ENSAE – Tutoriel SAS 2015-2016

Martin Chevalier (INSEE)

SAS [1] (*Statistical analysis system*) est un logiciel de traitement statistique très utilisé, que ce soit dans les grandes entreprises, les laboratoires de recherche ou les instituts nationaux de statistique (en particulier à l'Insee).

Il s'agit d'un logiciel statistique généraliste, qui propose des fonctionnalités permettant d'importer et de mettre en forme des données de sources variées et en volume parfois importants, mais aussi de procéder à des analyses statistiques plus ou moins complexes.

Les points forts du logiciel SAS sont sa grande diffusion (beaucoup de statisticiens ont été formés avec SAS) et sa capacité à traiter des données relativement volumineuses (plusieurs millions de lignes). Son principal point faible est son caractère propriétaire, qui induit un coût important et limite les possibilités d'extension par les utilisateurs. Ses principaux concurrents sont **R** [2] (libre) et SPSS [3] (propriétaire).

L'objectif de ce court tutoriel est de vous amener à acquérir quelques points de repères et réflexes dans l'utilisation du logiciel SAS : interface du logiciel, utilisation de l'aide, import et export de données, manipulation de données avec l'étape `DATA`, principales procédures statistiques et graphiques.

Les questions les plus complexes sont précédées par le signe * et peuvent être passées dans un premier temps.

1. https://www.sas.com/fr_fr/home.html
2. https://cran.r-project.org/
3. http://www-01.ibm.com/software/analytics/spss/

1 Interface du logiciel

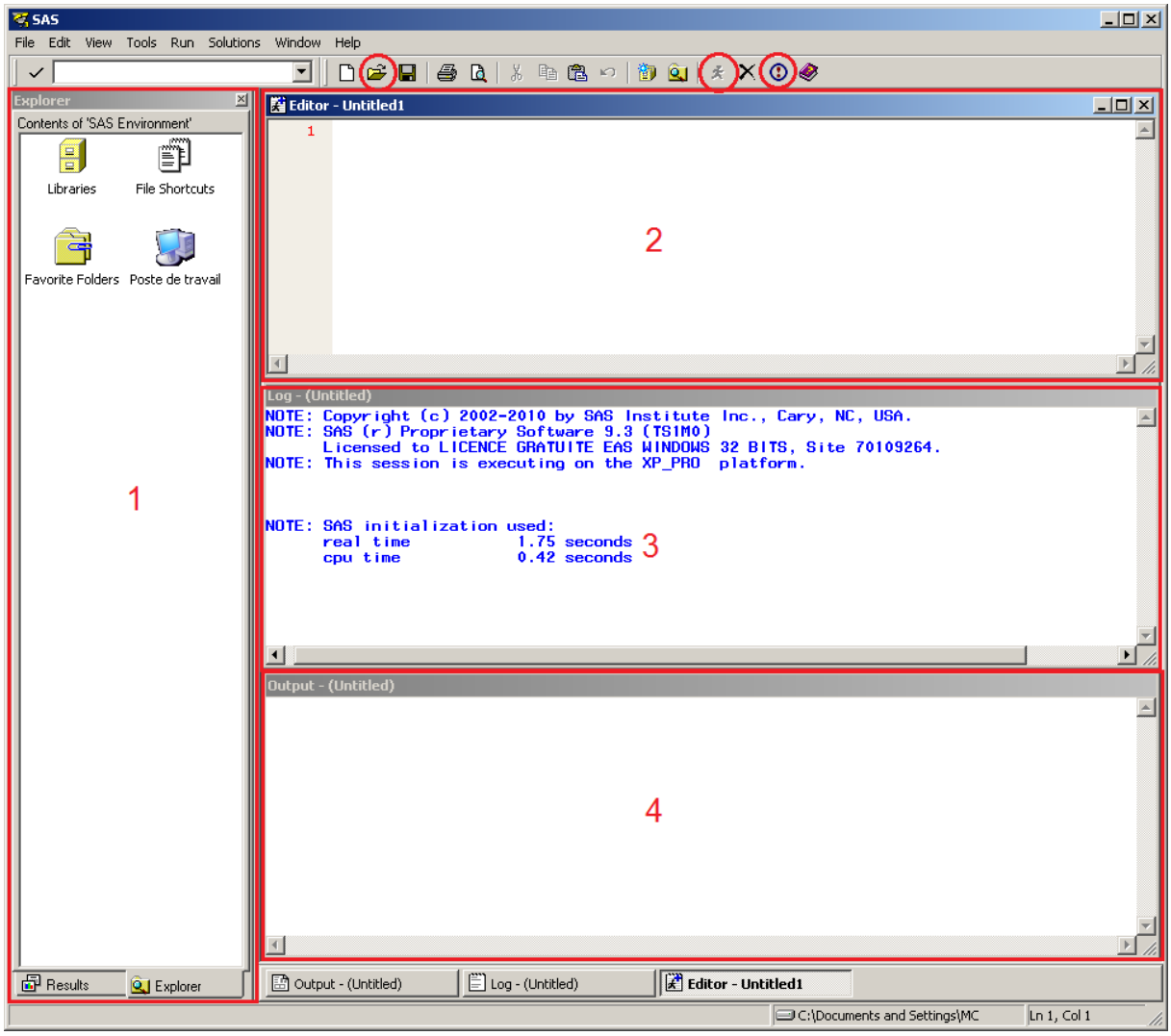
Les fenêtres et leurs interactions

Par défaut à l'ouverture de SAS, un explorateur et trois fenêtres sont ouvertes [4] (Figure 1) :

1. L'**explorateur de fichier** permet d'accéder aux fichiers de bases de données accessibles depuis l'ordinateur. La méthode privilégiée pour ce faire est la définition d'une bibliothèque (« *library* » en anglais) qui apparaîtra alors dans le sous-menu « *Libraries* » en haut à gauche.

   Conseil : Il arrive que l'explorateur de fichiers soit fermé par inadvertance : pour le rouvrir, cliquez sur Affichage > Contenu uniquement.

FIGURE 1 – Les fenêtres du logiciel SAS



2. L'**éditeur** est un fichier texte dans lequel les instructions à soumettre sont écrites et sauvegardées :

4. Souvent la fenêtre 4 est cachée derrière les fenêtres 2 et 3

- Pour ouvrir un code ou sauvegarder le contenu de l'éditeur, utilisez les boutons correspondant de la barre de menus ou les raccourcis clavier `Ctrl + O` et `Ctrl + S` respectivement.
- Pour soumettre une instruction, surlignez-la dans l'éditeur et cliquez sur le bouton « Soumettre » (entouré sur la Figure 1) ou sur la touche `F3`.
- Pour interrompre un traitement, cliquez sur le bouton interrompre (entouré sur la Figure 1).

3. Le **journal** (*Log*) recense toutes les instructions soumises depuis le début de la session et les réponses du logiciel :
   - **Les instructions soumises s'affichent en noir ;**
   - **Les notes s'affichent en bleu** : ces messages indiquent que les traitements se sont déroulés comme prévu ainsi que leur durée d'exécution ;
   - **Les avertissements s'affichent en vert** : ces messages attirent l'attention des utilisateurs sur des points susceptibles d'affecter les résultats sans (trop) perturber l'exécution des instructions.
   - **Les erreurs s'affichent en rouge** : ces messages indiquent que des erreurs ont été rencontrées et qu'une ou plusieurs instructions n'ont pas pu être exécutées.

   Il est également possible d'afficher soi-même des messages dans le journal, par exemple pour connaître l'avancement d'un programme long (boucles).

   Conseil : Dans certaines situations on peut souhaiter supprimer l'affichage des notes ou du code des instructions dans le journal. Pour ce faire, on peut utiliser l'instruction : `OPTIONS NONOTES NOSOURCE;`. Pour réactiver l'affichage des notes et du code des instructions, il suffit d'utiliser `OPTIONS NOTES SOURCE;`.

4. L'**afficheur de résultat** (*Output*) centralise les résultats des procédures statistiques mises en œuvre dans la session. Pour naviguer dans les résultats, vous pouvez utiliser :
   - l'ascenseur à droite de la fenêtre ;
   - les flèches du clavier ;
   - l'explorateur de résultats, qui est le premier onglet du panneau 1.

   Conseil : Pour vider le contenu de la fenêtre de résultats (par exemple en début de code), il suffit d'utiliser l'instruction `DM "clear output";`. Pour vider simultanément les fenêtres de résultat et d'output, utiliser `DM "clear log; clear output";`.

Question 1 Saisissez dans l'éditeur le code :

```
%PUT Hello world!;
```

et soumettez-le. Dans quelle fenêtre le texte s'est-il affiché ? À quoi sert la fonction `%PUT` à votre avis ?

Question 2 Saisissez dans l'éditeur le code :

```
PROC CONTENTS DATA = sashelp.class;
RUN;
```

et soumettez-le. Dans quelle fenêtre les résultats se sont-ils affichés ? À quoi sert la procédure **`CONTENTS`** à votre avis ?

Question 3 Saisissez dans l'éditeur le code :

```
/*Tutoriel SAS
Découverte de l'interface*/
```

et soumettez-le. À votre avis, à quoi servent les délimiteurs `/**/` ?

Question 4 Fermez les différentes fenêtres du logiciel et utilisez le contenu du menu « Affichage » (*View*) pour les réafficher. Regardez ce qu'il se passe dans la barre de menus quand vous changer de fenêtre (de l'éditeur à l'explorateur de fichiers par exemple).

Question 5 Naviguez dans l'explorateur de fichier pour identifier où se situe la table `class`. À l'aide d'un clic-droit, affichez les informations sur les variables de cette variable (équivalent de la **`PROC CONTENTS`**). Identifiez le bouton permettant de remonter au dossier parent dans l'explorateur de fichiers et rendez vous dans la bibliothèque `work`, qui devrait normalement être vide.

L'aide dans le logiciel et sur internet

L'aide de SAS est accessible *via* le menu Aide > Aide SAS et documentation. Cette aide est très complète, quoiqu'un peu complexe au premier abord. En particulier, c'est dans l'aide que vous trouverez le détail de la syntaxe et des options de l'ensemble des procédures que vous allez être amenés à utilisez.

Si le moteur de recherche interne de l'aide n'est pas toujours très efficace, le fait que l'ensemble de la documentation soit également en ligne permet d'utiliser des moteurs de recherches plus performants (par exemple celui de Google).

Question 6 Utilisez l'aide dans le logiciel ou sur internet pour trouver des informations sur l'option `SHORT` de la **`PROC CONTENTS`** et sur la manière de l'utiliser. Appliquez de nouveau la **`PROC CONTENTS`** à la table `class` en utilisant cette option.

2 Lire, importer et exporter des données

Lire des données

Dans SAS, la manière habituelle d'accéder aux données enregistrées sur l'ordinateur est de définir une bibliothèque. Une bibliothèque est un genre de « raccourci » vers un dossier qui, une fois défini, permet de faire référence aux tables contenues dans ce dossier.

Un nom de bibliothèque ne peut pas excéder 8 caractères et, comme tous les noms dans SAS, ne doit pas contenir d'espaces ou de caractères spéciaux (accents notamment).

Note : Il existe des bibliothèques pré-définies dans SAS : c'est en particulier le cas des bibliothèques `sashelp` et `work` mentionnées précédemment.

Pour définir une bibliothèque, il faut utiliser l'instruction `LIBNAME` :

```
/*Définition de la bibliothèque malib*/
LIBNAME malib "chemin\vers\mon\dossier";
```

`malib` est le nom de la bibliothèque nouvellement créée, qui pointe vers le dossier dont le chemin est `"chemin\vers\mon\dossier"`. Si la table `matable` se situe dans ce dossier, alors il suffit désormais de saisir `malib.matable` pour faire référence à `matable` dans SAS.

Question 7 Après avoir téléchargé les données du tutoriel et les avoir décompressées, créez la bibliothèque `tuto` qui pointe vers le répertoire correspondant. Naviguez dans l'explorateur de fichiers pour vérifier que la bibliothèque est bien créée et ouvrez la table `pays`.

Question 8 En double-cliquant sur un en-tête de colonne, affichez les propriétés d'une variable. Repérez le label, la longueur et le format (qui seront abordés dans les parties suivantes). En parcourant le menu `Affichage`, affichez le nom des variables plutôt que leur libellé dans la fenêtre de visualisation de la table.

Question 9 Dans l'explorateur Windows, rendez vous dans le dossier dans lequel vous avez décompressé les données du tutoriel et double-cliquez sur le fichier `pays.sas7bdat`. Que se passe-t-il ? Dans l'explorateur de fichiers de SAS, que constatez-vous quant aux bibliothèques existantes ? Quel est l'intérêt selon vous de définir manuellement une bibliothèque plutôt que d'ouvrir une table en double-cliquant dessus ?

Recopier les données dans la bibliothèque work

Une fois la ou les bibliothèque(s) pointant vers le ou les répertoire(s) contenant les données utiles définie(s), il est plus prudent et souvent plus efficace de ne pas travailler directement sur les tables d'origine mais sur des copies.

La bibliothèque `work` est précisément conçue pour offrir un espace de travail temporaire le temps de la session.

1. Elle est **pré-définie** : pas besoin de `LIBNAME` pour la définir. Son emplacement est souvent tel que l'écriture et la lecture dans la `work` sont plus rapides que dans les bibliothèques définies par l'utilisateur.

2. Elle est **temporaire** : à la fin de la session tous les fichiers contenus dans la `work` sont supprimés.
3. Elle est **implicite** : dans l'ensemble des instructions soumises au logiciel, quand le nom de la bibliothèque est omis il comprend qu'il est fait implicitement référence à la bibliothèque `work`. En d'autres termes, `work.matable` et `matable` sont synonymes dans SAS.

Pour recopier une table de sa bibliothèque d'origine dans la `work`, on peut utiliser une étape **`DATA`** (*cf. infra*) avec le minimum d'instruction :

```
/*Copie de la table matable de la bibliothèque malib dans la work*/
DATA matable;
    SET malib.matable;
RUN;
```

Question 10 Recopiez la table `pays` dans la `work` et renommez-la `pays_sas`.

***Question 11** Dans certains cas, il est nécessaire d'accéder à un grand nombre de tables sans avoir la possibilité de toutes les recopier dans la `work`. Recherchez dans l'aide une option de l'instruction `LIBNAME` qui permette de créer une librairie en lecture seule : ainsi, les tables qu'elle contient ne pourront pas être modifiées par inadvertance.

Importer des données

Il est fréquent de devoir utiliser des données qui ne sont pas stockées nativement dans le format de base de données de SAS, `.sas7bdat`. Le moyen le plus simple pour importer ces données dans SAS est d'utiliser la **`PROC IMPORT`** :

```
PROC IMPORT DATAFILE = "chemin\vers\le\fichier" OUT = matable REPLACE;
RUN;
```

L'option `REPLACE` indique de remplacer la table spécifiée à l'instruction `OUT` si elle existe déjà.

Question 12 Importez les fichiers `pays.csv`, `pays.xls` et `pays.dta` (format Stata) dans la `work` et créez les fichiers `pays_csv`, `pays_xls` et `pays_dta`.

Question 13 En utilisant la **`PROC CONTENTS`** ou la **`PROC COMPARE`**, comparez les caractéristiques des tables `pays_sas`, `pays_csv`, `pays_xls` et `pays_dta`. En sachant que les trois tables importées sont originellement construites à partir de la table `pays_sas`, quelle réflexion cela vous inspire-t-il à propos des importations et exportations de données ?

Exporter des données

La **`PROC EXPORT`** fonctionne en miroir de la **`PROC IMPORT`**. Elle permet d'exporter facilement des données depuis SAS vers d'autres logiciels :

```
PROC EXPORT DATA = matable OUTFILE = "chemin\vers\le\fichier" REPLACE;
RUN;
```

Question 14 Exportez la table `pays` dans un format susceptible d'être lu dans un logiciel de statistique que vous connaissez déjà (**R**, Stata) et réimportez-la dans ce logiciel. Comparez ses caractéristiques dans SAS et dans ce logiciel.

3 Travailler sur des données

L'étape DATA

Dans SAS, la quasi-totalité des opérations de transformation de base de données (construction de nouvelles variables, sélection de variables et d'observations, fusions de différentes tables) s'effectuent dans un environnement appelé « étape **DATA** » (*Data Step* en anglais).

L'étape **DATA** commence par le mot-clé **DATA** et s'achève avec le mot-clé **RUN** (comme la quasi-totalité des procédures SAS d'ailleurs). Chaque instruction d'une étape **DATA** se termine par le signe `;`.

```
/*Structure générale de l'étape DATA*/
DATA matable;
    /*Des instructions se finissant chacune par ;*/
RUN;
```

Le nom de table qui suit immédiatement le mot-clé **DATA** est le nom de la table que va créer l'étape **DATA**. Si cette table existe déjà, elle est écrasée.

Dans les configurations les plus simples, on cherche juste à modifier une table (pour rajouter une variable par exemple) : dans ce cas il suffit d'utiliser l'instruction `SET` avec un seul nom de table après.

```
/*Modification de la table matable*/
DATA matable;
    SET matable;
    /*Des instructions pour modifier matable : création de variables,
       etc.;*/
RUN;
```

Conseil : SAS ne peut pas modifier une table quand celle-ci est ouverte. Si c'est le cas, il affiche un message d'erreur pas toujours très compréhensible. De manière générale, prenez le réflexe de fermer toutes les tables ouvertes quand vous lancez une étape **DATA**.

Les variables et leurs attributs

Types de variable et conversions

Dans SAS les variables peuvent être de deux types : le type numérique et le type caractère. Le type d'une variable conditionne fortement ses autres attributs ainsi que les fonctions et procédures qui sont susceptibles de lui être appliquées.

Note : Comme dans tout logiciel statistique, le type des variables est une notion avant tout informatique qui rend compte de la manière dont le logiciel enregistre les informations stockées dans la variable. Le type n'a que peu de choses à voir avec la nature d'une variable (quantitative, qualitative ordonnée, qualitative non-ordonnée) qui est une notion statistique et qui permet de déterminer les traitements pertinents à lui appliquer.

Un grand nombre de traitements sont spécifiques aux variables de type numérique : opérations arithmétiques, procédures exploitant les propriétés algébriques des nombres (calcul de statistiques, régressions, etc.). Inversement, quelques opérations ne peuvent porter que sur des variables de type caractère : extraction de chaînes par expression régulière, etc.

On est donc parfois (souvent) amené à convertir des variables d'un type vers l'autre, tout particulièrement quand les données originales ont dû être importées depuis des fichiers dits « plats » (`.csv`, `.txt`).

Dans SAS une variable, une fois qu'elle existe dans une table, ne peut pas changer de type. Pour convertir cette variable, il faut en créer une seconde avec les fonctions `INPUT()` et `PUT()`.

```
DATA matable;
    SET matable;

    /*Conversion de caractère vers numérique avec INPUT()*/
    mavarnum1 = INPUT(mavarchar1,BEST12.);

    /*Conversion de numérique vers caractère avec PUT()*/
    mavarchar2 = PUT(mavarnum2,BEST12.);

RUN;
```

Note : Dans les deux cas le format `BEST12.` joue le rôle d'un « masque » utilisé pour opérer la conversion. Il fonctionne la plupart du temps mais pas toujours parfaitement.

Question 15 Dans la table `pays`, créez la variable `code_cont2` de type numérique à partir de la variable `code_cont`.

Question 16 Dans la même étape **`DATA`**, effectuez la conversion en sens inverse pour créer la variable `code_cont3` à partir de `code_cont2`. Comparez les variables `code_cont` et `code_cont3`. D'où peut selon vous provenir le problème ?

Question 17 Essayer de convertir la variable `code_pays` en variable de type numérique (variable `code_pays2`). Consultez le journal pour voir en détails ce qu'indique SAS, puis observez le fichier. Quelle réflexion cela vous inspire-t-il quant aux conversions de variables de type caractère en variables de type numérique ?

Longueur, format et label de variable

En plus du type, les principaux attributs de variables disponibles dans SAS sont la longueur, le format et le label.
- **Longueur** : nombre d'octets utilisés par SAS pour stocker une observation sur le disque.
- **Format** : règle utilisée pour afficher les valeurs de la variable. Les formats permettent de limiter le nombre de chiffres significatifs affichés (sans arrondir pour autant les valeurs sous-jacentes), d'afficher des signes à côté des valeurs (%, signes monétaires) ou de procéder à des recodages.
- **Label** : descriptif de la variable pouvant contenir des espaces et des caractères spéciaux.

L'ensemble de ces attributs peuvent être visualisés soit à l'aide d'un clic-droit sur le nom de la table dans l'explorateur de fichiers soit à l'aide d'une **`PROC CONTENTS`**.

Dans une étape **`DATA`**, les instructions `LENGTH`, `FORMAT` et `LABEL` permettent de (re)définir les attributs d'une ou plusieurs variables simultanément :

TABLE 1 – Attributs des variables de type numérique

Valeur manquante	`.` ou plus rarement `.a .b ... .z`
Longueur	Entre 3 et 8 octets. Plus la longueur de la variable augmente, plus sa précision est importante (3 décimales à 3 octets, 15 à 8 octets).
Format	Formats de type numérique (dont le nom ne commence pas par `$`), notamment : – `BEST12.` : meilleure représentation sur 12 positions ; – `8.2` : représentation sur 8 positions avec 2 décimales ; – `percent8.2` : représentation sur 8 positions avec 2 décimales et le signe % après multiplication par 100.
Label	Chaîne de 255 caractères ou moins.

TABLE 2 – Attributs des variables de type caractère

Valeur manquante	`""`
Longueur	Entre 1 et 32 767 octets. Chaque octet supplémentaire permet de stocker un caractère de plus.
Format	Formats de type caractère (dont le nom commence par `$`) de la forme `$longueur.`. Exemple : `$40.` pour une variable permettant de stocker des chaînes de 40 caractères maximum.
Labels	Chaîne de 255 caractères ou moins.

```
/*Modification des attributs des variables mavarchar et mavarnum*/
DATA matable;
    LENGTH mavarchar $ 10 mavarnum 5;
    SET matable;
    FORMAT mavarchar $5. mavarnum 8.3;
    LABEL
        mavarchar = "Ma super variable caractère"
        mavarnum = "Ma super variable numérique"
    ;
RUN;
```

Note : L'instruction `LENGTH` précède l'instruction `SET` car une fois la longueur d'une variable déterminée dans une étape **`DATA`**, elle ne peut plus être modifiée. En s'intercalant entre le **`DATA`** et le `SET`, l'instruction `LENGTH` prédéfinit la longueur des variables avant même que des données n'y soient chargées par l'instruction `SET`.

Une autre solution consiste à utiliser l'instruction `ATTRIB` pour modifier simultanément plusieurs attributs d'une même variable :

```
/*Modification des attributs de la variable mavarchar*/
DATA matable;
    ATTRIB mavarchar
        LENGTH = $ 10
        FORMAT = $5.
        LABEL = "Ma super variable caractère"
    ;
```

```
    SET matable;
RUN;
```

Note : Dans l'instruction ATTRIB, il n'y a qu'un seul point-virgule à la toute fin.

Question 18 Créez la variable pib2 égale à la variable pib_12 mais de longueur égale à 3. Comparez ces deux variables. Pouvait-on s'attendre à ce résultat ?

Question 19 Reformater la variable pib_12 de façon à ce qu'elle n'affiche qu'une seule décimale après la virgule, puis reformatez-la de nouveau avec le format BEST12. pour vérifier qu'aucune information n'est perdue dans l'opération.

***Question 20** Retour sur la conversion de variables. Utilisez le format 1. pour recoder la variable code_cont2 en caractère sans rencontrer le même problème qu'à la question de la partie précédente. Comment comprenez-vous ce qu'il se passe ?

Question 21 Utilisez l'instruction LABEL pour modifier le label d'une ou plusieurs variables de la table pays. Comment supprimer un label ?

***Question 22** À l'aide de l'instruction ATTRIB et du mot-clé _ALL_ (qui permet de désigner d'un coup toutes les variables d'une table, *cf. infra*), comment supprimeriez-vous en une seule instruction tous les labels d'une table ?

Créer de nouvelles variables

Générer des variables aléatoire avec la fonction RAND()

La fonction RAND() permet de générer des observations tirées dans de nombreuses lois de probabilités :

- RAND("NORMAL") pour une loi normale d'espérance 0 et de variance 1 ;
- RAND("UNIFORM") pour une loi uniforme à valeurs dans $[0\,;1]$;
- RAND("POISSON",1) pour une loi de Poisson de paramètre 1.

Note : Les fonctions RANUNI() et RANNOR() permettent également de créer des variables tirées dans les lois uniforme ou normale respectivement. Néanmoins, ces générateurs de nombre aléatoires possèdent de moins bonnes propriétés que la fonction RAND() [5] .

Recoder des variables avec les clauses IF THEN ELSE

Les clauses IF THEN ELSE permettent de recoder des variables de type numérique ou caractère. Quand il s'agit de créer une nouvelle variable de type caractère, il est indispensable de définir sa longueur au préalable pour éviter une troncature inopportune.

```
/*Utilisation de IF THEN ELSE pour un recodage*/
DATA matable;
    SET matable;
    LENGTH mavar2 $ 20;
    IF mavar1 NE . AND mavar < 10 THEN mavar2 = "Moins de 10";
    ELSE IF mavar1 >= 10 AND mavar1 < 20 THEN mavar2 = "Entre 10 et 20";
    ELSE mavar2 = "Plus de 20";
RUN;
```

5. http ://www.statsblogs.com/2013/07/10/six-reasons-you-should-stop-using-the-ranuni-function-to-generate-random-numbers/

Les opérateurs =, NE (non-égal), >=, >, <=, <, IN() (égal à un au moins dans la liste) ainsi que AND, OR et NOT permettent d'évaluer des conditions complexes.

Conseil : Pour les variables de type numérique, les valeurs manquantes . valent dans SAS $-\infty$. C'est la raison pour laquelle il est indispensable de bien exclure les non-valeurs de la première modalité dans l'exemple ci-dessus.

Quand on souhaite imbriquer des clauses IF THEN ELSE, il est nécessaire de créer un bloc DO qui s'achève par un END :

```
/*Imbrication de IF THE ELSE avec DO et END*/
DATA matable;
    SET matable;
    IF mavar1 NE . THEN DO;
        IF mavar2 = "1" THEN mavar3 = 0;
        IF mavar2 = "2" THEN mavar3 = 1;
        IF mavar2 = "3" THEN mavar3 = 5;
    END;
    ELSE mavar3 = -15;
RUN;
```

Une syntaxe plus légère est disponible pour créer des indicatrices :

```
/*Création d'une indicatrice*/
DATA matable;
    SET matable;
    mavar2 = (mavar1 >= 12);
RUN;
```

Manipuler des chaînes de caractères

SAS dispose de nombreuses fonctions pour manipuler les chaînes de caractère :
- COMPRESS(varchar) supprime les espaces (ou éventuellement d'autres signes) ;
- TRIM(varchar) supprime les espaces en début et en fin de chaîne ;
- LENGTH(varchar) renvoie la longueur de la chaîne de caractère (pas la longueur de la variable, mais celle de chacune de ses modalités) ;
- INDEX(varchar,motif) renvoie la position de motif (une chaîne de caractère) dans varchar (s'il n'est pas présent, la fonction renvoie 0) ;
- SUBSTR(varchar,position,longueur) permet d'extraire la chaîne de caractère commençant à position et de longueur longueur ;
- SCAN(varchar,position,delimiteur) extrait le « mot » en position position, les mots étant séparés par délimiteur.

Note : Plusieurs fonctions (en particulier la fonction PRXMATCH()) permettent d'utiliser dans SAS des expressions régulières[6]. Ces techniques permettent des analyses extrêmement fines des chaînes de caractères.

Question 23 Dans la table pays, créer une variable alea tirée dans une distribution uniforme dans [0 ;1]. Créez alors la variable echant qui vaille 1 si alea est strictement supérieur à 0,70 et 0 sinon.

6. http ://www2.sas.com/proceedings/sugi29/043-29.pdf

Question 24 Construisez la variable de PIB par habitant en 2012 puis créez une nouvelle variable de type caractère indiquant les pays dont le PIB est inférieur ou supérieur à 5 000 $ par habitant et par an en utilisant les modalités : « Inf 5 000 $ » et « 5 000 $ et plus ». Si vous ne définissez pas la longueur de la variable recodée au préalable, que constatez-vous ? Quelle est la valeur de la variable recodée quand la variable de PIB par habitant n'est pas renseignée ?

Question 25 Utilisez un bloc `DO END` pour créer l'indicatrice de l'espérance de vie à la naissance des hommes strictement supérieure à 75 ans mais uniquement pour les pays du continent européen.

Question 26 Créez une nouvelle variable qui vaille, pour chaque observation, la longueur du nom du pays. Déterminez alors la longueur maximale des noms de pays : la longueur de la variable `nom_pays` vous semble-t-elle optimale ? Si non reformatez-la.

***Question 27** À l'aide de `INDEX()` ou `SCAN()`, identifiez les noms de pays composés de deux mots ou plus.

Modifier une table

Trier une table

La **`PROC SORT`** permet de modifier l'ordre de tri des observations d'une table :

```
/*Syntaxe de base de la PROC SORT*/
PROC SORT DATA = matable;
    BY mavar;
RUN;
```

Il est possible de trier par plusieurs variables successivement ou par ordre décroissant. L'instruction suivante trie ainsi d'abord par ordre croissant des modalités de la variable `mavar1` puis par ordre décroissant des modalités de la variable `mavar2` :

```
/*Variables de tri multiples et tri par ordre décroissant*/
PROC SORT DATA = matable;
    BY mavar1 DESCENDING mavar2;
RUN;
```

L'option `NODUPKEY` de la **`PROC SORT`** permet de supprimer les observations présentant des valeurs identiques pour les variables de tri (une seule observation est conservée, les autres sont supprimées).

```
/*Utilisation de l'option NODUPKEY*/
PROC SORT DATA = matable NODUPKEY;
    BY mavar;
RUN;
```

Conseil : Cette option est particulièrement utile pour vérifier que deux observations ne présentent pas de valeur identiques pour une ou plusieurs variables identifiantes.

Supprimer ou renommer des variables

Les instructions `DROP` et `KEEP` de l'étape **`DATA`** permettent de supprimer des variables d'une table.

```
/*Utilisation du DROP : supprimer une ou plusieurs variables*/
DATA matable2;
    SET matable;
    DROP mavar1;
RUN;

/*Utilisation du KEEP : ne conserver que les variables indiquées*/
DATA matable2;
    SET matable;
    KEEP mavar2;
RUN;
```

L'instruction `RENAME` permet de renommer des variables :

```
/*Utilisation du RENAME*/
DATA matable2;
    SET matable;
    RENAME mavar2 = mavar3;
RUN;
```

Quand on souhaite désigner un grand nombre de variables, il est parfois plus efficace d'utiliser des **listes de variables** : il s'agit de syntaxes reconnues par SAS pour désigner rapidement plusieurs variables (Table 3).

TABLE 3 – Syntaxes de listes de variables reconnues par SAS

`_ALL_`	Liste de toutes les variables de la table.
`_NUMERIC_ _CHARACTER_`	Listes de toutes les variables de type numérique de la table et de toutes les variables de type caractère de la table respectivement.
`mavar1-mavar99`	Liste des variables `mavar1 mavar2 ... mavar99`.
`mavar1--mavar99`	Liste des variables situées entre `mavar1` et `mavar99` dans l'ordre du fichier.
`mavar1-NUMERIC-mavar99` `mavar1-CHARACTER-mavar99`	Listes des variables respectivement numériques et caractères situées entre `mavar1` et `mavar99` dans l'ordre du fichier.
`mavar:`	Liste de toutes les variables de la table dont le nom commence par `mavar`.

Supprimer des observations ou les rediriger vers différentes tables

Les mots-clés `DELETE` et `OUTPUT` de l'étape **`DATA`**, combinés avec une structure `IF THEN ELSE`, permettent de supprimer des observations d'une table.

```
/*Utilisation de DELETE : supprimer une ou plusieurs observations*/
DATA matable2;
    SET matable;
    IF mavar1 = "1" THEN DELETE;
RUN;

/*Utilisation de OUTPUT : ne conserver que les observations indiquées*/
DATA matable2;
```

```
    SET matable;
    IF mavar1 NE "1" THEN OUTPUT;
RUN;
```

Note : Dans l'exemple précédent, `THEN OUTPUT` n'est pas absolument nécessaire : la ligne `IF mavar NE "1";` suffit à indiquer que l'on souhaite ne conserver que les observations qui respecte cette condition. Il est également possible d'utiliser l'instruction `WHERE`.

Le mot-clé `OUTPUT` est en réalité d'un usage beaucoup plus général. Par défaut, si aucun mot-clé `OUTPUT` n'apparaît dans une étape **`DATA`**, SAS en introduit un automatiquement juste avant le **`RUN;`**.

Utilisé explicitement, le mot-clé `OUTPUT` permet de contrôler très finement la manière dont les données sont traitées et réorganisées par une étape **`DATA`** :

```
/*Utilisation de OUTPUT pour rediriger des observations vers différentes
   tables*/
DATA matable1 matable2 matable3;
    SET matable;
    IF mavar = 1 THEN OUTPUT matable1;
    IF mavar = 2 THEN OUTPUT matable2;
    IF mavar = 3 THEN OUTPUT matable3;
RUN;
```

À noter que les tables indiquées après une instruction `OUTPUT` (`matable1`, `matable2` et `matable3` ici) figurent toutes les trois sur la première ligne de l'étape **`DATA`** (sans quoi SAS renvoie une erreur).

Les options de table

Certaines transformations de bases de données peuvent être effectuées en dehors de l'étape **`DATA`** *via* des options de tables.

```
/*Trois exemples d'utilisation des options de table*/
PROC CONTENTS DATA = matable(DROP = mavar1);
RUN;

PROC SORT DATA = matable(RENAME = (mavar2 = mavar1));
    BY mavar1;
RUN;

DATA matable2(KEEP = mavar3);
    SET matable1(WHERE = (mavar2 NE "1"));
RUN;
```

Note : Dans le troisième exemple, la structure de base est une étape **`DATA`** mais aucune transformation n'est effectuée à l'intérieur (tout est fait `via` les options de table).

Les options de tables, qui sont insérées entre parenthèses après un nom de table, donnent ainsi un peu plus de souplesse aux opérations sur les bases de données dans SAS : certaines manipulations simples peuvent être exécutées sans avoir à effectuer une étape **`DATA`** spécifique (par exemple en entrée ou en sortie d'une procédure, *cf. infra*).

Dans certains cas les options de tables permettent des gains significatifs de performance en évitant que SAS ne charge dans « son espace de travail » (*cf.* le polycopié « Langage

SAS », pp. 11-12) des variables ou des observations sur lesquelles on ne souhaite *in fine* pas travailler.

Question 28 Restreignez la table `pays` aux seuls pays africains et ne conservez que les variables `code_pays` et `pib_12`. Utilisez deux méthodes, une qui utilise des instructions et une autre des options de tables.

Question 29 Utilisez la syntaxe des listes de variables pour sélectionner rapidement toutes les variables commençant par `exp` (relatives à l'espérance de vie) ainsi que toutes les variables situées entre `code_pays` et `co2_09` dans le fichier.

Question 30 Utilisez plusieurs mots-clés `OUTPUT` pour créer autant de nouvelles tables qu'il y a de continents représentés dans la table `pays`.

Empiler ou fusionner plusieurs tables

Empiler des tables avec `SET`

Quand on souhaite travailler sur des tables présentant la même structure (les mêmes variables) mais portant sur des individus différents, il suffit d'empiler les différentes tables en saisissant plusieurs noms dans l'instruction `SET`.

```
/*Empilement de table avec SET*/
DATA matable3;
    SET matable1 matable2;
RUN;
```

var1	var2	var3
a	c	e
b	d	f

matable1

var1	var2	var4
gg	i	k
hh	j	l

matable2

var1	var2	var3	var4
a	c	e	
b	d	f	
g	i		k
h	j		l

matable3

Les attributs des variables de la table empilée sont déterminés à partir de ceux des variables des tables d'origine dans l'ordre dans lequel elles sont renseignées dans l'instruction `SET`. Par exemple, si une variable n'a pas la même longueur dans deux tables, sa longueur dans la table finale est celle qu'elle a dans celle des deux tables qui est renseignée en premier dans l'instruction `SET`.

Dans l'exemple qui précède, `var1` est de longueur 1 dans `matable1` et de longueur 2 dans `matable2`. Comme `matable1` est renseignée en premier dans l'instruction `SET`, la longueur de `var1` dans `matable3` est 1 et la valeur des observations provenant de `matable2` est tronquée. Pour régler ce problème, il suffit d'utiliser une instruction `LENGTH` :

```
/*Définition des longueurs de variable avant empilement*/
DATA matable3;
    LENGTH var1 $ 2;
    SET matable1 matable2;
RUN;
```

Fusionner des tables avec `MERGE BY`

Il est également fréquent que l'on dispose d'informations sur les mêmes individus dans des tables différentes. Pour rassembler ces informations dans une même table et les analyser, il est nécessaire de fusionner ces tables selon un identifiant. Dans SAS, toute fusion doit être précédée par un tri par la ou les variables de fusion (ici l'identifiant).

```
/*Tri des deux tables par id*/
PROC SORT DATA = matable1;
    BY id;
RUN;
PROC SORT DATA = matable2;
    BY id;
RUN;

/*Fusion selon l'identifiant id*/
DATA matable3;
    MERGE matable1 matable2;
    BY id;
RUN;
```

id	var1	var2
1	a	d
2	b	e
3	c	f

matable1

id	var3	var4
2	g	j
3	h	k
4	i	l

matable2

id	var1	var2	var3	var4
1	a	d		
2	b	e	g	j
3	c	f	h	k
4			i	l

matable3

Dans le cas des fusions de table, il est particulièrement utile d'identifier de quelle table proviennent les différentes observations. Pour ce faire, on utilise l'option de tables `IN =` :

```
/*Tri des deux tables par id*/
PROC SORT DATA = matable1;
    BY id;
RUN;
PROC SORT DATA = matable2;
    BY id;
RUN;

/*Fusion intérieure en utilisant l'option de table IN =*/
DATA matable3;
    MERGE matable1(IN = a) matable2(IN = b);
    BY id;
    IF a = 1 AND b = 1;
RUN;
```

Dans l'exemple qui précède, la variable `a` identifie les observations qui sont présentes dans `matable1` et la variable `b` identifie les observations qui sont présentes dans `matable2`. En ajoutant la condition : `IF a = 1 AND b = 1;`, `matable3` ne contiendra plus que les observations qui sont présentes à la fois dans `matable1` et `matable2`.

Question 31 La table `amerique` a la même structure que la table `pays` mais porte sur les pays du continent américain. Construire la table `monde` en empilant ces deux tables dans l'ordre `pays amerique`. Observez la valeur de `code_pays` pour les observations provenant de la table `amerique` et comparez-les avec leur contrepartie dans la table `monde`. D'où provient le problème à votre avis et comment le résoudre ?

Question 32 La table `class` porte sur les mêmes observations que les tables `pays` et `amerique`, dont `code_pays` est l'identifiant. Créez la table `monde_class` en fusionnant les tables `monde` et `class` par la variable d'identifiant.

Question 33 Créez la table `amerique_class` en fusionnant la table `amerique` avec la variable `class` par `code_pays`. Observez la table obtenue et, à l'aide de l'option de tables `IN =`, restreignez la table `amerique_class` aux seules observations qui proviennent de la table `amerique`.

4 Calculer des statistiques et représenter des données

Analyser une ou plusieurs variables qualitatives

Effectuer des tris à plat et des tris croisés : la PROC FREQ

La **PROC FREQ** est le principal outil pour effectuer des tris à plat et des tris croisés sur des variables qualitatives. Sa syntaxe est particulièrement simple :

```
/*Tris à plat avec la PROC FREQ*/
PROC FREQ DATA = matable;
    TABLES mavar1 mavar2;
RUN;
```

Pour pondérer des données (par exemple par un facteur de taille ou un poids de sondage), il suffit d'utiliser l'instruction `WEIGHT` :

```
/*Utiliser la pondération dans la PROC FREQ*/
PROC FREQ DATA = matable;
    TABLES mavar1 mavar2;
    WEIGHT pond;
RUN;
```

Des options permettent de personnaliser l'affichage : en particulier, l'option `MISSING` permet d'indiquer de ne pas exclure les valeurs manquantes mais de les présenter comme une modalité à part.

```
/*Les options de l'instruction TABLES*/
PROC FREQ DATA = matable;
    TABLES mavar1 / MISSING;
    WEIGHT pond;
RUN;
```

Pour analyser les croisements entre deux variables ou plus, il suffit de les séparer par le signe `*` :

```
/*Analyser deux variables ou plus*/
PROC FREQ DATA = matable;
    TABLES mavar1 * mavar2;
RUN;
```

Quand plusieurs variables sont analysées conjointement, il est possible de demander des statistiques d'association, comme la statistique du χ^2 ou le τ_b de Kendall :

```
/*Calculer des statistiques d'association avec la PROC FREQ*/
PROC FREQ DATA = matable;
    TABLES mavar1*mavar2 / CELLCHI2 CHISQ MEASURES;
RUN;
```

Utiliser des formats pour recoder des variables : la PROC FORMAT et l'instruction FORMAT

On l'a vu, les formats conditionnent la manière dont les données sont affichées dans une table. Plus encore, il est possible d'utiliser des formats dans des traitements pour recoder

des variables. Ces formats personnalisés jouent le rôle de table de correspondance entre les modalités d'une variable et des libellés [7].

La définition des formats est effectuée dans une **PROC FORMAT** :

```
/*Création du format $monformat.*/
PROC FORMAT;
    VALUE $monformat
        "1" = "Le libellé de la modalité 1"
        "2" = "Le libellé de la modalité 2"
        "3" = "Le libellé de la modalité 3"
    ;
RUN;
```

Note : Les noms des formats définis par les utilisateurs ne peuvent pas se terminer par un chiffre.

Une fois le format défini, il suffit de l'appliquer en utilisant l'instruction `FORMAT`, dans une **PROC FREQ** par exemple :

```
/*Application du format $monformat à la variable mavar*/
PROC FREQ DATA = pays;
    TABLES mavar;
    FORMAT mavar $monformat.;
RUN;
```

Note : De manière générale, les noms de format définis par les utilisateurs se terminent toujours par un point (comme ici dans l'instruction `FORMAT` de la **PROC FREQ**). Le seul endroit où ce n'est pas le cas est l'instruction `VALUE` de la **PROC FORMAT**.

Comme les variables, les formats ont des types : les formats de type numérique ne peuvent être appliqués qu'à des variables de type numérique et les formats de type caractère ne peuvent être appliqués qu'à des variables de type caractère. Par convention, **les formats de type caractère commencent par $ et les formats de type numérique non**.

Les formats de type caractère (comme le format `$monformat.` utilisé dans les exemples précédents) servent à expliciter les modalités des variables mais aussi à les recoder. Pour ce faire, il suffit de rassembler plusieurs modalités dans un même libellé :

```
/*Création d'un format de type caractère qui procède à un recodage*/
PROC FORMAT;
    VALUE $monformatchar
        "" = "Les valeurs manquantes"
        "1","2" = "Les deux premières modalités ensemble"
        "3" = "La troisième modalité à part"
        other = "Toutes les autres modalités ensemble"
    ;
RUN;
```

Les formats de type numérique servent principalement à effectuer des recodages, avec une syntaxe un peu différente :

7. À certains égards, cette utilisation des formats dans SAS rappelle le fonctionnement du type **factor** dans **R**.

```
/*Création d'un format de type numérique*/
PROC FORMAT;
    VALUE monformat
        . = "Les valeurs manquantes"
        low-20 = "Les valeurs inférieures ou égales à 20"
        20<-40 = "Les valeurs strictement supérieures à 20 et inférieures
           ou égales à 40"
        40<-high = "Les valeurs strictement supérieures à 40"
    ;
RUN;
```

Dans tous les cas, il faut appliquer le format à la variable que l'on souhaite recoder pour que celui-ci soit effectif le temps de la procédure, à l'aide d'une instruction `FORMAT` :

```
/*Application de plusieurs formats dans une PROC FREQ*/
PROC FREQ DATA = matable;
    TABLES mavarchar mavarnum;
    FORMAT mavarchar $monformatchar. mavarnum monformatnum.;
RUN;
```

Question 34 Effectuez un tri à plat sur la variable `code_cont`, d'abord sans pondérer puis en pondérant par la population en 2012.

Question 35 Définissez un format qui explicite les modalités de la variable `code_cont` et appliquez-le dans les tris à plat de la question précédente.

Question 36 Définissez un nouveau format qui recode la variable `class_14` en trois modalités (« *High income* », « *Middle income* », « *Lower income* ») et effectuez un tri à plat sur cette variable.

Rappel : C'est la table `class` qui contient la variable `class14` : pour l'utiliser de concert avec les variables de la table `pays`, il faut fusionner ces deux bases selon `code_pays`.

Question 37 Effectuez un tri croisé sur les variables `code_cont` et de `class_14`. Calculez la statistique du χ^2 et affichez les valeurs des χ^2 de cellule. Le test du χ^2 vous semble-t-il valide ici ?

Analyser une ou plusieurs variables quantitatives

Analyser une variable en détails : la PROC UNIVARIATE

La **`PROC UNIVARIATE`** produit un très grand nombre de statistiques : statistiques de tendance centrale (moyenne, médiane, mode), de dispersion (variance, écart-type, coefficient de variation), d'asymétrie (*skewness*), d'aplatissement (*kurtosis*), quantiles.

```
/*Syntaxe de la PROC UNIVARIATE*/
PROC UNIVARIATE DATA = matable;
    VAR mavar;
RUN;
```

Note : Les options `TRIMMED =` et `WINSORIZED =` de l'instruction **`PROC UNIVARIATE`** permettent de calculer respectivement des moyennes tronquées et winsorizées.

Comme dans la **`PROC FREQ`**, la pondération éventuelle est indiquée avec l'instruction `WEIGHT`. Il est néanmoins indispensable dans la plupart des cas de rajouter l'option `VARDEF = WGT` pour que les statistiques de dispersion aient un sens :

```
/*Utilisation d'une pondération*/
PROC UNIVARIATE DATA = matable VARDEF = WGT;
    VAR mavar;
    WEIGHT pond;
RUN;
```

Il est possible de restreindre un traitement à un sous-ensemble de la table en utilisant l'instruction `WHERE` :

```
/*Utilisation de l'instruction WHERE*/
PROC UNIVARIATE DATA = matable;
    WHERE mavar2 NE "1";
    VAR mavar1 ;
RUN;
```

L'instruction `BY` permet de ventiler les traitements selon les modalités d'une ou plusieurs variables qualitatives, pour autant que la table soit triée au préalable :

```
/*Utilisation de l'instruction BY*/
PROC SORT DATA = matable;
    BY mavar2;
RUN;
PROC UNIVARIATE DATA = matable;
    BY mavar2;
    VAR mavar1;
RUN;
```

Note : Les instructions `WHERE` et `BY` sont disponibles dans la plupart des procédures de SAS.

Comparer les statistiques univariées de plusieurs variables : la PROC MEANS

La **PROC MEANS** produit essentiellement les mêmes statistiques que la **PROC UNIVARIATE** mais les présente sous une forme qui facilite les comparaisons entre plusieurs variables.

```
/*Syntaxe de la PROC MEANS*/
PROC MEANS DATA = matable;
    VAR mavar1 mavar2;
RUN;
```

Par défaut, la **PROC MEANS** ne calcule que quelques statistiques élémentaires. Pour afficher davantage de statistiques, il suffit d'ajouter le mot-clé qui leur correspond à l'instruction **PROC MEANS** [8]

```
/*Choix des statistiques dans une PROC MEANS*/
PROC MEANS DATA = matable N NOBS SUM STD P10;
    VAR mavar1 mavar2;
RUN;
```

La **PROC MEANS** dispose d'une instruction `CLASS` analogue à l'instruction `BY` mais qui présente les résultats de façon beaucoup plus lisible et ne nécessite pas de tri préalable.

8. http ://support.sas.com/documentation/cdl/en/proc/61895/HTML/default/viewer.htm, paragraphe *statistic-keyword(s)*.

```
/*Utilisation de l'instruction CLASS de la PROC MEANS*/
PROC MEANS DATA = matable;
    CLASS mavar2;
    VAR mavar1;
RUN;
```

Calculer des coefficients de corrélation : la PROC CORR

La **PROC CORR** calcule plusieurs indicateurs de liaison entre variables quantitatives, en premier lieu le coefficient de corrélation linéaire de Pearson :

```
/*Syntaxe de la PROC CORR*/
PROC CORR DATA = matable;
    VAR mavar1 mavar2;
RUN;
```

Les options SPEARMAN et KENDALL de l'instruction **PROC CORR** permettent de calculer le coefficient de corrélation des rangs de Sperman et le τ de Kendall respectivement.

```
/*Utilisation des options SPEARMAN et KENDALL*/
PROC CORR DATA = matable SPEARMAN KENDALL;
    VAR mavar1 mavar2;
RUN;
```

Question 38 Construisez la variable de PIB par habitant et analysez-en la distribution à l'aide de la **PROC UNIVARIATE**, d'abord sans pondérer puis en pondérant par le nombre d'habitants. Vérifiez que l'option VARDEF = WGT modifie fortement les statistiques de dispersion.

Question 39 Utilisez les quartiles de la distribution non-pondérée pour définir un format de type numérique et effectuez un tri à plat sur la variable de PIB par habitant recodée avec ce format. Êtes-vous surpris(e) du résultat ?

Question 40 Comparez le neuvième décile d'espérance de vie moyenne des hommes et des femmes, d'abord avec une **PROC UNIVARIATE** puis avec une **PROC MEANS**.

Question 41 Ventilez les statistiques sur l'espérance de vie des femmes par continent, d'abord avec l'instruction BY de la **PROC UNIVARIATE** puis avec l'instruction CLASS de la **PROC MEANS**. Utilisez un format pour expliciter le code des continents.

Question 42 Analysez les corrélations entre PIB par habitant et émissions de CO_2 par habitant, en utilisant le coefficient de corrélation linéaire de Pearson, le coefficient de corrélation des rangs de Spearman puis le τ de Kendall.

Pour aller plus loin : La **PROC TABULATE** permet de facilement croiser des variables qualitatives et quantitatives (comme l'instruction CLASS de la **PROC MEANS**) et dispose d'options de mise en forme avancées.

Représenter des données dans SAS

Représenter une variable qualitative : la PROC GCHART

La **PROC GCHART** permet de représenter une variable qualitative sous la forme de diagrammes circulaires ou en bâtons.

```
/*Syntaxe de la PROC GCHART*/
PROC GCHART DATA = matable;
    PIE mavar; /*Pour un diagramme circulaire*/
    HBAR mavar; /*Pour un diagramme en bâtons horizontal*/
    VBAR mavar; /*Pour un diagramme en bâtons vertical*/
RUN; QUIT;
```

Note : Le mot-clé **QUIT** est utilisé ici car la plupart des procédures graphiques sont dites interactives : même après le **RUN** elles continuent à s'exécuter pour accepter de nouvelles instructions graphiques. Pour les quitter définitivement, il faut utiliser le mot-clé **QUIT**.

La **PROC GCHART** accepte les formats et il est possible d'utiliser les pondérations *via* l'option `FREQ =` des instructions `PIE`, `HBAR` et `VBAR`. Des options de mises en forme permettent également d'améliorer l'affichage [9].

```
/*Utilisation de l'option FREQ = et d'un format avec l'instruction PIE*/
PROC GCHART DATA = matable;
    PIE mavar / FREQ = pond;
    FORMAT mavar $monformat;
RUN; QUIT;
```

Construire des histogrammes et des densités empiriques : la PROC UNIVARIATE

La **PROC UNIVARIATE** dispose d'instructions spécifiques pour créer des représentations graphiques. L'instruction `HISTOGRAM` permet ainsi de produire un histogramme avec une estimation de la densité empirique :

```
/*L'instruction HISTOGRAM de la PROC UNIVARIATE*/
PROC UNIVARIATE DATA = matable;
    VAR mavar;
    HISTOGRAM / MIDPOINTS = 1 TO 100 BY 10 KERNEL (C = 1);
RUN;
```

L'option `MIDPOINTS = 1 TO 100 BY 10` indique que les classes qui constituent l'histogramme sont de largeur 10 et couvrent les valeurs allant de 1 à 100.

L'option `KERNEL` et son paramètre `C` indiquent d'estimer la densité empirique par une méthode de Kernel avec une fenêtre dont la largeur est indexée par le paramètre `C`.

Représenter des nuages de points ou des séries : la PROC GPLOT

La **PROC GPLOT** permet de représenter des nuages de points :

```
/*Syntaxe de la PROC GPLOT*/
SYMBOL VALUE = DOT;
PROC GPLOT DATA = matable;
    PLOT y * x; /*y en ordonnée, x en abscisse*/
RUN; QUIT;
```

9. https ://support.sas.com/documentation/cdl/en/graphref/63022/HTML/default/viewer.htm#gchart-bar.htm

L'instruction SYMBOL permet de personnaliser la présentation du graphique. En particulier, le mot-clé VALUE = (abrégé V =) permet de préciser le motif utilisé pour représenter les points.

Il est possible de changer la couleur des points selon les modalités d'une variable qualitative :

```
/*Ventilation selon les modalités d'une variable quali */
SYMBOL VALUE = DOT;
PROC GPLOT DATA = matable;
    PLOT y * x = mavarquali;
RUN; QUIT;
```

Pour représenter des séries, il faut modifier l'option INTERPOL (abrégée I) :

```
/*Représenter une série*/
SYMBOL INTERPOL = JOIN;
PROC GPLOT DATA = matable;
    PLOT y * x;
RUN; QUIT;
```

Note : Les points sont joints les uns aux autres en fonction de leur ordre dans le fichier : il est souvent utile avant une **PROC GPLOT** de trier la table selon la variable en abscisse.

Il est également possible de représenter deux séries en fonction de la même variable en abscisse avec l'option OVERLAY :

```
/*Représenter deux séries simultanément*/
SYMBOL INTERPOL = JOIN;
PROC GPLOT DATA = matable;
    PLOT (y1 y2) * x / OVERLAY;
RUN; QUIT;
```

Note : Pour annuler l'effet de `SYMBOL INTERPOL = JOIN;`, saisir `SYMBOL INTERPOL = NONE VALUE = DOT;`.

Construire des « boîtes à moustaches » : la PROC BOXPLOT

La **PROC BOXPLOT** permet de représenter et de comparer facilement la distribution d'une variable quantitative ventilée selon les différentes modalités d'une variable qualitative, pour autant que la table soit triée au préalable.

```
/*Syntaxe de la PROC BOXPLOT*/
PROC SORT DATA = matable;
    BY mavarquali;
RUN;
PROC BOXPLOT DATA = matable;
    VAR mavarquanti * mavarquali;
RUN;
```

L'option BOXSTYLE = SCHEMATIC permet d'isoler les valeurs extrêmes de la distribution :

```
/*Utilisation de l'option BOXSTYLE = SCHEMATIC*/
PROC SORT DATA = matable;
    BY mavarquali;
RUN;
```

```
PROC BOXPLOT DATA = matable;
    VAR mavarquanti * mavarquali / BOXSTYLE = SCHEMATIC;
RUN;
```

Question 43 Représentez la répartition des pays puis de la population mondiale par continent sous la forme d'un diagramme circulaire.

Question 44 Représentez l'histogramme et la densité du PIB par habitant en ajustant manuellement la largeur des bâtons et la fenêtre de la fonction Kernel.

Question 45 Représentez le nuage de points croisant PIB par habitant et émissions de CO_2 par habitant en colorant différemment chaque continent.

Question 46 Représentez la distribution du PIB par habitant selon les continents en isolant les valeurs extrêmes.

Pour aller plus loin : La **PROC SGPLOT** permet de produire des graphiques de bonne qualité et de superposer plusieurs types de graphiques.

Exporter des résultats depuis SAS

La principale méthode pour exporter des résultats depuis SAS est l'*Output Delivery Service* (ODS) dont le principe est le suivant :

1. À un moment donné du code (par exemple au début), on initialise l'ODS avec une instruction du type `ODS XXX FILE = "chemin\vers\un\fichier";`
2. On effectue les traitements (**PROC FREQ**, **PROC MEANS**, etc.) comme si de rien n'était.
3. À un moment donné du code (par exemple à la fin), on clôt l'ODS avec une instruction du type `ODS XXX CLOSE;`.

Une fois l'instruction `ODS XXX CLOSE;` soumise, un fichier apparaît à l'emplacement indiqué qui contient les résultats des instructions soumises tant que l'ODS était actif.

Note : Cette méthode est principalement intéressante pour les tableaux. Pour les graphiques, un clic-droit permet de les sauvegarder sous forme d'images. Des méthodes permettent également d'exporter automatiquement les graphiques, notamment ceux produits par la **PROC SGPLOT**.

Plusieurs formats sont disponibles pour les exports en ODS (dont les noms remplacent XXX dans l'exemple précédent), mais deux sont particulièrement utiles :
– `RTF` : crée un fichier `.rtf` dont les éléments sont faciles à réutiliser.
– `PDF` : crée un fichier `.pdf` facile à partage et à imprimer.

```
/*Exemple d'utilisaion de l'ODS RTF*/

/*1. On initialise l'ODS RTF*/
ODS RTF FILE = "chemin\vers\mon\fichier.rtf";

/*2. On effectue les traitements*/
PROC FREQ DATA = matable;
    TABLES mavar;
RUN;

/*3. On clot l'ODS*/
ODS RTF CLOSE;
```

Conseil : Certains tableaux sont parfois trop larges pour des exports corrects au format portrait. Pour faire passer l'ensemble des exports au format paysage, soumettez : `OPTIONS ORIENTATION = LANDSCAPE;`. Pour revenir en portrait, soumettez : `OPTIONS ORIENTATION = PORTRAIT;`

Question 47 Exporter l'ensemble des traitements de cette partie dans un fichier `.pdf` et dans un fichier `.rtf`.