSD;
INPUT Name $ Score Team $ Div $;
datalines;
Joseph,76,"Red Racers, Washington",AAA
Mitchel,82,"Blue Bunnies, Richmond",AAA
Sue Ellen,74,"Green Gazelles, Atlanta",AA
;
RUN;
```

Output:

Obs	Name	Score	Team	Div
1	Joseph	76	Red Race	AAA
2	Mitchel	82	Blue Bun	AAA
3	Sue Elle	74	Green Ga	AA

I ara:

```
DATA scores;
INFILE datalines DSD;
INPUT Name : $9. Score Team ~ $25. Div $;
datalines;
Joseph,76,"Red Racers, Washington",AAA
Mitchel,82,"Blue Bunnies, Richmond",AAA
Sue Ellen,74,"Green Gazelles, Atlanta",AA
;
RUN;
```

Output:

Obs	Name	Score	Team	Div
1	Joseph	76	Red Racers, Washington	AAA
2	Mitchel	82	Blue Bunnies, Richmond	AAA
3	Sue Ellen	74	Green Gazelles, Atlanta	AA

COMPROVAR UNA CONDICIÓ ABANS DE CREAR UNA OBSERVACIÓ

```
DATA red_team;
INPUT Team $ 13-18 @;
 IF Team= 'red' ;
 INPUT IdNumber 1-4 StartWeight 20-22 EndWeight 24-26;
datalines;
1023 David  red    189 165
1049 Amelia yellow 145 124
1219 Alan   red    210 192
1246 Ravi   yellow 194 177
1078 Ashley red    127 118
1221 Jim    yellow 220   .
;
RUN;
```

Seria el mateix (però més eficient) que:

```
DATA red_team;
INPUT IdNumber 1-4 Team $ 13-18 StartWeight 20-22
  EndWeight 24-26;
datalines;
1023 David  red    189 165
1049 Amelia yellow 145 124
1219 Alan   red    210 192
1246 Ravi   yellow 194 177
1078 Ashley red    127 118
1221 Jim    yellow 220   .
;
RUN;
DATA red_team;
 SET red_team;
 IF Team='red';
RUN;
```

Quan tenim dos tipus de dades diferents en els registres:

```
DATA class;
INPUT Type $ 1 @;
 IF Type='c' then INPUT curs $ prof $;
 IF Type='s' then INPUT Nom $ Id;
datalines;
c HISTO1 Watson
s Williams 0459
s Lopez 0357
s Sanchez 0224
c MATH1 Alegre
c ECONOMET1 Calonge
;
RUN;
```

CREANT MÚLTIPLES OBSERVACIONS A PARTIR D'UN REGISTRE

```
DATA body_fat;
INPUT gender $ PercentFat @@;
datalines;
m 13.3 f 22 m 14.5 f 22.1
m 10.1 m 22.3 f 24.5 m 28.7
f 23.5 f 18.1 m 19.3 f 21.4
;
RUN;
```

Proveu ara sense el @@....

LLEGINT MÚLTIPLES REGISTRES PER CREAR UNA OBSERVACIÓ

```
DATA club2;
INPUT IdNumber 1-4 Name $ 6-23;
INPUT Team $;
INPUT StartWeight 1-3 EndWeight 5-7;
datalines;
1023 David Shaw
red
189 165
1049 Amelia Serrano
yellow
145 124
1219 Alan Nance
red
210 192
1246 Ravi Sinha
yellow
194 177
;
RUN;
```

Seria més eficient fer un sol INPUT:

```
DATA club2;
INPUT IdNumber 1-4 Name $ 6-23 / Team $ / StartWeight 1-3
  EndWeight 5-7;
datalines;
1023 David Shaw
red
189 165
1049 Amelia Serrano
yellow
145 124
1219 Alan Nance
red
210 192
1246 Ravi Sinha
yellow
194 177
;
RUN;
```

Si només volem llegir les variables IdNumber, StartWeight i EndWeight:

```
DATA club2;
INPUT IdNumber 1-4 // StartWeight 1-3 EndWeight 5-7;
datalines;
1023 David Shaw
red
189 165
1049 Amelia Serrano
yellow
145 124
1219 Alan Nance
red
210 192
1246 Ravi Sinha
yellow
194 177
;
RUN;
```

Llegint variables de múltiples registres en qualsevol ordre:

```
DATA club2;
INPUT #2 Team $
      #1 IdNumber 1-4
      #3 StartWeight 1-3 EndWeight 5-7
      #1 Name $ 6-23;
datalines;
1023 David Shaw
red
189 165
1049 Amelia Serrano
yellow
145 124
1219 Alan Nance
red
210 192
1246 Ravi Sinha
yellow
194 177
;
RUN;
```

3. Importar dades existents usant les opcions del menú Import

1. File → Import Data
   Archivo → Importar datos...

2. Standard data source
   → seleccionar el format de l0'arxiu a importar

3. Especificar la ubicació de l'arxiu o *Browse* per seleccionar-lo

4. Donar nom a la nova base de dades i especificar la llibreia

- Formats compatibles

  - Microsoft Excel
  - Microsoft Access
  - Comma Separate (.csv)
  - Tab Delimited (.txt)
  - dBASE (.dbf)
  - JMP
  - SPSS Files
  - Lotus
  - Stata
  - Paradox

4. Entrada manual de dades usant l'Editor de Taules

1. Tools → Table Editor
   Herramientas → Editor de Tablas

2. Entrar dades manualment en la taula
   - Observacions per files
   - Variables en columnes

3. Clic botò dreta Columna → Atributs de la Columna
   Nom de la variable, Etiqueta de la variable, Tipus (Caràcter/Numèrica), Format, Informat
   Informats determinen com es llegeixen les dades
   Formats determinen com es visualitzen

4. Cerrar ventana → Guardar cambios → Si
   → Especificar el nom de l'arxiu i el directori

MANIPULACIÓ DE BASES DE DADES

- Crear una nova base de dades SAS usant una d'existent
  - Especificar el nom de la nova base de dades SAS després de la comanda DATA
  - Usar la comanda SET per identificar la base de dades SAS a llegir
  - Sintaxi:

```
DATA output_data_set;
    SET input_data_set;
    <additional SAS statements>;
RUN;
```

  - Per defecte la comanda SET llegeix totes les observacions i totes les variables de la base de dades existent i les copia a la nova base de dades SAS.

MANIPULACIÓ DE BASES DE DADES (2)

- Comandes del DATA
  - Avaluar una expressió
  - Assignar un valor a una variable
  - Forma general: variable = expressió;
  - Exemple: `km_per_hora = distancia/temps;`

- Funcions SAS
  - Funcions aritmètiques, càlcul d'estadístic senzills, manipulació de dates, etc.
  - Forma general: variable=funció(arg1, arg2,...);
  - Exemple:

```
Temps_treb = sum(Dia1,Dia2, Dia3, Dia4, Dia5);
```

MANIPULACIÓ DE BASES DE DADES (3)

- Seleccionant Variables
  - Amb DROP i KEEP determinem quines variables s'inclouran en la nova base de dades SAS.
  - Dues formes:
    - DROP i KEEP com a comandes
      - ```
        DROP = Variable1 Variable2;
        KEEP = Variable3 Variable4 Variable5;
        ```
    - Opcions DROP i KEEP en la comanda SET
      - ```
        SET input_data_set (KEEP=Var1);
        ```

MANIPULACIÓ DE BASES DE DADES (4)

- Processament d'observacions amb condicions

- Usant IF-THEN-ELSE
  - Forma general:

```
IF <expression1> THEN <statement>;
ELSE IF <expression2> THEN <statement>;
ELSE <statement>;
```

  - <expression> és una comanda verdader/fals, com per exemple:
    - Day1=Day2, Day1 > Day2, Day1 < Day2
    - Day1+Day2=10
    - Sum(day1,day2)=10
    - Day1=5 and Day2=5

MANIPULACIÓ DE BASES DE DADES (5)

- Si <expression1> és verdader , es processa <statement>

- ELSE IF i ELSE només es processen si <expression1> és fals

- Usem DO i END per executar grups de comandes
- Forma general:

```
IF <expression> THEN DO;
    <statements>;
END;
ELSE DO;
    <statements>;
END;
```

MANIPULACIÓ DE BASES DE DADES (6)

- Seleccionar files (Observacions)
  - Dues formes
    - Usant la comanda IF
    - Usant l'opció WHERE en la comanda SET
  - Comanda IF
    - Només selecciona les observacions en la nova base de dades per les què l'expressió és verdadera;
    - Forma general: IF <expression>;
    - Exemple:

```
IF career = 'Teacher';
IF sex ne 'M';
```

    - En el segon exemple, només les observacions per les què sex no és igual a 'M' s'escriuran en la nova base de dades

MANIPULACIÓ DE BASES DE DADES (7)

- Opció WHERE en la comanda SET
  - Es tracta de només llegir aquelles observacions de la base de dades per les què l'expressió és certa
  - Forma General :

    SET input_data_set (where=(<expression>));
  - Exemple:

    ```
    SET vacation (where=(destination='Bermuda'));
    ```

- Comparació
  - El resultat en la nova base de dades és el mateix
  - Comanda IF – es llegeixen totes les files de la base de dades original
  - Opció WHERE – només es llegeixen les files per les què l'expressió és verdadera
  - La diferència està en el temps de CPU quan treballem amb grans bases de dades

MANIPULACIÓ DE BASES DE DADES (8)

- El PROC SORT ordena dades d'acord amb els valors de les variables que s'especifiquen
- Forma General :

```
PROC SORT DATA=input_data_set <options>;
    BY Variable1 Variable2;
RUN;
```

- En l'exemple, s'ordenen les dades d'acord als valors de la Variable1 i després als de la Variable2;
- Per defecte, SAS ordena de forma ascendent: Números de més petit a més gran, i d'A a Z
- Especifiquem la comanda DESCENDING per fer-ho en ordre descendent

```
BY Ciutat DESCENDING Població;
```

SAS ordena primer per ciutats en ordre ascendent (A a Z) i després per Població de més gran a més petita

MANIPULACIÓ DE BASES DE DADES (9)

- Opcions del PROC SORT:
  - NODUPKEY
    - Elimina les observacions que tenen els mateixos valors per les variables especificades en la comanda BY
  - OUT=output_data_set
    - Per defecte, el PROC SORT substitueix la base de dades original amb les observacions ordenades d'acord amb les variables especificades
    - Usant aquesta opció OUT, PROC SORT crea una nova base de dades amb les observacions ordenades deixant la base de dades orignal inalterada

PROCESSAMENT DE DADES

- Processament de bases de dades
  - L'etapa DATA llegeix dades d'una base de dades existent o d'un arxiu de dades extern registre a registre
  - La dinàmica és:
    1. Llegir el registre actual i posar-lo en el Vector de Dades del Programa (PDV)
    2. Processar les comandes SAS
    3. Copiar el contingut del PDV a la nova base de dades
    4. Passar al següent registre de la base de dades
    5. Iterar en l'etapa 1
  - Es processa un registre cada vegada, per la qual cosa no podem sumar el valor d'una variable en una observació amb el valor de la mateixa variable de la següent observació

COMBINACIÓ DE BASES DE DADES DE SAS

- Concatenació vertical
  - “Enganxar” bases de dades una a sobre de l’altra
  - Si una de les bases de dades no conté una variable que les altres bases de dades si tenen, aleshores la variable en la nova base de dades tindrà ‘missings’ per a les observacions d’aquella base de dades.
  - Forma General:

```
DATA output_data_set;
  SET data1 data2;
run;
```

  - El PROC APPEND també es pot utilitzar:

```
PROC APPEND
  BASE=dades_base
  DATA=dades_afegir;
```

```
DATA papes;
INPUT famid nom $ renda ;
CARDS;
2 Art 22000
1 Bill 30000
3 Paul 25000
;
RUN;
DATA mames;
INPUT famid nom $ renda ;
CARDS;
1 Bess 15000
3 Pat 50000
2 Amy 18000
;
RUN;
DATA papmam;
SET papes mames;
RUN;

PROC PRINT DATA=papmam;
RUN;
```

Els dos arxius tenen noms de variables diferents pel mateix atribut. Per exemple, la renda s'anomena *rendpap* pels pares i *rendmam* per les mares

```
DATA papes;
INPUT famid nom $ rendpap ;
DATALINES;
2 Art 22000
1 Bill 30000
3 Paul 25000
;
RUN;
DATA mames;
INPUT famid nom $ rendmam ;
DATALINES;
1 Bess 15000
3 Pat 50000
2 Amy 18000
;
RUN;
DATA papmam;
SET papes(IN=pap) mames(IN=mam);
IF pap=1 THEN papmam ="pap";
IF mam=1 THEN papmam ="mam";
run;
PROC PRINT DATA=papmam;
RUN;
```

Solució 1

```
DATA papmam;
SET papes(IN=pap) mames(IN=mam);
 IF pap=1 THEN DO;
  papmam="pap";
  renda=rendpap;
 END;
 IF mam=1 THEN DO;
  papmam="mam";
  renda=rendmam;
END;
RUN;
PROC PRINT DATA=papmam;
RUN;
```

Solució 2

```
DATA papmam;
SET papes(RENAME=(rendpap=renda)) mames(RENAME=(rendmam=renda));
RUN;
PROC PRINT DATA=papmam;
RUN;
```

```
DATA paps;
 INPUT famid nom $ renda fulltime;
DATALINES;
2 Art  22000 0
1 Bill 30000 1
3 Paul 25000 1
;
RUN;

DATA mames;
 INPUT famid nom $ renda fulltime $1.;
DATALINES;
1 Bess 15000 N
3 Pat  50000 Y
2 Amy  18000 N
;
RUN;
```

COMBINACIÓ DE BASES DE DADES DE SAS (2)

- Concatenació Horitzontal (Merging Data Sets)
  - Unió horitzontal registre per registre
    - Un únic registre d'una base de dades correspon a un únic registre de l'altre base de dades
    - Exemple: Informació d'Edat i Sexe d'individus
  - Unió d'un registre a més d'un registres
    - Unint una observació d'una base de dades a múltiples observacions d0una altra base de dades
    - Exemple: Província i Comunitat Autònoma
  - Les dades cal ordenar-les, mitjançant el PROC SORT, prèviament a la unió de les bases de dades

COMBINACIÓ DE BASES DE DADES DE SAS (3)

- Unió registre per registre
  - Generalment, necessitarem al menys una variable en comú en les bases de dades a unir
  - En l'exemple, un ID de l'individu pel que coneixem Edat i Sexe
  - No necessitem variables comunes si totes les bases de dades a unir es troben en el mateix ordre

  - Forma General:

```
DATA output_data_set;
  MERGE input_data_set1 input_data_set2;
   BY variable1 variable2;
RUN;
```

```
DATA pares;
  INPUT famid nom $ renda ;
CARDS;
2 Art  22000
1 Bill 30000
3 Paul 25000
;
RUN;
DATA rendfam;
  INPUT famid rfam96 rfam97 rfam98 ;
CARDS;
3 75000 76000 77000
1 40000 40500 41000
2 45000 45400 45800
;
PROC SORT DATA=pares OUT=pares2;
  BY famid;
RUN;
PROC SORT DATA=rendfam OUT=rendfam2;
  BY famid;
RUN;
DATA parfam ;
  MERGE pares2 rendfam2;
  BY famid;
RUN:
PROC PRINT DATA=dadfam;
RUN;
```

COMBINACIÓ DE BASES DE DADES DE SAS (4)

- Unió un registre per múltiples registres
  - Necessitem almenys una variable en comú en les bases de dades a unir
  - En l'exemple, la Província ha d'estar en les dues bases de dades
  - Si les dues bases de dades tenen variables amb el mateix nom, les variables en la segona base de dades sobreescriuran les variables de la primera

  - Forma General:

```
DATA output_data_set;
 MERGE Data1 Data2 Data3;
    BY Variable1 Variable2;
RUN:
```

```
DATA pares;
  INPUT famid nom $ renda ;
CARDS;
2 Art  22000
1 Bill 30000
3 Paul 25000
;
RUN;

DATA fills;
  INPUT famid nomfill $ ordre edat pes sexe $ ;
CARDS;
1 Beth 1 9 60 f
1 Bob  2 6 40 m
1 Barb 3 3 20 f
2 Andy 1 8 80 m
2 Al   2 6 50 m
2 Ann  3 2 20 f
3 Pete 1 6 60 m
3 Pam  2 4 40 f
3 Phil 3 2 20 m
;
RUN;
```

```
PROC SORT DATA=pares OUT=pares2;
  BY famid;
RUN;
PROC SORT DATA=fills OUT=fills2;
  BY famid;
RUN;

DATA parfill;
  MERGE pares2 fills2;
  BY famid;
RUN;

PROC PRINT DATA=parfill;
RUN;
```

TRANSPOSICIÓ DE BASES DE DADES DE SAS

- El PROC TRANSPOSE transposa una base de dades SAS canviant observacions per variables i viceversa

  - PROC TRANSPOSE crea una variable en la nova base de dades transposada que conté els noms de les variables de la base de dades original.

  - Per defecte només es transposaran les variables numèriques. Si es volen transposar també les variables caràcter cal especificar-les totes en la comanda VAR.

  - Forma General:

```
PROC TRANSPOSE DATA=origen OUT=transp;
 VAR variablecaracter;
RUN:
```

```
data ampla1;
  input famid faminc96 faminc97 faminc98 ;
cards;
1 40000 40500 41000
2 45000 45400 45800
3 75000 76000 77000
;
run;

proc transpose data=ampla1 out=llarga1;
run;

proc print data=llarga1;
run;
```

```
data ampla1;
  input famid faminc96 faminc97 faminc98 ;
cards;
1 40000 40500 41000
2 45000 45400 45800
3 75000 76000 77000
;
run;

proc transpose data=ampla1 out=llarga1;
 BY famid;
run;

proc print data=llarga1;
run;
```

```
data ampla1;
  input famid $ faminc96 faminc97 faminc98 ;
cards;
1 40000 40500 41000
2 45000 45400 45800
3 75000 76000 77000
;
run;

proc transpose data=ampla1 out=llarga1;
run;

proc print data=llarga1;
run;

proc transpose data=ampla1 out=llarga2;
 VAR famid;
run;
```

PROCEDIMENTS BÀSICS

El sistema SAS està composat de diversos mòduls específics per realitzar diferents anàlisis estadístiques i d'investigació operativa:

SAS/BASE
SAS/STAT
SAS/ETS (Sèries Temporals)
SAS/QC (Control de qualitat)
SAS/OR (Investigació Operativa)
etc...

Cada mòdul està format per diferents procediments (PROC's)

PROCEDIMENTS BÀSICS (2)

Tots els PROC's tenen comandes en comú i comandes específiques. En general, les comandes associades a la majoria de procediments són:

```
PROC NAME  DATA=SAS-data-set <others options>;
   BY variables ; (És necessari ordenar la base de dades)
   ID variables ;
   VAR variables/<options> ;
   WEIGHT variable ;
   OUTPUT <OUT=SAS-data-set> <keyword1=names
   ...keywordk=names>;
```

SAS/BASE: PROC PRINT

- PROC PRINT s'usa per llistar les dades en la finestra Output
- Per defecte, lista totes les observacions i variables de la base de dades SAS
- Sintaxi General:

```
PROC PRINT DATA=input_data_set <options>
                    <optional SAS statements>;
```

- Algunes Opcions
  - `input_data_set (obs=n)` Especifica el nombre d'observacions
  - `NOOBS` Suprimeix el número d'observació
  - `LABEL` Llista les etiquetes de les variables en comptes del seu nom

SAS/BASE: PROC PRINT

- `Optional SAS statements`
  - `BY variable1 variable2 variable3;`
    - Inicia una nova secció de l'output per a cada nou valor de les variables declarades a la comanda BY
  - `ID variable1 variable2 variable3;`
    - Llista les variables declarades a la comanda ID variables a l'esquerra en lloc del nombre de cada observació
  - `SUM variable1 variable2 variable3;`
    - Llista al final la suma dels valors de les variables indicades
  - `VAR variable1 variable2 variable3;`
    - Llista només les variables indicades

SAS/BASE: PROC PLOT O GPLOT

- PROC PLOT s'utilitza per crear gràfics senzills de les dades
- PROC GPLOT o PROC SGPLOT s'utilitzen per gràfics més sofisticats
- Sintaxi General:

```
PROC PLOT DATA=input_data_set;
PLOT vertical_variable*horizontal_variable/<options>;
RUN;
```

- Per defecte, SAS usa lletres per marcar els punts en els gràfics
  - A per una única observació, B per dues observacions en el mateix punt, etc.
- Per especificar un caràcter diferent que representi un punt
  - `PLOT vertical_variable * horizontal variable = ' * ';`

SAS/BASE: PROC PLOT O GPLOT

- Per especificar una tercera variable els valors de la qual marquin els punts
  - `PLOT vertical_var * horizontal_var = third_var;`

- Per representar més d'una variable en l'eix vertical
  - `PLOT vertical_var1 * horizontal_var= '2'`
    `vertical_var2 * horizontal_var= '1' / OVERLAY;`

SAS/BASE: PROC UNIVARIATE

- PROC UNIVARIATE : analitza la distribució d'unes dades
- Genera estadístics resum per una variable
  - Inclou mitjana, mediana, moda, desviació stàndard, asimetria, curtosi, quantils, etc.

- Sintaxi General:

```
PROC UNIVARIATE DATA=input_data_set <options>;
            VAR variable1 variable2 variable3;
            RUN ;
```

- Si no s'especifica la comanda VAR, aleshores s'obtenen els estadístics resum per totes les variables numèriques de la base de dades.

- `<options>` inclou:
  - `PLOT` genera Stem-and-leaf plot, Box plot, i Normal probability plot
  - `NORMAL` genera contrastos de Normalitat

SAS/BASE: PROC UNIVARIATE

```
PROC UNIVARIATE <options> ;
    BY variables ;
    CDFPLOT <variables> < / options> ;
    CLASS variable-1 <(v-options)> <variable-2 <(v-
    options)>> </ KEYLEVEL= value1 | ( value1 value2 )> ;
    FREQ variable ;
    HISTOGRAM <variables> < / options> ;
    ID variables ;
    INSET keyword-list </ options> ;
    OUTPUT <OUT=SAS-data-set> <keyword1=names
    ...keywordk=names> <percentile-options> ;
    PPPLOT <variables> < / options> ;
    PROBPLOT <variables> < / options> ;
    QQPLOT <variables> < / options> ;
    VAR variables ;
    WEIGHT variable ;
```

SAS/BASE: PROC MEANS

- Semblant a PROC UNIVARIATE
- Sintaxi General:

```
PROC MEANS DATA=input_data_set options;
                <Optional SAS statements>;
              RUN;
```

- Sense `options` o sense `<optional SAS statements>`, el Proc Means llistarà el nombre de valors no-missing, la mitjana, desviació stàndard, minim i màxim per totes les variables numèriques

SAS/BASE: PROC MEANS

- `Options`
  - Estadístics disponibles

CLM	Two-Sided Confidence Limits	**RANGE**	Range
CSS	Corrected Sum of Squares	**SKEWNESS**	Skewness
CV	Coefficient of Variation	**STDDEV**	Standard Deviation
KURTOSIS	Kurtosis	**STDERR**	Standard Error of Mean
LCLM	Lower Confidence Limit	**SUM**	Sum
MAX	Maximum Value	**SUMWGT**	Sum of Weight Variables
MEAN	Mean	**UCLM**	Upper Confidence Limit
MIN	Minimum Value	**USS**	Uncorrected Sum of Squares
N	Number Non-missing Values	**VAR**	Variance
NMISS	Number Missing Values	**PROBT**	Probability for Student's t
MEDIAN (or P50)	Median	**T**	Student's t
Q1 (P25)	25% Quantile	**Q3 (P75)**	75% Quantile
P1	1% Quantile	**P5**	5% Quantile
P10	10% Quantile	**P90**	90% Quantile
P95	95% Quantile	**P99**	99% Quantile

SAS/BASE: PROC MEANS

- `Optional SAS Statements`
  - `VAR Variable1 Variable2;`
    - Especifica les variables numèriques per les què es calcularan els estadístics
  - `BY Variable1 Variable2;`
    - Calcula estadístics per cada combinació de BY variables
  - `Output out=output_data_set;`
    - Crea bases de dades SAS amb els estadístics per defecte

SAS/BASE: PROC MEANS

```
PROC MEANS <option(s)> <statistic-keyword(s)>;
BY <DESCENDING> variable-1 <... <DESCENDING> variable-
n><NOTSORTED>;
CLASS variable(s) </ option(s)>;
FREQ variable;
ID variable(s);
OUTPUT <OUT=SAS-data-set> <output-statistic-specification(s)>
<id-group-specification(s)> <maximum-id-specification(s)>
<minimum-id-specification(s)> </ option(s)> ;
TYPES request(s);
VAR variable(s) </ option(s)>;
WEIGHT variable;
```

SAS/BASE: PROC FREQ

- PROC FREQ : genera taules de freqüències
- Sintaxi General :

```
PROC FREQ DATA=input_data_set;
  TABLE variable1*variable2*variable3/<options>;
  RUN;
```

- Options
  - LIST – no crea la taula sinó només la llista de valors i les seves freqüències
  - MISSING – els valors missing s'inclouran en les tabulacions
  - OUT=output_data_set – crea una base de dades amb les frequencies
  - NOPRINT – no llista les freqüències en la finestra output

- Es pot usar la comanda BY per obtenir freqüències per grups de valors o categories de la variable especificada

SAS/BASE: PROC FREQ

```
PROC FREQ <options> ;
BY variables ;
EXACT statistic-options </ computation-options> ;
OUTPUT <OUT=SAS-data-set> options ;
TABLES VAR1 </ options> ;
TABLES VAR1*VAR2 </ options> ;
TEST options ;
WEIGHT variable </ option> ;
```

(Taula de freqüències)

(Taula de contingència)

SAS/BASE: PROC CORR

- PROC CORR calcula correlacions entre variables
- Sintaxi General:

```
PROC CORR DATA=input_data_set <options>
        VAR Variable1 Variable2;
        With Variable3;
        RUN;
```

- Si VAR i WITH no s'especifiquen aleshores es calculen correlacions per a totes les variables dos a dos
- `options` inclou
  - SPEARMAN Rangs de Spearman
  - KENDALL Tau de Kendall
  - HOEFFDING D de Hoeffding

SAS/STAT: PROC REG

- PROC REG estima models de regressió lineal per mínims quadrats
- És un més dels molts PROC's de SAS per fer regressió
- Sintaxi General :

```
PROC REG DATA=input_data_set <options>
     MODEL dependent=indep1 indep2/<options>;
     <optional statements>;
RUN;
```

- PROC REG `options` inclou
  - `CORR` dóna la matriu de correlació entre les variables independents del model
- MODEL `options` inclou
  - `COLLIN` dóna eines per analitzar la multicol·linealitat
  - `STB` dóna les estimacions dels paràmetres estandarditzats
  - `CLB` calcula intervals de confiança per les estimacions dels paràmetres
- `Optional statements` inclou
  - `PLOT Dependent*Independent1` genera gràfics de les variables

SAS/STAT: PROC REG

```
PROC REG <options> ;
<label:>MODEL dependents=<regressors> </ options> ;
BY variables ;
FREQ variable ;
ID variables ;
VAR variables ;
WEIGHT variable ;
ADD variables ;
DELETE variables ;
RESTRICT equation, ...,equation ;
<label:>TEST <equation, ...,equation> </ options> ;
OUTPUT <OUT=SAS-data-set>< keyword=names> <...keyword=names> ;
PLOT <yvariable*xvariable> <=symbol> <...yvariable*xvariable>
<=symbol> </ options>;
```

SAS/STAT: PROC GENMOD

- PROC GENMOD estima models lineals en els què la variable dependent no és necessàriament normal
- Els models Logit, Probit i de Poisson són exemples de models lineals generalitzats
- Sintaxi General:

```
PROC GENMOD DATA=input_data_set;
CLASS independent1;
MODEL dependent = indep1 indep2/dist= <option> link=<option>;
run;
```

- `DIST =` especifica la distribució de la variable resposta
- `LINK=` especifica la funció d'enllaç pel predictor lineal
- Exemples

Regressió Logística: `DIST = binomial` `LINK = logit`

Regressió Poisson: `DIST = poisson` `LINK = log`

SAS/STAT: PROC GENMOD

Models lineals generalitzats

```
PROC GENMOD <options> ;
BY variables ;
CLASS variables ;
CONTRAST 'label' effect values <...effect values> / <options> ;
DEVIANCE variable = expression ;
FREQ | FREQUENCY variable ;
MODEL response = <effects > / < options> ;
OUTPUT <OUT=SAS-data-set><keyword=name...keyword=name> ;
WEIGHT | SCWGT variable ;
VARIANCE variable = expression ;
ZEROMODEL <effects > / < options> ;
```

SAS/STAT: PROC GENMOD

Models lineals generalitzats

```
PROC GENMOD <options> ;
MODEL dependents=independents /link=  ;
```

LINK=	Link Function
CUMCLL CCLL	cumulative complementary log-log
CUMLOGIT CLOGIT	cumulative logit
CUMPROBIT CPROBIT	cumulative probit
CLOGLOG CLL	complementary log-log
IDENTITY ID	identity
LOG	log
LOGIT	logit
PROBIT	probit
POWER(*number*) | POW(*number*)	power with = *number*

SAS/STAT: PROC GENMOD

Models lineals generalitzats

```
PROC GENMOD <options> ;
MODEL dependents=independents /dist=  ;
```

DIST=	Distribution	Default Link Function
BINOMIAL | BIN | B	binomial	logit
GAMMA | GAM | G	gamma	inverse (power(1))
GEOMETRIC | GEOM	geometric	log
IGAUSSIAN | IG	inverse Gaussian	inverse squared (power(2))
MULTINOMIAL | MULT	multinomial	cumulative logit
NEGBIN | NB	negative binomial	log
NORMAL | NOR | N	normal	identity
POISSON | POI | P	Poisson	log

SAS/STAT: PROC LOGISTIC

Models logit i probit

```
PROC LOGISTIC <options> ;
BY variables ;
CLASS variable <(options)><variable <(options)>></ options> ;
CONTRAST 'label' effect values<, effect values,></ options> ;
FREQ variable ;
<label:> MODEL events/trials=<effects></ options> ;
<label:> MODEL variable <(variable_options)>=<effects></
options> ;
OUTPUT <OUT=SAS-data-set><keyword=name <keyword=name>></
option> ;
<label:> TEST equation1 <,equation2,></ option> ;
WEIGHT variable </ option> ;
```

SAS/STAT: PROC LOGISTIC

Models logit i probit

```
PROC LOGISTIC <options> ;
<label:> MODEL variable / link=probit ;
<label:> MODEL variable / link=logit ;
```

SAS / IML

Què és SAS IML?

- Interactive
- Matrix
- Language

Definint Matriu A

```
proc iml;
  reset print;

 A = {1 3 5, 4 4 1, 2 2 6 };
                  /*  Defineix  A matriu 3 x 3  */
```

```
A        3 rows       3 cols     (numeric)

1           3           5
4           4           1
2           2           6
```

Definint Matriu B

B = {**1 3 4**, **3 5 2**, **4 2 1** };

/* Defineix B matriu 3 x3 simètrica I definida positiva*/

```
 B       3 rows       3 cols    (numeric)

1           3           4
3           5           2
4           2           1
```

Definint Matriu C

C = {**2 1 1**, **3 4 6**};

/* C és una matriu 2 x 3 */

```
C       2 rows      3 cols    (numeric)

2          1          1
3          4          6
```

Definint Matriu D

D = {**2 2**, **4 5**, **5 1**};

/* D és una matriu 3 x 2 */

```
 D       3 rows       2 cols     (numeric)

2           2
4           5
5           1
```

Definint Matriu X

X = {**1 1 0**, **1 1 0**, **1 1 0**, **1 0 1**, **1 0 1**, **1 0 1**};

```
 X       6 rows       3 cols    (numeric)

1           1           0
1           1           0
1           1           0
1           0           1
1           0           1
1           0           1
```

Càlcul de la Inversa d' A

inversaA = inv(A); /* calcula la inversa d' A*/

```
INVERSEA      3 rows      3 cols    (numeric)

-0.5  0.1818182  0.3863636
 0.5  0.0909091 -0.431818
 0   -0.090909   0.1818182
```

Trasposta de C

```
   transposeC      =      t(C);    /* o bé C`  */
/* calcula la trasposta de C */


 TRANSPOSEC       3 rows       2 cols    (numeric)

2           3
1           4
1           6
```

Suma de matrius: A+B

AplusB = A + B;

/* Suma 2 matrius, d'iguals dimensions */

```
 APLUSB       3 rows       3 cols     (numeric)

2            6            9
7            9            3
6            4            7
```

Producte matricial: A*B

```
   AtimesB        =      A * B;
/*  Producte Matricial   */
```

```
 ATIMESB        3 rows       3 cols     (numeric)
30          28          15
20          34          25
32          28          18
```

Pregunta:

- Si volem multiplicar dues matrius element a element, en lloc de fer el producte matricial?

$$\begin{bmatrix} 3 & 1 \\ 4 & 2 \end{bmatrix} ? \begin{bmatrix} 2 & 2 \\ 4 & 4 \end{bmatrix} = \begin{bmatrix} 6 & 2 \\ 16 & 8 \end{bmatrix}$$

- Quin és l'operador?
- Resposta: #
- Comparem amb *, que hem vist abans

$$\begin{bmatrix} 3 & 1 \\ 4 & 2 \end{bmatrix} * \begin{bmatrix} 2 & 2 \\ 4 & 4 \end{bmatrix} = \begin{bmatrix} 10 & 10 \\ 16 & 16 \end{bmatrix}$$

Operadors de Reducció

Asumrows = A[,+];

/* Operador de reducció, suma de files */

```
ASUMROWS      3 rows      1 col     (numeric)

9
9
10
```

Operadors de Reducció

Asumcols = A[+,];

/* Suma de columnes */

```
ASUMCOLS        1 row        3 cols     (numeric)

                7            9          12
```

Pregunta: Com obtenir el producte de files o columnes?

- prodC = C[op,]

$$C = \begin{bmatrix} 2 & 1 & 1 \\ 3 & 4 & 6 \end{bmatrix}$$

$$prodc = \begin{bmatrix} 6 & 4 & 6 \end{bmatrix}$$

- Quin és op?
- Resposta: #

Pregunta

- Suposem que volem sumar tots els elements de la matriu A. Com ho fem?

- Resposta: sumA = sum(A);
- Resposta : sumA = A[+,][,+];
- Resposta : sumA = A[,+][+,];
- Resposta : sumA = A[+,+];

Concatenant Matrius

AnextB = A||B;
/* Posa 2 matrius costat per costat */

```
ANEXTB       3 rows       6 cols    (numeric)

1            3            5            1            3            4
4            4            1            3            5            2
2            2            6            4            2            1
```

Concatenant Matrius

AtopB = A//B

/* Posa 2 matrius una a sobre de l'altra */

```
ATOPB        6 rows        3 cols     (numeric)

1             3             5
4             4             1
2             2             6
1             3             4
3             5             2
4             2             1
```

Diagonalització

diagA = diag(A);

/* Canvia els elements de fora de la diagonal principal a 0, mantenint els de la diagonal principal */

```
 DIAGA       3 rows       3 cols     (numeric)

1            0            0
0            4            0
0            0            6
```

Diagonalització

vdiagA = vecdiag(A);

/* Col·loca els elements de la diagonal principal de A, en un vector columna */

```
 VDIAGA       3 rows       1 col      (numeric)

1
4
6
```

Funcions

logA = log(A);

/* Logaritme de cada element */

```
 LOGA       3 rows       3 cols     (numeric)

            0 1.0986123 1.6094379
    1.3862944 1.3862944         0
    0.6931472 0.6931472 1.7917595
```

Una altra funció

```
ssgvdiagA =     ssq(vdiagA);
/*  Eleva al quadrat cada element de vdiagA, i
    després els suma tots  */

 SSGVDIAGA       1 row        1 col     (numeric)

53
```

Elements de Programació

A més de les funcions i operadors que hem vist abans, també existeixen comandes de programació:

- If/Then
- Do
- Goto
- Mòduls

Mòduls

- Els Mòduls són semblants a subrutines, o funcions; poden ser definits a qualsevol lloc del programa IML, i ser reutilitzats més tard.

Mòduls

```
start TransMatrix(D);

Dcol      = ncol(D);              /*  Nombre de columnes a D */
Drow    = nrow(D);            /* Nombre de files a D       */

Dtranspose_temp  = shape(.,Dcol, Drow);
do i = 1 to Dcol;
        do j = 1 to Drow;
                Dtranspose_temp[i,j] = D[j,i];      /* trasposta de matriu D */
        end;
end;
return (Dtranspose_temp);
finish TransMatrix;

Dtranspose = TransMatrix(X);
print  Dtranspose;
quit;
```

Accedir a una base de dades SAS amb IML convertint variables en matrius

```
proc iml;
use dset1;                    /* base de dades a llegir */
read all var{x} into xobs;    /* la X de la base de dades en la matriu XOBS */
print xobs;
quit;
```

```
XOBS      10 rows       1 col      (numeric)

   550
   200
   280
   340
   410
   160
   380
   510
   510
   475
```

Crear una base de dades des de IML convertint matrius en variables

```
proc iml;
.....
 create dset2 from XOBS [colname={‘x2'}];
   append from XOBS;
   close dset2;
quit;
proc print data=dset2;
run;
```