SAS®
Initiation et manipulation de données

Nicolas Poulin

poulin@math.unistra.fr

Créer plusieurs tables SAS

- Il est possible de créer plusieurs tables SAS dans une même étape DATA.
- Il faut énumérer le nom des tables qui seront créées dans l'instruction DATA.

$$\texttt{DATA Table1 Table2} \cdots \texttt{ Tablen;}$$

- L'instruction OUTPUT est utilisée pour diriger les observations vers la table souhaitée.

poulin@math.unistra.fr

Remarques

- Si aucun nom de table n'est spécifié dans l'instruction DATA, SAS créera des tables avec les noms data1, data2,..., dans la bibliothèque WORK.
- Les tables SAS nommées dans l'instruction OUTPUT doivent avoir été déclarées dans l'instruction DATA.
- Une instruction OUTPUT sans argument écrit l'observation dans chacune des tables déclarées dans l'instruction DATA.

poulin@math.unistra.fr

Rappel : IF-THEN/ELSE

IF condition THEN expression1;

ELSE expression2;

poulin@math.unistra.fr

PROC IMPORT : exemple

```
PROC IMPORT OUT=work.mais
            DATAFILE="folders/myfolders/UEdata/mais.xlsx"
            DBMS= xlsx;
     GETNAMES=YES;
RUN;
```


poulin@math.unistra.fr

Exemple : Créer plusieurs tables SAS

```
DATA Est Ouest Nord Sud;

    SET work.mais;

    IF Parcelle='Est' THEN OUTPUT Est;

    ELSE IF Parcelle='Ouest' THEN OUTPUT Ouest;

    ELSE IF Parcelle='Nord' THEN OUTPUT Nord;

    ELSE OUTPUT Sud;

RUN;
```


poulin@math.unistra.fr

Imprimer plusieurs tables SAS

- La PROC PRINT ne peut imprimer qu'une seule table.
- Il faut donc une PROC PRINT par table

```
TITLE'Parcelle Est';
PROC PRINT DATA=Est;
RUN;

...

TITLE'Parcelle Sud';
PROC PRINT DATA=Sud;
RUN;
```


poulin@math.unistra.fr

Alternative : SELECT

- Un groupe SELECT peut être utilisée comme forme alternative au IF/THEN/ELSE.

- Structure d'un groupe SELECT :

```
SELECT expression;
        WHEN (condition1) instruction1;
        ...
        WHEN (conditionn) instructionn;
        OTHERWISE instruction;
END;
```

Remarque

L'expression de SELECT doit être simple. Généralement, elle se limite à un nom de variable

poulin@math.unistra.fr

Exemple : groupe SELECT

```
DATA Est Ouest Nord Sud;
     SET work.mais;
     SELECT (Parcelle);
            WHEN ('Est') OUTPUT Est;
            WHEN ('Ouest') OUTPUT Ouest;
            WHEN ('Nord') OUTPUT Nord;
            OTHERWISE OUTPUT Sud;
     END;
RUN;
```


poulin@math.unistra.fr

Exemple : groupe SELECT

L'expression du SELECT peut être omise :

```
DATA Est Ouest Nord Sud;
        SET work.mais;
        SELECT;
                WHEN (Parcelle='Est') OUTPUT Est;
                WHEN (Parcelle='Ouest') OUTPUT Ouest;
                WHEN (Parcelle='Nord') OUTPUT Nord;
                OTHERWISE OUTPUT Sud;
        END;
RUN;
```


poulin@math.unistra.fr

Remarques

- Les instructions WHEN sont traitées de haut en bas. Pour un grand jeu de données, il est donc plus efficace de vérifier les valeurs par ordre de fréquence décroissante.
- Le résultat est exactement le même que si on avait utilisé IF/THEN/ELSE.
- L'instruction OTHERWISE est facultative.
- Omettre l'instruction OTHERWISE générera l'affichage d'une erreur lorsqu'aucune des conditions WHEN n'est vérifiée.
- Pour éviter ce message d'erreur :

  OTHERWISE;

poulin@math.unistra.fr

Exemple : groupe SELECT

Des valeurs multiples peuvent être énumérées dans une expression WHEN :

```
DATA EO NS;
        SET work.mais;
        SELECT (Parcelle);
                WHEN ('Est','Ouest') OUTPUT EO;
                OTHERWISE OUTPUT NS;
        END;
RUN;
```


poulin@math.unistra.fr

Exemple : groupe SELECT

Un groupe DO peut être utilisé pour exécuter des instructions multiples :

```
DATA EO NS;
        SET work.mais;
        SELECT (Parcelle);
                WHEN ('Est','Ouest') DO;
                        Parcelle='EO';
                        OUTPUT EO;
                END;
                OTHERWISE DO;
                        Parcelle='NS';
                        OUTPUT NS;
                END;
        END;
RUN;
```


poulin@math.unistra.fr

Contrôler quelles observations sont lues

- Par défaut, SAS traite toutes les observations du jeu de données, de la première à la dernière ligne.
- Il est possible de spécifier à partir de quelle observation l'utilisateur veut que la lecture commence grâce à l'option FIRSTOBS.
- Il est possible de spécifier combien d'observation l'utilisateur veut traiter grâce à l'option OBS.
- Les options FIRSTOBS et OBS ne peuvent être utilisées que sur la table d'entrée. Elles ne peuvent être utilisées sur les tables de sortie.

poulin@math.unistra.fr

Exemple : FIRSTOBS et OBS

```
DATA work.mais1;
        SET work.mais (FIRSTOBS=2 OBS=35);
RUN;


PROC PRINT DATA=work.mais1;
RUN;
```


poulin@math.unistra.fr

PROC PRINT

- Les options FIRSTOBS et OBS peuvent être utilisées dans une PROC PRINT.

```
PROC PRINT DATA=work.mais (FIRSTOBS=2 OBS=35);
        VAR Hauteur Masse Parcelle;
RUN;
```

- Une instruction WHERE peut aussi être utilisée.

```
PROC PRINT DATA=work.mais (FIRSTOBS=2 OBS=35);
        WHERE Parcelle='Nord';
        VAR Hauteur Masse Parcelle;
RUN;
```


poulin@math.unistra.fr

Instruction BY dans une étape DATA

- L'instruction BY peut être utilisée dans un certain nombre de procédures.
- Il est possible d'utiliser l'instruction BY dans une étape DATA afin de traiter des données à l'intérieur de groupes.
- Pour utiliser l'instruction BY dans une étape DATA, il faut que la table ait été préalablement triée par rapport à la variable utilisée dans le BY.

Remarque

Plusieurs variables peuvent être utilisées dans une instruction BY.

poulin@math.unistra.fr

Instruction BY dans une étape DATA

- Si un BY est utilisé sur une variable nommée *mavar*, alors SAS va créer 2 variables temporaires : First.mavar et Last.mavar
- La variable First.mavar a une valeur de 1 pour la première observation du groupe BY, 0 sinon.
- La variable Last.mavar a une valeur de 1 pour la dernière observation du groupe BY, 0 sinon.

poulin@math.unistra.fr

Variables First. et Last.

Parcelle	First.Parcelle	Last.Parcelle
Est		
Est		
...	...	...
Est		
Nord		
...	...	...

poulin@math.unistra.fr

Variables First. et Last.

Parcelle	First.Parcelle	Last.Parcelle
Est	1	0
Est	0	0
...	...	...
Est	0	1
Nord	1	0
...	...	...

poulin@math.unistra.fr

Exemple : Instruction BY dans une étape DATA

- On souhaite obtenir la masse totale des pieds de maïs pour chaque parcelle.
- On va utiliser l'instruction BY sur la variable Parcelle dans une étape DATA.
- Il faut commencer par trier le tableau par rapport à la variable Parcelle.

```
PROC SORT DATA=work.mais ;

    BY Parcelle;

RUN;
```


poulin@math.unistra.fr

Exemple : Instruction BY dans une étape DATA

- On va créer une table SAS nommée MassTot qui ne contiendra que 2 variables : la parcelle et la masse totale (masse_totale).

```
DATA MassTot (KEEP=Parcelle masse_totale) ;
        SET work.mais;
        BY Parcelle;
...
RUN;
```


poulin@math.unistra.fr

Exemple : Instruction BY dans une étape DATA

- On va définir la masse totale (variable masse_totale) que l'on initialise à 0 au début de chaque groupe BY .

```
DATA MassTot (KEEP=Parcelle masse_totale) ;
        SET work.mais;
        BY Parcelle;
        IF First.Parcelle=1 THEN masse_totale=0;
...
RUN;
```


poulin@math.unistra.fr

Exemple : Instruction BY dans une étape DATA

- Lorsque des variables booléennes sont utilisées dans un IF, il n'est pas nécessaire de préciser la valeur : 1 correspond au succès, 0 à l'échec.

```
DATA MassTot (KEEP=Parcelle masse_totale) ;
        SET work.mais;
        BY Parcelle;
        IF First.Parcelle THEN masse_totale=0;
...
RUN;
```


poulin@math.unistra.fr

Exemple : Instruction BY dans une étape DATA

- On rajoute à chaque ligne, la masse de l'observation en cours à la variable masse_totale.

```
DATA MassTot (KEEP=Parcelle masse_totale) ;
     SET work.mais;
     BY Parcelle;
     IF First.Parcelle THEN masse_totale=0;
     masse_totale+Masse;
...
RUN;
```

Remarque

Il n'y a pas besoin de mettre "masse_totale=masse_totale+Masse;" ce qui générerait en plus des données manquantes. Il n'est pas nom plus possible d'utiliser la fonction SUM.

poulin@math.unistra.fr

Exemple : Instruction BY dans une étape DATA

- On veut garder uniquement la dernière ligne pour chaque Parcelle car c'est sur cette ligne que se trouve le total final pour la Parcelle en cours.
- On va utiliser un IF pour trouver cette dernière ligne.

```
DATA MassTot (KEEP=Parcelle masse_totale) ;
     SET work.mais;
     BY Parcelle;
     IF First.Parcelle THEN masse_totale=0;
     masse_totale+Masse;
     IF Last.Parcelle THEN OUTPUT;
RUN;
```


poulin@math.unistra.fr

IF sans THEN

- L'instruction de sous-ensemble IF définit une condition que l'observation doit satisfaire pour être traitée.
- Elle est généralement suivie d'un THEN. après lequel on stipule une instruction.
- La forme générale du IF ne nécessite pas obligatoirement la présence d'un THEN.

poulin@math.unistra.fr

IF sans THEN

IF expression;

- Si l'expression est vraie, l'instruction est de poursuivre le traitement de l'observation courante.
- Si l'expression est fausse, l'instruction est de retourner au début de l'étape DATA, c'est-à-dire de traiter l'observation suivante.

poulin@math.unistra.fr

Exemple : Instruction BY dans une étape DATA

- On peut donc utiliser un IF sans mettre d'instruction.

```
DATA MassTot (KEEP=Parcelle masse_totale) ;
     SET work.mais;
     BY Parcelle;
     IF First.Parcelle THEN masse_totale=0;
     masse_totale+Masse;
     IF Last.Parcelle;
RUN;

PROC PRINT DATA=MassTot NOOBS;
RUN;
```


poulin@math.unistra.fr

Instruction BY sur plusieurs variables

- On souhaite obtenir la masse totale des pieds de maïs pour chaque parcelle et statut d'attaque ainsi que le nombre de pieds de maïs dans chaque combinaison.
- On va utiliser l'instruction BY sur les variables Parcelle et Attaque dans une étape DATA.
- Il faut commencer par trier le tableau par rapport aux variables Parcelle et Attaque.

```
PROC SORT DATA=work.mais ;
        BY Parcelle Attaque;
RUN;
```


poulin@math.unistra.fr

Exemple : Instruction BY sur plusieurs variables

- On va créer une table SAS nommée MassTot2 qui ne contiendra que 4 variables :
  - Parcelle
  - Attaque
  - masse_totale
  - Nb_pieds : le nombre de pieds de maïs

```
DATA MassTot2 (KEEP=Parcelle Attaque masse_totale Nb_pieds) ;
      SET work.mais;
      BY Parcelle Attaque;
...
RUN;
```


poulin@math.unistra.fr

Variables First. et Last.

- Comme un BY est utilisé sur les variables Parcelle et Attaque, SAS va créer 4 variables temporaires :
  - First.Parcelle
  - First.Attaque
  - Last.Parcelle
  - Last.Attaque

- Comment SAS définit-il les valeurs pour ces 4 variables?

Remarque

Dans les slides suivants ces noms sont raccourcis uniquement dans un soucis de présentation.

poulin@math.unistra.fr

Variables First. et Last.

Parcelle	Attaque	First.Par	First.Att	Last.Par	Last.Att
Est	Non				
Est	Non				
...	...	...	...	...	...
Est	Non				
Est	Oui				
Est	Oui				
...	...	...	...	...	...
Est	Oui				
Nord	Non				
...	...	...	...	...	...

poulin@math.unistra.fr

Variables First. et Last.

Parcelle	Attaque	First.Par	First.Att	Last.Par	Last.Att
Est	Non	1	1	0	0
Est	Non	0	0	0	0
...	...	...	...	...	...
Est	Non	0	0	0	1
Est	Oui	0	1	0	0
Est	Oui	0	0	0	0
...	...	...	...	...	...
Est	Oui	0	0	1	1
Nord	Non	1	1	0	0
...	...	...	...	...	...

poulin@math.unistra.fr

Variables First. et Last.

- Comme un BY est utilisé sur les variables Parcelle et Attaque, dans cet ordre, SAS considère que :
  - Parcelle est la variable primaire
  - Attaque est la variable secondaire

- First.primaire=1 $\Longrightarrow$ First.secondaire=1

- Last.primaire=1 $\Longrightarrow$ Last.secondaire=1

poulin@math.unistra.fr

Exemple : Instruction BY sur plusieurs variables

- Compléter le code suivant pour obtenir un tableau comportant une ligne par combinaison de Parcelle et Attaque

```
DATA MassTot2 (KEEP=Parcelle Attaque masse_totale Nb_pieds) ;
        SET work.mais;
        BY Parcelle Attaque;
...
RUN;
```


poulin@math.unistra.fr

Exemple : Instruction BY sur plusieurs variables

```
DATA MassTot2 (KEEP=Parcelle Attaque masse_totale Nb_pieds) ;
        SET work.mais;
        BY Parcelle Attaque;
        IF First.Attaque THEN DO;
            masse_totale=0;
            Nb_pieds=0;
        END;
        masse_totale+Masse;
        Nb_pieds+1;
        IF Last.Attaque;
RUN;
```


poulin@math.unistra.fr