

Le LAG expliqué à ma fille

La fonction LAG est souvent présentée comme un cas de sorcellerie appliquée à l'étape DATA. Elle fonctionne de manière caractérielle, un peu à son humeur, et pas toujours de manière cohérente. Pourtant, de nombreuses personnes en donnent une définition simple : « la fonction qui renvoie la valeur de l'observation précédente ». Petite enquête au pays du LAG pour savoir si elle a vraiment mauvais caractère, ou si son comportement n'est pas abusivement simplifié par la définition précédente.

Que se passe-t-il dans une étape Data ?

Lors de la lecture de données dans une étape Data (que ce soit avec une instruction SET, MERGE ou INFILE), les informations qui composent les observations sont stockées de manière temporaire dans un espace appelé Vecteur de Travail (ou PDV dans les documents en anglais).

Le traitement des données s'effectue ainsi :

1. copie des informations lues dans le vecteur de travail
2. opérations (calculs) demandés (création de nouvelles variables, modifications de variables existantes)
3. écriture du contenu du vecteur de travail dans la table créée (éventuellement sous condition, en présence de WHERE ou de IF ... THEN OUTPUT¹)
4. suppression des informations contenues dans le vecteur de travail
5. retour à l'étape 1.

Etant donné qu'on ne lit qu'une seule observation à la fois, l'existence d'une fonction comme LAG est troublante.

Ce qui se passe quand on utilise la fonction LAG

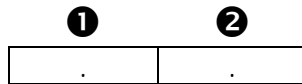
En réalité, la fonction LAG fait l'objet d'un traitement à part du vecteur de travail, sous la forme d'une file d'attente. C'est-à-dire qu'il y a, en plus du vecteur de travail, un espace mémoire comprenant une « case » de plus que la profondeur du LAG (soit 2 cases pour LAG, 3 cases pour LAG2, 4 pour LAG3, etc.). A chaque observation, la première fois que la fonction LAG est exécutée, la file se voit ajouter à sa *dernière* case une nouvelle valeur. Lors de cet ajout, pour des raisons de place, on perd le contenu de la première case (elle est poussée hors de la file). La nouvelle valeur ajoutée est la valeur courante, celle qui est stockée dans le vecteur de travail. La fonction LAG, de son côté, renvoie le contenu de la première case de la pile.

Déroulons le film du remplissage et du vidage de cette file d'attente, quand on lit la table SAS suivante :

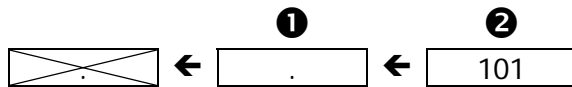
<u>_N_</u>	<u>X</u>
1	101
2	111
3	99

A l'initialisation, la file d'attente est vide.

¹ Qui peut s'abrégé en IF condition ; avec le même effet que IF condition THEN OUTPUT ; — si la condition est vérifiée, l'observation est copiée dans la table en sortie

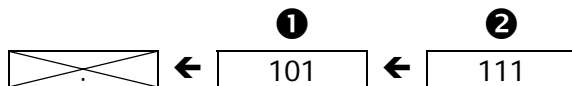


Puis le vecteur de travail charge la 1^e observation. Si on appelle la fonction LAG(X), la file d'attente devient :

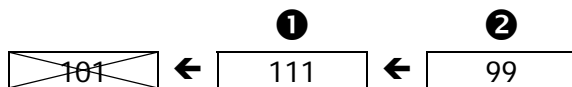


L'arrivée de la valeur courante en queue de file d'attente (position 2) décale les cases existantes, supprimant l'ancienne valeur 1. Le résultat de LAG(X) vaut ., le contenu de la case 1.

Après chargement de la 2^e observation, à l'exécution de la fonction LAG(X), on a à nouveau un décalage, qui introduit en position 2 la valeur courante de X, 111, et décale à la position 1 la valeur précédemment stockée dans la case 2. LAG(X) renvoie la valeur 101 contenue dans la case 1.



On observe encore le même décalage lorsqu'on exécute LAG(X) à la 3^e observation.



LAG(X) vaut 111.

Jusque là, tout est conforme à la définition donnée dans l'introduction : LAG(X) renvoie bien la valeur stockée à l'observation précédente. La compréhension de ce système de file d'attente ne fait que lever le mystère sur le mécanisme qui permet de récupérer une valeur provenant d'une autre observation que celle qui est actuellement chargée dans le vecteur de travail.

Cependant, le point essentiel dans la compréhension de la fonction LAG est que la file d'attente n'est pas alimentée automatiquement, mais seulement lors de l'exécution de la fonction LAG. On peut donc avoir en résultat une valeur qui n'est pas celle de l'observation immédiatement précédente si la fonction LAG est exécutée de manière conditionnelle, dans un IF ou un SELECT WHEN.

Par exemple :

```
DATA work.test ;
  SET sashelp.class ;
  valeurPrecedente = LAG(name) ;
  IF age=14 THEN valeurLAGdansIF = LAG(name) ;
RUN ;
PROC PRINT DATA=work.test NOOBS ;
  VAR name age valeurPrecedente valeurLAGdansIF ;
RUN ;
```

Dans ce cas, on constate bien que les valeurs remontées par LAG de la variable NAME ne sont pas celles de l'observation immédiatement précédente si la fonction est exécutée de manière conditionnelle. La valeur sera celle de la *dernière observation où la condition était vérifiée*.

Name	Age	valeurPrecedente	valeurLAGdansIF
Alfred	14		
Alice	13	Alfred	
Barbara	13	Alice	
Carol	14	Barbara	Alfred
Henry	14	Carol	Carol
James	12	Henry	
Jane	12	James	
Janet	15	Jane	
Jeffrey	13	Janet	
John	12	Jeffrey	
Joyce	11	John	
Judy	14	Joyce	Henry
Louise	12	Judy	
Mary	15	Louise	
Philip	16	Mary	
Robert	12	Philip	
Ronald	15	Robert	
Thomas	11	Ronald	
William	15	Thomas	

PS : à quoi le LAG est bon (et à quoi il ne l'est pas)

Il est assez commun d'utiliser la fonction LAG pour établir que la valeur d'une variable n'a pas évolué entre une observation et la suivante. Ce n'est pas du tout l'utilité prévue pour cette fonction : elle existe principalement pour l'implémentation de projections dans des séries chronologiques. Si on imagine qu'une variable a une observation par instant d'observation T, la fonction LAG permet de récupérer la valeur à T-1, LAG2 à T-2, etc.

Pour tester qu'une valeur ne change pas d'une observation à l'autre, il est préférable de trier la table au préalable et d'utiliser une instruction de lecture (SET ou MERGE) avec BY. On a ainsi des pseudo-variables FIRST et LAST qui indique si la valeur précédente, et la suivante, sont identiques à la valeur courante, y compris à l'intérieur de blocs définis par d'autres variables.