Introduction au logiciel PSPP *version 0.4.0*

Julie Séguéla

Merci à Joseph Saint Pierre, ingénieur au CICT, pour l'aide précieuse qu'il a apportée à la rédaction de cette documentation.

Préambule

L'objectif de cette documentation n'est pas de présenter de manière exhaustive les commandes du logiciel PSPP. Cette introduction au logiciel présente les commandes essentielles au traitement des gros fichiers de données issus d'enquêtes, dans l'optique d'une analyse ultérieure. Les commandes statistiques qui y sont décrites sont élémentaires, mais aussi fondamentales en tant que base de tout calcul statistique plus avancé.

Pour familiariser l'utilisateur au logiciel, de nombreux exemples simples illustrent les commandes. À la fin de la documentation, on peut trouver un exemple concret présentant une manière d'utiliser le logiciel dans une situation réelle.

Les commandes qui ne sont pas décrites dans cette documentation peuvent être trouvées dans le manuel GNU PSPP à l'adresse suivante :
http ://www.gnu.org/software/pspp/manual/.

PSPP est un logiciel libre en cours de développement. Lors de son utilisation, on rencontre donc quelques bugs qui toutefois n'altèrent pas son efficacité dans le domaine décrit ci-dessus.

Table des matières

1 Présentation

1.1 La philosophie des logiciels libres

L'expression *logiciel libre* (en anglais *free software*), donnée par Richard Stallman, fait référence à quatre libertés pour les utilisateurs ayant acquis une version du logiciel, définies par la licence de ce logiciel :

- Liberté 0 : La liberté d'exécuter le programme, pour tous les usages ;
- Liberté 1 : La liberté d'étudier le fonctionnement du programme ;
- Liberté 2 : La liberté de redistribuer des copies ;
- Liberté 3 : La liberté d'améliorer le programme et de publier ses améliorations.

Le principe du logiciel libre est le suivant : si on se procure le code source du logiciel, on a le droit de le redistribuer, et on peut le modifier et l'adapter à ses besoins si on le veut. On doit aussi s'assurer que les personnes à qui on fait parvenir le logiciel connaissent ces droits. Enfin, si on apporte des améliorations au logiciel, on doit les rendre publiques.

Les logiciels libres sont protégés par des licences libres garantissant le respect des quatres libertés fondamentales. En 1983, Richard Stallman crée le projet GNU, qui a pour objectif de construire un Unix dont la totalité des sources seraient libres. Parallèlement, il fonde la *Free Software Foundation* et rédige avec l'aide d'un avocat la *licence publique générale GNU* (*GNU General Public License*). Cette licence, la plus connue et utilisée pour les logiciels libres, garantit que tout logiciel dérivé du logiciel protégé soit également placé sous la même licence, et soit donc libre. C'est ainsi qu'un cadre juridique a été créé, encourageant le développement du logiciel libre.

Attention, logiciel *libre* ne veut pas forcément dire *gratuit*. En effet, la commercialisation du logiciel n'est pas contraire aux libertés fondamentales.

1.2 Présentation de PSPP

PSPP, logiciel libre distribué sous les termes de la *GNU General Public License*, est un outil pour l'analyse statistique des données qui interprète les commandes du langage SPSS. PSPP peut lire et traiter les fichiers SPSS.

La version la plus récente de PSPP, version 0.4.0, est incomplète, ce qui rend le logiciel assez pauvre en termes de calcul statistique. PSPP peut donc être utilisé pour les statistiques élémentaires, la gestion et la manipulation des gros fichiers de données. Par exemple, PSPP peut être utilisé pour transformer et préparer pour l'étude des fichiers de type dépouillement d'enquêtes, de sondages, etc.

Si on se place dans le cadre des logiciels libres, l'utilisation du logiciel *R* pour les calculs statistiques est compatible avec l'utilisation de PSPP, qui peut effectuer des opérations de gestion de données moins accessibles avec *R*. L'annexe 2, section 13.2, page 116, explique comment travailler avec *R* sur des données traitées une première fois avec PSPP.

Une documentation sur PSPP est disponible sur le site du *Projet GNU* à l'adresse *http ://www.gnu.org/software/pspp/manual/*.

1.3 Comment installer PSPP ?

PSPP est un logiciel qui fonctionne uniquement avec le système UNIX. Avec le système d'exploitation Windows, une solution est proposée dans la section 2 de ce paragraphe pour utiliser PSPP. On peut se procurer le code source de PSPP dans un format compressé sur le site *ftp ://ftp.gnu.org/pub/gnu/pspp/*. Il ne faut pas oublier de décompresser le code source avant de commencer l'installation de PSPP.

1.3.1 Installer PSPP sous UNIX

Pour installer PSPP, il faut suivre les étapes suivantes dans l'ordre (seules les étapes indispensables sont énumérées ici, pour les étapes optionnelles, consulter la documentation sur le site *http ://www.gnu.org/software/pspp/manual/*) :

1. Se placer dans le répertoire contenant le code source de PSPP.
2. Taper *./configure* pour configurer le système d'exploitation et le compilateur. Cette opération prend un certain temps et pendant son déroulement, des messages précisant les caractéristiques contrôlées apparaissent.
3. Taper *make* pour compiler.
4. Taper *make check* pour lancer les tests automatiques sur l'ensemble compilé.
5. Taper *make install* pour installer le code binaire de PSPP, par défaut dans */usr/local/bin/*. Le répertoire */usr/local/share/pspp/* est créé et contient les fichiers dont PSPP a besoin au cours de l'exécution. Cette étape entraine aussi l'installation de la documentation dans le répertoire */usr/local/info/*, mais seulement si le répertoire existe déjà.

1.3.2 Comment faire avec Windows ?

Pour utiliser PSPP sous Windows, il faut installer *Cygwin*. *Cygwin* est un logiciel qui crée un environnement UNIX sous Windows et offre les utilitaires de GNU. Ainsi, il est possible d'utiliser les logiciels GNU et de porter la plupart des applications UNIX sous Windows.

Pour installer *Cygwin*, on peut se rendre à l'adresse suivante :
http ://www.cygwin.com. Attention, au cours du téléchargement, il faut spécifier les paquets à installer. Il est préférable de tout installer afin que Cygwin possède tous les utilitaires indispensables au fonctionnement de PSPP.

1.4 Les commandes UNIX indispensables

Pour faciliter l'utilisation de PSPP, voici quelques commandes à retenir :
- `cd` *chemin_d'accès* : positionne sur le répertoire désigné
- `ls` *nom_repertoire* : liste le contenu d'un répertoire (`-l` : avec informations détaillées)
- `pwd` : affiche le répertoire courant
- `mkdir` *nom_repertoire* : crée un répertoire
- `cp` *nom_fichier nom_copie* : copie un fichier dans le répertoire courant
- `cp` *nom_fichier chemin_d'accès* : copie un fichier dans le répertoire désigné
- `mv` *nom_fichier/nom_repertoire nouveau_nom* : renomme un fichier ou un répertoire
- `mv` *nom_fichier/nom_repertoire chemin_d'accès* : déplace un fichier ou un répertoire dans le répertoire désigné
- `ln` *nom_fichier nouveau_nom* : crée une nouvelle référence pour un fichier
- `rm` *nom_fichier/nom_repertoire* : supprime la référence d'un fichier ou d'un répertoire
- `grep` "*expression*" *nom_fichier*(`>`*nom_destination*) : affiche les lignes d'un fichier contenant l'expression (`-i` : majuscules et minuscules identiques ; `-n` : avec numéros de lignes ; `-v` : seulement les lignes ne contenant pas l'expression)
- `head` *nom_fichier*(`>`*nom_destination*) : liste les premières lignes d'un fichier (`-n` : nombre de lignes affichées, 10 par défaut)
- `tail` *nom_fichier*(`>`*nom_destination*) : liste les dernières lignes d'un fichier (`-n` : nombre de lignes affichées, 10 par défaut)
- `wc` *nom_fichier* : affiche le nombre de caractères ou mots ou lignes d'un fichier (`-c` : nombre de caractères ; `-w` : nombre de mots ; `-l` : nombre de lignes)

1.5 Comment utiliser PSPP ?

PSPP peut fonctionner en *mode ligne*, mais les sections qui suivent présentent une utilisation plus pratique du logiciel.

1.5.1 Créer un fichier syntaxe

On utilise un éditeur pour taper les commandes PSPP dans un fichier syntaxe. Pour ouvrir un fichier avec un éditeur, GNU Emacs par exemple, on tape la commande :

```
emacs nom_fichier &
```

Avec X Window, le symbole '`&`' permet de garder la main pour taper de nouvelles commandes dans le shell UNIX.

On peut utiliser un autre éditeur de texte mais GNU Emacs, créé par Richard Stallman, est l'éditeur de référence dans le monde du logiciel libre. En effet, GNU Emacs présente de nombreuses fonctionnalités qui facilitent la manipulation du

texte (coloration syntaxique, commandes d'édition, macros, etc.). De plus, il est possible de partager la zone d'édition en plusieurs fenêtres, ce qui permet à l'utilisateur d'un logiciel d'afficher en même temps le fichier de commandes et le résultat de la compilation.

1.5.2 Exécuter un fichier syntaxe

Pour exécuter un fichier syntaxe, on tape la commande :

```
pspp nom_fichier
```

Pour exécuter deux fichiers ou plus, on tape la commande :

```
pspp fichier1 fichier2 ...
```

Entre deux exécutions, le dictionnaire est effacé. Pour que le dictionnaire ne soit pas effacé entre deux exécutions de fichiers, on tape la commande :

```
pspp fichier1 + fichier2 ...
```

Le symbole '+' permet de concaténer les deux fichiers qui sont alors exécutés comme un unique fichier.

Il existe un autre moyen d'exécuter un fichier syntaxe si le système d'exploitation le permet. Si PSPP est installé dans le répertoire */usr/local/bin*, on tape la ligne suivante au début du fichier :

```
#! /usr/local/bin/pspp
```

Si PSPP est installé dans un autre répertoire, on remplace */usr/local/bin* par le chemin d'accès au répertoire.

Ensuite, on tape la ligne suivante dans le shell UNIX :

```
chmod +x nom_fichier
```

Maintenant, on peut exécuter le fichier en tapant seulement son nom dans le shell UNIX.

Remarque : PSPP ignore toutes les lignes commençant par '`#!`'.

1.5.3 Visualiser les résultats

Les résultats obtenus suite à l'exécution d'un fichier sont écrits dans le fichier *pspp.list*. On peut visualiser les résultats avec la commande *less*, si le système d'exploitation le permet, ou avec d'autres outils comme *cat* ou *more*. On peut aussi voir les résultats en ouvrant le fichier *pspp.list* avec l'éditeur Emacs (en double fenêtre par exemple).

Attention, à chaque nouvelle exécution, le fichier *pspp.list* est écrasé par les nouveaux résultats. Pour conserver les derniers résultats obtenus, il faut renommer le fichier *pspp.list* en tapant la commande :

```
mv pspp.list nom_fichier
```

1.5.4 Editer les résultats

On peut éditer les résultats obtenus dans le fichier *pspp.list* avec Emacs pour les inclure dans un rapport ou une documentation, et éventuellement y apporter des modifications.

2 Le langage

Cette section a pour but de décrire les généralités du langage PSPP.

2.1 Les identifiants

Les identifiants sont des noms qui spécifient des variables, des commandes, ou des sous-commandes. Le premier caractère d'un identifiant doit être une lettre, '`#`', ou '`@`'. Les caractères suivants doivent être des lettres, des chiffres, ou l'un des caractères spéciaux suivants : '`. _ $ # @`'.

Les identifiants peuvent avoir n'importe quelle longueur mais seulement les 64 premiers octets sont significatifs. Un nom écrit en majuscules ou en minuscules représente le même identifiant : `exemple`, `EXEMPLE`, `Exemple`, `ExEmPlE` sont différentes représentations du même identifiant. Certains identifiants sont réservés et ne peuvent être utilisés que dans certains contextes qui seront précisés ultérieurement. Voici la liste des identifiants réservés :

```
ALL AND BY EQ GE GT LE LT NE NOT OR TO WITH
```

2.2 Les mots-clés

Les mots-clés forment une sous-classe des identifiants. Les noms des commandes et sous-commandes sont des mots-clés. Les mots-clés peuvent être abrégés par leurs trois (ou plus) premiers caractères, si l'abréviation n'est pas ambiguë. Par exemple : FREQ pour FREQUENCIES, ou DES pour DESCRIPTIVES. Les noms de variables peuvent être utilisés comme des mots-clés.

2.3 Les nombres

Les nombres sont exprimés en valeurs décimales. La partie décimale est séparée de la partie entière par un point (facultatif). Les nombres peuvent être exprimés en notation scientifique en ajoutant le symbole 'e' suivi d'un exposant en base 10. Ainsi, '`1.234e3`' représente la valeur '`1234`'. Les nombres négatifs sont précédés par le signe '`-`'. Attention, dans les situations où le symbole '`-`' est attendu, un nombre à première vue négatif est traité comme '`-`' suivi d'un nombre positif. Aucun espace vide n'est autorisé à l'intérieur d'un nombre, sauf entre le signe '`-`' et le nombre qui le suit. Voici quelques exemples de nombres :

```
-2   3.14159   7e10  -.45   0.789   560.  3e-5
```

2.4 Les chaînes de caractères

Les chaînes de caractères (*strings*) sont spécifiées entre deux apostrophes simples (`'`) ou doubles (`''`). Pour inclure dans la chaîne de caractères le type d'apostrophe utilisé, il faut le doubler : `'L''exemple'`. Dans les chaînes de caractères, les espaces vides et les majuscules sont significatifs.

Les chaînes de caractères peuvent être concaténées en utilisant le symbole '+'. Ainsi, `''ex'' + 'em' + 'ple'` est équivalent à `'exemple'`. Après concaténation, la longueur maximale d'une chaîne est de 255 caractères.

2.5 Les signes de ponctuation et les opérateurs

```
, / = ( ) + - * ** < <= <> > >= ~= & | .
```

Ces symboles peuvent apparaître dans la syntaxe des commandes mais le signe '.' est utilisé seulement pour marquer la fin d'une commande.

2.6 La structure des commandes

Les commandes PSPP ont une structure commune. Une commande commence par un nom de commande, comme DATA LIST ou FREQUENCIES, qui peut être suivi d'une ou plusieurs sous-commandes. De même, ces dernières commencent par un nom de sous-commande et peuvent admettre une ou plusieurs spécifications. Le nom de la sous-commande est séparé de ses spécifications par un signe '=' ou par un espace vide. Les spécifications sont séparées les unes des autres par des virgules ou par des espaces vides. Chaque sous-commande est séparée de la suivante par le symbole '/'. Le moyen le plus commun de marquer la fin d'une commande est de terminer la ligne de commande avec un point. Par défaut, une ligne blanche peut aussi terminer la commande.

2.7 Les observations manquantes

PSPP dispose d'un support spécial pour les valeurs numériques inconnues. Une valeur spéciale, appelée valeur *system-missing* et indiquant l'absence de valeur, est attribuée aux observations manquantes. Ainsi, ces dernières sont exclues de l'analyse des données.

La valeur *system-missing* n'existe que pour les valeurs numériques ; les chaînes de caractères sont toujours définies par une valeur, même s'il s'agit d'une suite d'espaces.

On peut assigner aux variables de type *numeric* ou *string* une valeur appelée valeur *user-missing*. Les valeurs *user-missing* sont traitées comme les valeurs *system-missing*, à savoir l'exclusion automatique des analyses. Les variables de type *string* de plus de 8 caractères ne peuvent pas avoir de valeur *user-missing*.

Pour plus de détails sur l'exclusion des valeurs manquantes, référez-vous aux sections décrivant les commandes statistiques.

2.8 Les variables

Les données sont stockées dans des variables dont les caractéristiques sont décrites dans les sections qui suivent. Le *dictionnaire* contient l'ensemble des variables et de leurs attributs.

2.8.1 Les attributs des variables

- **Name** : c'est un identifiant, il doit être différent pour chaque variable. Le dernier caractère d'un nom de variable ne doit pas être '.' : si l'identifiant se trouve en fin de commande, le signe '.' sera interprété comme indiquant la fin de la commande. Le dernier caractère ne doit pas non plus être le symbole '_' car de tels identifiants sont utilisés par certaines procédures de PSPP. Pour plus d'informations sur les identifiants, reportez-vous à la section 2.1, page 9.
- **Type** : numérique (*numeric*) ou chaîne de caractères (*string*).
- **Width** : seulement pour les variables de type *string*. Les chaînes dont la longueur est inférieure ou égale à huit caractères sont appelées chaînes de caractères courtes (*short string variables*). Ces dernières peuvent être utilisées dans certaines procédures où les chaînes dont la longueur est supérieure à huit caractères (*long string variables*) ne sont pas admises.
- **Position** : les variables sont rangées dans le dictionnaire dans un ordre particulier. Pour voir cet ordre, vous pouvez utiliser la commande *DISPLAY*, cf. section 5.1, page 26.
- **Missing values** : cf. section 2.7, page 10.
- **Variable label** : c'est une chaîne de caractères qui décrit la variable. Cf. commande *VARIABLE LABELS*, section 5.2, page 27.
- **Value label** : c'est une chaîne de caractères associée à une valeur possible de la variable. Cf. commande *VALUE LABELS*, section 5.3, page 28.
- **Print format** : affiche la largeur, le format, et pour les variables numériques, le nombre de chiffres dans la partie décimale. Cf. commande *FORMATS*, section 5.9, page 34.
- **Write format** : similaire à print format, mais utilisé par certaines commandes qui écrivent des fichiers binaires. Cf. commande *FORMATS*, section 5.9, page 34.

2.8.2 Les variables connues par PSPP

Ces variables, dont les valeurs ne peuvent pas être modifiées, sont utilisables seulement dans les expressions :

- `$CASENUM` : numéro de l'observation du moment.
- `$DATE` : date du jour (JJ MMM AA), en format A9.
- `$JDATE` : nombre de jour entre le 15 octobre 1582 et la date du jour.

Ces identifiants sont réservés et ne peuvent être utilisés que pour désigner les variables décrites ci-dessus.

2.8.3 Lister des noms de variables

Pour faire référence à un ensemble de variables, il faut écrire la liste des noms des variables, en les séparant avec un espace ou une virgule. Pour inclure dans la liste une suite de variables du dictionnaire, il faut écrire le nom de la première variable de la suite, puis '`TO`', puis le nom de la dernière variable. Ainsi, si le dictionnaire contient les variables suivantes dans cet ordre : NOM,

PRENOM, AGE, TAILLE, POIDS, alors '`PRENOM TO TAILLE`' fait référence à la liste PRENOM, AGE, TAILLE.

'`TO`' peut aussi être utilisé pour définir une suite de variables ayant la même racine et finissant par des entiers consécutifs. Ainsi, '`X1 TO X5`' définit une suite de 5 variables : X1, X2, X3, X4, X5. '`EX008 TO EX011`' définit 4 variables : EX008, EX009, EX010,EX011. Les syntaxes '`EX001 TO EX4`', et '`EX5 TO EX1`' ne sont pas valides.

Exemple d'utilisation avec la commande DATA LIST :

```
DATA LIST TABLE /X01 TO X04 1-4 (1).
```

2.8.4 Spécifier des noms de fichiers

Certaines commandes nécessitent la spécification d'un nom de fichier dans une sous-commande *FILE* ou *OUTFILE*. Par convention, les noms de fichiers sont spécifiés entre apostrophes comme des chaînes de caractères, mais ce n'est pas obligatoire. En revanche, comme pour les chaînes de caractères, les majuscules et minuscules sont significatives dans un nom de fichier.

3 Entrer les données

Une observation est un ensemble d'informations correspondant à une unité statistique ("individu" ou "cas"). Voici une manière de présenter un ensemble d'observations :

observation	var1	var2	var3
1	info11	info12	info13
2	info21	info22	info23
3	info31	info32	info33

Cette section décrit les commandes qui permettent de définir des variables, et de lire et écrire des données.

3.1 DATA LIST

DATA LIST est la commande de lecture des données la plus fondamentale. Cette commande admet plusieurs formes qui sont décrites dans les sections qui suivent.

3.1.1 DATA LIST FIXED

Syntaxe

```
DATA LIST [FIXED]
        {TABLE,NOTABLE}
        FILE='filename'
        RECORDS=record_count
        /var_list start-end [format_spec] (ou /var_list (fortran_spec)).
```

Description

La commande DATA LIST FIXED est utilisée pour lire des fichiers de données dont les valeurs sont situées à des positions fixes sur chaque ligne d'un enregistrement. Le mot-clé *FIXED* est facultatif.

La sous-commande *FILE* doit être utilisée si les données proviennent d'un fichier extérieur. Il faut préciser le nom du fichier source, qui peut être un fichier de type *handle* (cf. commande *FILE HANDLE*, section 3.3, page 20).
Si la sous-commande *FILE* n'est pas utilisée, il faut alors spécifier les données à l'aide de la commande *BEGIN DATA*, cf.section 3.2, page 19.

Dans la plupart des fichiers, un enregistrement correspond à une ligne. Si ce n'est pas le cas, la sous-commande *RECORDS* permet de préciser le nombre de lignes occupées par un enregistrement. Cette sous-commande est facultative, DATA LIST permet le calcul du nombre de lignes par enregistrement à partir de la spécification des variables.

La commande DATA LIST peut retourner une table décrivant la manière dont le fichier de données est lu. Par défaut, la table est retournée (*TABLE*). Pour ne pas la retourner, on utilise la sous-commande *NOTABLE*.

La liste des variables vient en dernier. Si un enregistrement est composé de plusieurs lignes, la spécification des variables de chaque ligne doit commencer par un slash ('/'). Le nombre de variables à lire n'est pas limité.

La spécification des variables consiste en une liste de noms de variables suivis par leur localisation dans la ligne. Pour spécifier la localisation d'une variable dans une ligne, on peut utiliser le style colonne, ou le style *FORTRAN*. Avec le style colonne, on indique après le nom de la variable les numéros, séparés par un trait d'union, de la première et de la dernière colonne occupées par la variable. Ainsi, si une variable occupe les cinq premières colonnes, on écrit '`1-5`'. Par défaut, les variables sont considérées comme étant au format 'F' (format numérique standard). Pour utiliser un autre format pour une variable, il faut le préciser entre parenthèses après le nom et la localisation. Pour le format 'A' (format standard pour les chaînes de caractères), on écrit '`(A)`'. Un nombre entier peut être lu comme un nombre ayant une partie décimale à d chiffres si on écrit '`(d)`'. Par exemple, pour que la valeur '`35689`' soit interprétée comme la valeur '`35.689`', il faut mettre la notation '`(3)`'.

Exemple 1

```
DATA LIST
    FILE=exemple1
    /NOM 1-10 (A) PRENOM 12-21 (A) SEXE 23 AGE 25-27 TAILLE 29-31 (2).
```

Cette commande définit les variables suivantes :
- NOM : une chaîne de 10 caractères, de la colonne 1 à la colonne 10 ;
- PRENOM : une chaîne de 10 caractères, de la colonne 12 à la colonne 21 ;
- SEXE : une variable numérique, dans la colonne 23 ;
- AGE : une variable numérique, de la colonne 25 à la colonne 27 ;
- TAILLE : une variable numérique, de la colonne 29 à la colonne 31. Les données concernant la taille sont entrées en centimètres et sont lues en mètres.

Les données lues se trouvent dans le fichier 'exemple1' :

```
Dupont     Pierre     1  24 183
Durant     Sophie     2  32 168
Boyer      Danielle   2  42 163
```

La sortie est la suivante :

```
1.1 DATA LIST.  Reading 1 record from file exemple1.
+--------+------+-------+------+
|Variable|Record|Columns|Format|
#========#======#=======#======#
|NOM     |     1|  1- 10|A10   |
|PRENOM  |     1| 12- 21|A10   |
|SEXE    |     1| 23- 23|F1.0  |
|AGE     |     1| 25- 27|F3.0  |
|TAILLE  |     1| 29- 31|F3.2  |
+--------+------+-------+------+
```

Exemple 2

```
DATA LIST
    FIL=exemple2
    REC=2
    /QUES01 TO QUES05 1-5
    /QUES06 TO QUES10 1-5.
```

Cette commande définit les variables numériques QUES01, QUES02, QUES03, QUES04, QUES05 sur la première ligne de l'enregistrement, et les variables numériques QUES06, QUES07, QUES08, QUES09, QUES10 sur la deuxième ligne.

Les données sont extraites du fichier 'exemple2' :

```
00001
22550
22244
88569
54662
57745
```

La sortie est la suivante :

```
1.1 DATA LIST.  Reading 2 records from file exemple2.
+--------+------+-------+------+
|Variable|Record|Columns|Format|
#========#======#=======#======#
|QUES01  |     1|  1-  1|F1.0  |
|QUES02  |     1|  2-  2|F1.0  |
|QUES03  |     1|  3-  3|F1.0  |
|QUES04  |     1|  4-  4|F1.0  |
|QUES05  |     1|  5-  5|F1.0  |
|QUES06  |     2|  1-  1|F1.0  |
|QUES07  |     2|  2-  2|F1.0  |
|QUES08  |     2|  3-  3|F1.0  |
|QUES09  |     2|  4-  4|F1.0  |
|QUES10  |     2|  5-  5|F1.0  |
+--------+------+-------+------+
```

3.1.2 DATA LIST FREE

syntaxe

DATA LIST FREE
('c1','c2',...)
{NOTABLE,TABLE}
FILE='filename'
/var_list [format_spec] (ou /var_list *).

Description

La commande DATA LIST FREE est utilisée pour lire des fichiers où les données sont écrites dans un format "libre", c'est-à-dire des fichiers où les données n'ont pas d'emplacement précis dans une ligne. De plus, le nombre de lignes pour un enregistrement n'est pas défini. En format libre, le fichier de données est structuré comme une série de caractères groupés, séparés par des espaces, des tabulations, des virgules ou des retours à la ligne. Un espace simple entre deux groupes de caractères est considéré comme n'appartenant à aucun groupe. Les suites d'espaces, tabulations et retours a la ligne sont équivalents à des espaces simples dans le rôle de séparation des groupes de caractères.

Des délimiteurs de groupes peuvent être spécifiés explicitement par l'utilisateur. Ces derniers doivent être des chaînes à un seul caractère. Immédiatement après le mot-clé FREE, il faut écrire la liste des nouveaux délimiteurs entre parenthèses, et séparés par des virgules. Si des délimiteurs sont explicitement spécifiés, seulement les caractères donnés et les retours à la ligne peuvent séparer les groupes.

Les sous-commandes *NOTABLE* et *TABLE* sont les mêmes que pour la commande *DATA LIST FIXED* (cf. section 3.1.1, page 13). Par défaut, la sous-commande NOTABLE est active.

On définit les variables en écrivant la liste des noms des variables après un slash ('/'). Après chaque nom de variable, on peut spécifier le format (FORTRAN) de la variable entre parenthèses. Le symbole '*' peut être utilisé pour indiquer que toutes les variables qui précèdent sont au format '`F8.0`'. Si rien n'est spécifié après les noms de variables, le format par défaut est '`F8.0`'.

Exemple1

```
DATA LIST FREE ('>')
   TABLE
   FILE=exemple3
   /SEXE (A1) AGE (F2.0) TAILLE (F4.2).
```

Données du fichier 'exemple3' :

```
h>20>1.85>f>35>1.67>f>42>1.71>h>23>1.95>h>13>1.70>f>18>1.65
f>16>1.70>h>36>1.82>h>47
1.86>f>24>1.72>f>31>1.62>h>52>1.93
```

À partir des données ci-dessus, cette commande définit les variables suivantes :
- SEXE : chaîne de un caractère ;
- AGE : nombre entier à deux chiffres ;
- TAILLE : nombre décimal à trois chiffres, dont deux après la virgule.

Les valeurs des variables sont séparées par le caractère '>'.

La sous-commande TABLE provoque la sortie suivante :

```
1.1 DATA LIST.  Reading free-form data from file exemple3.
+--------+------+
|Variable|Format|
#========#======#
|SEXE    |A1    |
|AGE     |F2.0  |
|TAILLE  |F4.2  |
+--------+------+
```

Exemple 2

```
DATA LIST FREE
   FILE=exemple3bis
   /X1 TO X4.
```

Données du fichier exemple3bis :

```
1 4  52 5  5  6 61 4 7 5 12 5 1 5
85 4 1   2 72 3   1 4 23 5
```

Cet exemple illustre la lecture des données en format "libre". À partir des données ci-dessus, les variables X1, X2, X3 et X4 sont créées (elles prennent le format '`F8.0`'). À titre d'exemple, la variable X2 prend les valeurs `4 6 5 5 2 4`.

3.1.3 DATA LIST LIST

Syntaxe

DATA LIST LIST
('c1','c2',...)
{NOTABLE,TABLE}
FILE='filename'
/var_list [format_spec] (ou /var_list *).

Description

La commande DATA LIST LIST est équivalente à la commande *DATA LIST FREE* (cf. section 3.1.2, page 16). La seule différence est que chaque enregistrement doit correspondre à exactement une ligne.

3.2 BEGIN DATA

Syntaxe

BEGIN DATA.
...
END DATA.

Description

La commande BEGIN DATA est utilisée pour entrer des données ASCII dans un fichier syntaxe à exécuter avec PSPP. Cette commande doit être précédée par la commande *DATA LIST*, qui permet de définir au préalable les variables à lire (cf. section 3.1, page 13). L'utilisation de BEGIN DATA est compatible avec les trois commandes DATA LIST, les données tapées dans le fichier syntaxe sont lues comme les données d'un fichier input. Si la commande BEGIN DATA est utilisée, aucun fichier input ne doit être spécifié.

Exemple

```
DATA LIST LIST NOTABLE /SEXE (A1) AGE (F3) INFO1 TO INFO3 (F1).
BEGIN DATA.
   f  24 2  3 1
 h  37  1  1 2
  h  31  1  3  2
   f 43  3   3 1
END DATA.
```

La commande BEGIN DATA définit quatre observations :

observation	SEXE	AGE	INFO1	INFO2	INFO3
1	f	24	2	3	1
2	h	37	1	1	2
3	h	31	1	3	2
4	f	43	3	3	1

3.3 FILE HANDLE

Syntaxe

FILE HANDLE handle_name
/NAME='filename'
/MODE={CHARACTER,IMAGE}
/LRECL=rec_length.

Description

On utilise la commande FILE HANDLE pour associer un identifiant (*handle name*) à un fichier. Dans les sous-commandes accédant à des fichiers, on peut ainsi spécifier un fichier en se référant à son nom de fichier handle. Comme les noms des fichiers texte peuvent être directement spécifiés dans ces sous-commandes, FILE HANDLE n'est nécessaire que pour les fichiers ne contenant pas des lignes de texte. Cependant, il peut être pratique d'utiliser la commande FILE HANDLE pour des fichiers textes car elle permet d'y faire référence de manière plus simple.

Un nom de fichier handle ne doit être utilisé que pour un seul fichier. Dans la ligne de commande, le nom de l'identifiant vient immédiatement après FILE HANDLE.

La seule sous-commande obligatoire est *NAME*. Elle permet de préciser le nom du fichier associé au fichier handle. Ici, les apostrophes sont obligatoires pour spécifier le nom du fichier.

La sous-commande *MODE* spécifie le type de fichier auquel on fait référence avec le fichier handle. Le mode par défaut est le mode *CHARACTER*, le fichier de données est ouvert en mode texte ANSI C, de telle sorte que chaque ligne de texte soit lue comme un enregistrement.

Le mode *IMAGE* est utilisé pour ouvrir des fichiers de données en mode binaire ANSI C, les enregistrements ont une longueur fixe. Si on choisit le mode IMAGE, il faut préciser la longueur des enregistrements avec la sous-commande *LRECL*. Par défaut, la longueur d'un enregistrement est de 1024 octets.

Exemple

```
FILE HANDLE EX5
   /NAME='exemple5'
   /MODE=CHARACTER.
DATA LIST FILE=EX5 /X1 TO X7 1-35.
```

La commande FILE HANDLE définit EX5 comme le fichier handle du fichier 'exemple5'. La sous-commande MODE indique que le fichier contient des données en mode CHARACTER. La sous-commande FILE de la commande DATA LIST fait référence au fichier handle défini précédemment.

4 Les sorties de données

4.1 LIST

Syntaxe

LIST
[/VARIABLES=]var_list
/CASES=FROM start_index TO end_index BY incr_index
/FORMAT={UNNUMBERED,NUMBERED}.

Description

La commande LIST affiche dans le fichier listing (*pspp.list*) les valeurs des variables spécifiées. La sous-commande *VARIABLES* indique la liste des variables dont les valeurs doivent être affichées (le mot-clé VARIABLES est facultatif). Si cette sous-commande n'est pas précisée, alors les valeurs de toutes les variables sont affichées.

La sous-commande *CASES* peut être utilisée pour préciser l'ensemble des observations à afficher. On écrit *FROM* et le numéro de la première observation à afficher, puis *TO* et le numéro de la dernière observation à afficher. *BY* indique le nombre d'observations à passer entre deux affichages. Si CASES n'est pas spécifiée, alors toutes les observations sont affichées.

La sous-commande *FORMAT* permet de modifier la sortie : *NUMBERED* affiche le numéro des observations. Par défaut, les numéros ne sont pas précisés (*UNNUMBERED*).
La première observation porte le numéro 1.

Exemple

```
DATA LIST LIST NOTABLE /SEXE (A1) AGE (F2) INFO1 TO INFO3 (F1).
BEGIN DATA.
f  24  2 3 1
   h  37 1 1 2
 h  31 1  3 2
f 43 3  3 1
 f    14 1  2 2
  h  50  2  3 3
 f  22  3 3 2
END DATA.
LIST SEXE AGE INFO3 /CASES=FROM 1 TO 7 BY 2 /FORMAT=NUMBERED.
```

On imprime une observation sur deux pour les variables SEXE, AGE et INFO3. Le numéro de ces observations est précisé à l'aide du mot-clé *NUMBERED*.

On obtient la sortie suivante :

```
Case# SEXE AGE INFO3
----- ---- --- -----
    1    f  24     1
    3    h  31     2
    5    f  14     2
    7    f  22     2
```

4.2 PRINT

Syntaxe

PRINT
OUTFILE='filename'
RECORDS=n_lines
{NOTABLE,TABLE}
/'string' [start-end]
ou var_list start-end [format_spec]
ou var_list (fortran_spec)
ou var_list *.

Description

La commande PRINT affiche dans un fichier output les valeurs prises par les variables pour chaque observation. PRINT est exécutée lorsqu'une commande entraine l'analyse des données. Pour exécuter PRINT sans spécifier une telle commande, on fait suivre PRINT par la commande *EXECUTE*.
Toutes les sous-commandes sont optionnelles.

La sous-commande *OUTFILE* spécifie le fichier où la sortie est imprimée. On doit indiquer le nom du fichier, qui peut être un fichier *handle* (cf. commande *FILE HANDLE*, section 3.3, page 20). Si le fichier sortie n'est pas précisé, les valeurs sont affichées dans le fichier listing (*pspp.list*).

RECORDS indique le nombre de lignes affichées par observation.

TABLE entraine l'affichage dans le fichier listing d'un tableau décrivant ce qui est imprimé dans le fichier sortie. Par défaut, le tableau n'est pas affiché (*NOTABLE*).

Un enregistrement peut être affiché sur plusieurs lignes. On introduit chaque ligne d'un enregistrement avec un slash ('/'), suivi des noms des variables à afficher sur cette ligne.

Il est possible d'afficher des chaînes de caractères sur les lignes d'un enregistrement. On peut spécifier la position de ces chaînes de caractères en indiquant la colonne de départ et la colonne de fin.

Les variables à afficher peuvent être spécifiées de la même manière qu'elles le sont dans la commande DATA LIST FIXED, cf. section 3.1.1, page 13. De plus, si la liste des variables est suivie par le symbole '*', les variables sont alors affichées avec le format du dictionnaire, et séparées par des espaces. Si la liste des variables est terminée par un slash ou un point, l'affichage est le même qu'avec le signe '*'.

Exemple 1

```
DATA LIST NOTABLE FILE=exemple6 /SEXE 1 (A) AGE 3-4 TAILLE 6-8 (2).
PRINT TABLE /SEXE TAILLE *.
EXECUTE.
```

Données du fichier 'exemple6' :

```
h 20 185
f 35 167
f 42 171
h 23 195
h 13 170
f 18 165
f 16 170
h 36 182
```

Cette commande entraine l'affichage d'un tableau descriptif et des valeurs prises par les variables SEXE et TAILLE. Cet affichage se fait dans le fichier listing.

On obtient la sortie suivante :

```
1.1 PRINT.  Writing 1 record(s) to the listing file.
+--------+------+-------+------+
|Variable|Record|Columns|Format|
#========#======#=======#======#
|SEXE    |     1|  1-  1|A1    |
|TAILLE  |     1|  3-  6|F4.2  |
+--------+------+-------+------+
h 1.85
f 1.67
f 1.71
h 1.95
h 1.70
f 1.65
f 1.70
h 1.82
```

Exemple 2

```
DATA LIST NOTABLE /SEXE 1 (A) AGE 3-4 TAILLE 6-8 (2).
BEGIN DATA.
h 20 185
f 35 167
f 42 171
h 23 195
END DATA.
PRINT OUTFILE=exemple7 RECORDS=3 /SEXE 1 (A) /AGE 3-4 /TAILLE 6-9 (2).
EXECUTE.
```

La sortie est imprimée dans le fichier 'exemple7'. Un enregistrement est composé de trois lignes.

La sortie obtenue est la suivante :

```
h
  20
     1.85
f
  35
     1.67
f
  42
     1.71
h
  23
     1.95
```

4.3 WRITE

Syntaxe

WRITE
OUTFILE='filename'
RECORDS=n_lines
{NOTABLE,TABLE}
/'string' [start-end]
ou var_list start-end [format_spec]
ou var_list (fortran_spec)
ou var_list *.

Description

La commande WRITE permet d'écrire dans un fichier sortie des données binaires ou du texte. Cette commande est similaire à la commande PRINT

(cf. section 4.2, page 22) au niveau syntaxe et utilisation. La principale différence entre ces deux commandes est que PRINT utilise des formats print (*print formats*), et WRITE des formats write (*write formats*).

WRITE est exécutée lorsqu'une commande entraine l'analyse des données. Pour exécuter WRITE sans spécifier une telle commande, on fait suivre WRITE par la commande *EXECUTE*.

Se référer à la commande *FILE HANDLE* (section 3.3, page 20) pour des informations sur la manière de déclarer un fichier binaire.

5 La manipulation des variables

Cette section présente des commandes utiles pour la gestion des variables.

5.1 DISPLAY

Syntaxe

DISPLAY
{NAMES,INDEX,LABELS,VARIABLES,DICTIONARY}
[var_list].

Description

La commande DISPLAY affiche dans le fichier listing les informations demandées sur les variables du dictionnaire. On peut choisir de décrire seulement certaines variables en indiquant la liste de leurs noms à la fin de la ligne de commande.

Les variables sont affichées dans l'ordre où elles apparaissent dans le dictionnaire.

Un des mots-clés de la liste suivante peut être utilisé pour spécifier les informations à afficher :

- *NAMES* : les noms des variables sont affichés.
- *INDEX* : les noms des variables et leurs positions dans le dictionnaire sont affichés.
- *LABELS* : les noms des variables, les positions et les labels des variables (cf. commande *VARIABLE LABELS*, section 5.2, page 27) sont affichés.
- *VARIABLES* : les noms des variables, les positions, les formats (cf. commande *FORMATS*, section 5.9, page 34) et les valeurs manquantes (cf. commande *MISSING VALUES*, section 5.6, page 31) sont affichés.
- *DICTIONARY* : les noms des variables, les positions, les formats, les valeurs manquantes, les labels des variables et les labels des valeurs (cf. commande *VALUE LABELS*, section 5.3, page 28) sont affichés.

Exemple

```
DATA LIST NOTABLE
 FILE=exemple8
/NOM 1-10 (A) PRENOM 12-21 (A) SEXE 23 AGE 25-27 TAILLE 29-31 (2).
DISPLAY VARIABLES NOM TO SEXE TAILLE.
```

Cette commande affiche la description des variables NOM, PRENOM, SEXE et TAILLE. On obtient la sortie suivante :

```
1.1 DISPLAY.
+--------+-----------------------------------------+--------+
|Variable|Description                              |Position|
#========#=========================================#========#
|NOM     |Format: A10                              |       1|
+--------+-----------------------------------------+--------+
|PRENOM  |Format: A10                              |       2|
+--------+-----------------------------------------+--------+
|SEXE    |Format: F1.0                             |       3|
+--------+-----------------------------------------+--------+
|TAILLE  |Format: F4.2                             |       5|
+--------+-----------------------------------------+--------+
```

5.2 VARIABLE LABELS

Syntaxe

VARIABLE LABELS
    var_list 'var_label'
    [/var_list 'var_label']
    ...
    [/var_list 'var_label'].

Description

VARIABLE LABELS associe un nom, qui peut avoir un rôle explicatif, à une ou plusieurs variables. Ce nom, appelé *label*, est affiché avec les résultats des analyses statistiques.

Pour donner un nom à un groupe de variables, on écrit la liste des variables que l'on fait suivre du nom, spécifié comme une chaîne de caractères. On peut donner des labels à plusieurs variables, ou groupes de variables, dans une même commande en séparant les listes de variables avec un slash ('/').

Exemple

```
DATA LIST LIST NOTABLE /INFO1 TO INFO3 *.
BEGIN DATA.
24 161 50
 37 170 55
  31 165 48
   43 183 75
END DATA.
VARIABLE LABELS INFO1 'Age' /INFO2 'Taille en cm' /INFO3 'Poids en kg'.
DISPLAY LABELS.
```

On assigne des labels aux variables INFO1, INFO2 et INFO3. La commande DISPLAY affiche la sortie suivante dans le fichier listing :

```
1.1 DISPLAY.
+--------+-----------------------------------------+--------+
|Variable|Label                                    |Position|
#========#=========================================#========#
|INFO1   |Age                                      |       1|
+--------+-----------------------------------------+--------+
|INFO2   |Taille en cm                             |       2|
+--------+-----------------------------------------+--------+
|INFO3   |Poids en kg                              |       3|
+--------+-----------------------------------------+--------+
```

5.3 VALUE LABELS

Syntaxe

VALUE LABELS
/var_list value 'label' [value 'label']...
/...

Description

VALUE LABELS permet d'associer des "étiquettes" (ou labels) aux valeurs de variables numériques ou de type *short string* (cf. section 2.8.1, page 11). Les chaînes de caractères longues ne peuvent pas avoir de label.

Pour donner des labels à un ensemble de valeurs, il faut spécifier la liste des noms des variables dont les valeurs possibles sont les mêmes après un slash ('/'), suivie de la liste des valeurs et de leurs labels.

Si une valeur possède déjà un label, le nouveau label écrase l'ancien. On peut utiliser la commande *ADD VALUE LABELS* (cf. section 5.4, page 29) pour ajouter un label sans effacer le label déjà existant.

Exemple

```
DATA LIST NOTABLE FILE=exemple10 /SEXE 1 CAT 4.
VARIABLE LABELS CAT 'Categorie socio-professionnelle'.
VALUE LABELS
    /SEXE 0 'Homme' 1 'Femme'
    /CAT 1 'Agriculteur' 2 'Artisan' 3 'Commercant' 4 'Chef d''entreprise'
         5 'Profession liberale' 6 'Cadre' 7 'Profession intermediaire'
         8 'Employe' 9 'Ouvrier'.
DISPLAY DICTIONARY.
```

On attribue le label 'Categorie socio-professionnelle' à la varible CAT. La commande VALUE LABELS explique les valeurs prises par les variables. La commande DISPLAY DICTIONARY donne la sortie suivante :

```
1.1 DISPLAY.
+--------+----------------------------------------+--------+
|Variable|Description                             |Position|
#========#========================================#========#
|SEXE    |Format: F1.0                            |       1|
|        +-----+----------------------------------+        |
|        |    0|Homme                             |        |
|        |    1|Femme                             |        |
+--------+-----+----------------------------------+--------+
|CAT     |Categorie socio-professionnelle         |       2|
|        |Format: F1.0                            |        |
|        +-----+----------------------------------+        |
|        |    1|Agriculteur                       |        |
|        |    2|Artisan                           |        |
|        |    3|Commercant                        |        |
|        |    4|Chef d'entreprise                 |        |
|        |    5|Profession liberale               |        |
|        |    6|Cadre                             |        |
|        |    7|Profession intermediaire          |        |
|        |    8|Employe                           |        |
|        |    9|Ouvrier                           |        |
+--------+-----+----------------------------------+--------+
```

5.4 ADD VALUE LABELS

Syntaxe

ADD VALUE LABELS
/var_list value 'label' [value 'label']...
/...

Description

ADD VALUE LABELS a la même syntaxe et le même rôle que VALUE LABELS (cf. section 5.3, page 28), la seule différence est que si un label existe déjà pour une valeur, le nouveau label n'efface pas l'ancien.

Exemple

```
DATA LIST FREE FILE=exemple11 /PAYS1 TO PAYS3 (A3).
ADD VALUE LABELS PAYS1 TO PAYS3
'FRA' 'France' 'ESP' 'Espagne' 'ALL' 'Allemagne'
'ITA' 'Italie' 'POR' 'Portugal'.
```

Données du fichier exemple11 :

```
FRA ITA POR
ESP FRA ESP ALL
ALL ITA
```

Les valeurs possibles pour les variables PAYS1, PAYS2 et PAYS3 sont 'FRA', 'ESP', 'ALL', 'ITA', et 'POR'. À chaque valeur est attribué un label.

5.5 RENAME VARIABLES

Syntaxe

RENAME VARIABLES
(old_names=new_names) ...

Description

La commande RENAME VARIABLES permet de changer le nom d'une ou de plusieurs variables. Il faut spécifier entre parenthèses la liste des anciens noms et la liste des nouveaux noms, en les séparant avec le signe '='. À l'intérieur des parenthèses, il doit y avoir autant de noms anciens que de noms nouveaux. On peut spécifier plusieurs groupes de noms entre parenthèses.

RENAME VARIABLES prend effet immédiatement.

Exemple1

```
DATA LIST
   FILE=exemple1
   /NOM 1-10 (A) PRE 12-21 (A) SEX 23 AGE 25-27 TAI 29-31 (2).
RENAME VARIABLES (PRE SEX TAI=PRENOM SEXE TAILLE).
DISPLAY NAMES.
```

Exemple2

```
DATA LIST
   FILE=exemple1
   /NOM 1-10 (A) PRE 12-21 (A) SEX 23 AGE 25-27 TAI 29-31 (2).
RENAME VARIABLES (PRE=PRENOM) (SEX=SEXE) (TAI=TAILLE).
DISPLAY NAMES.
```

Les deux exemples effectuent les mêmes changements de noms de deux manières différentes. PRE devient PRENOM, SEX devient SEXE et TAI devient TAILLE.

On obtient donc la même sortie pour les deux exemples :

```
1.1 DATA LIST.  Reading 1 record from file exemple1.
+--------+------+-------+------+
|Variable|Record|Columns|Format|
#========#======#=======#======#
|NOM     |     1|  1- 10|A10   |
|PRE     |     1| 12- 21|A10   |
|SEX     |     1| 23- 23|F1.0  |
|AGE     |     1| 25- 27|F3.0  |
|TAI     |     1| 29- 31|F3.2  |
+--------+------+-------+------+

Variable
---------
NOM
PRENOM
SEXE
AGE
TAILLE
```

5.6 MISSING VALUES

Syntaxe

MISSING VALUES
var_list (missing_values).

'missing_values' peut prendre une des formes suivantes :
num1
num1, num2
num1, num2, num3
num1 THRU num2
num1 TRHU num2, num3
string1
string1, string2
string1, string2, string3

Description

La commande MISSING VALUES permet à l'utilisateur de déclarer des valeurs manquantes pour les variables numériques ou de type string à moins de huit caractères. Ainsi, ces valeurs déclarées manquantes sont exclues des analyses.

On spécifie une variable, ou une liste de variables, suivie de la liste des valeurs manquantes entre parenthèses. La liste des valeurs manquantes peut comprendre

jusqu'à trois valeurs discrètes, ou une suite de valeurs numériques définie par le mot-clé *THRU* (qui peut être accompagnée d'une valeur numérique discrète).

Les mots-clés *LOWEST* et *HIGHEST*, dont les abréviations respectives sont *LO* et *HI*, peuvent être utilisés en combinaison avec THRU pour indiquer la valeur numérique la plus faible ou la plus élevée.

Cette commande prend effet immédiatement.

Exemple

```
DATA LIST NOTABLE FILE=exemple14 /Q1 TO Q6 1-6.
MISSING VALUES Q1 TO Q3 (9) Q4 Q6 (5 THRU HI, 0).
DISPLAY VARIABLES.
```

La valeur 9 est déclarée manquante pour les variables numériques Q1, Q2 et Q3. Pour les variables Q4 et Q6, les valeurs supérieures ou égales 5, ainsi que la valeur 0, sont déclarées manquantes.

DISPLAY VARIABLES permet d'afficher la sortie suivante :

```
1.1 DISPLAY.
+--------+-----------------------------------------+--------+
|Variable|Description                              |Position|
#========#=========================================#========#
|Q1      |Format: F1.0                             |       1|
|        |Missing Values: 9                        |        |
+--------+-----------------------------------------+--------+
|Q2      |Format: F1.0                             |       2|
|        |Missing Values: 9                        |        |
+--------+-----------------------------------------+--------+
|Q3      |Format: F1.0                             |       3|
|        |Missing Values: 9                        |        |
+--------+-----------------------------------------+--------+
|Q4      |Format: F1.0                             |       4|
|        |Missing Values: 5 THRU HIGHEST; 0        |        |
+--------+-----------------------------------------+--------+
|Q5      |Format: F1.0                             |       5|
+--------+-----------------------------------------+--------+
|Q6      |Format: F1.0                             |       6|
|        |Missing Values: 5 THRU HIGHEST; 0        |        |
+--------+-----------------------------------------+--------+
```

5.7 NUMERIC

Syntaxe

NUMERIC
/var_list [(format_spec)]
/...

Description

NUMERIC permet de déclarer de nouvelles variables numériques, avec la possibilité de spécifier leurs formats. Ces nouvelles variables peuvent ensuite être utilisées par certaines commandes sans qu'une valeur ne leur soit attribuée.

Les variables créées avec NUMERIC sont initialisées à une valeur appelée *system-missing value* (cf. section 2.7, page 10).

On spécifie la liste des nouvelles variables numériques après un slash ('/'). On peut préciser le format des variables en donnant le format FORTRAN entre parenthèses après la liste des noms. Par défaut, le format est `F8.2`.

Si on précise le format, les variables ayant des formats différents sont séparées par un slash.

Exemple

```
NUMERIC X1 X2 (F5.1) /X3 TO X6 (F3).
```

On crée les variables numériques X1, X2 au format `F5.1`, et X3, X4, X5, X6 au format `F3.0`.

5.8 STRING

Syntaxe

STRING
/var_list (format_spec)
/...

Description

STRING crée une nouvelle variable de type string à utiliser pour les transformations de données.

Les variables créées sont initialisées à un ensemble d'espaces.

Il faut spécifier après un slash ('/') les noms des variables de type string à créer et leur format entre parenthèses. Pour créer des variables de formats différents, on sépare deux groupes avec un slash.

Exemple

```
STRING NOM (A20) /PRENOM1 TO PRENOM3 (A10).
```

STRING déclare la variable NOM au format A20 et les variables PRENOM1, PRENOM2, PRENOM3 au format `A10`.

5.9 FORMATS

Syntaxe

FORMATS
var_list (format_spec) ...

Description

La commande FORMATS change les formats *print* et *write* des variables numériques spécifiées.

On spécifie une liste de variables suivie du nouveau format entre parenthèses.
Les listes de variables correspondant à des formats différents sont séparées par un espace vide.
Pour changer seulement les formats print ou les formats write, on utilise respectivement les commandes *PRINT FORMATS* et *WRITE FORMATS*, avec une syntaxe identique à celle de la commande FORMATS.
La commande FORMATS prend effet immédiatement.

Exemple

```
FORMATS V1 TO V5 (F5.2) V6 (F3).
```

Le format des variables V1, V2, V3, V4, V5 devient `F5.2`, et le format de V6 devient `F3`.

6 La transformation des données

Cette section décrit les commandes essentielles pour la manipulation et la préparation des données aux analyses ultérieures.

6.1 COMPUTE

Syntaxe

COMPUTE
variable=expression.

Description

COMPUTE crée une variable numérique ou modifie la valeur d'une variable numérique ou de type string. Pour chaque observation, la valeur de l'expression à droite du signe '=' est assignée à la variable spécifiée à gauche. Contrairement aux variables de type string, les variables numériques n'ont pas besoin d'être déclarées avant d'être spécifiées dans la commande COMPUTE. Lorsque COMPUTE crée une variable, celle-ci prend le format `F8.2`.

L'expression à droite du signe égal peut faire appel à toutes les expressions mathématiques offertes par PSPP. Pour plus d'informations sur les fonctions mathématiques proposées, voir *LES EXPRESSIONS MATHEMATIQUES*, section 11, page 82.

Exemple

```
DATA LIST NOTABLE /X1 1-3.
BEGIN DATA.
635
210
150
323
END DATA.
COMPUTE X2=X1*2.
COMPUTE X1=X1/2.
FORMATS X2 (F4) X1 (F5.1).
LIST.
```

Pour chaque valeur de X1, on calcule la valeur prise par X2, nouvelle variable qui correspond au double de X1. Les nouvelles valeurs de la variable X1 sont les anciennes valeurs divisées par 2. On spécifie ensuite un nouveau format pour ces deux variables.

LIST entraine la sortie suivante :

```
   X1   X2
----- ----
317.5 1270
105.0  420
 75.0  300
161.5  646
```

6.2 COUNT

Syntaxe

COUNT
var_name=var_list(value...)
/...

'value...' est un ensemble de valeurs qui peuvent prendre les formes suivantes :
number
string
num1 THRU num2
MISSING
SYSMIS

'num1' et 'num2' peuvent être remplacés respectivement par *LOWEST* (ou *LO*) et *HIGHEST* (ou *HI*), pour désigner la valeur la plus faible et la valeur la plus élevée.

Description

COUNT crée une variable numérique qui compte pour chaque observation le nombre d'apparitions d'une ou plusieurs valeurs parmi une liste de variables. La variable ainsi créée est appelée variable *cible*.

Les variables cibles sont définies au format `F8.2` et sont initialisées à 0.

Les variables servant au comptage des valeurs sont appelées variables tests. Les variables numériques et les chaînes de caractères, courtes et longues, peuvent être des variables tests.

Les valeurs manquantes définies par l'utilisateur (*user-missing values*, cf. commande *MISSING VALUES*, section 5.6, page 31) qui apparaissent dans les variables tests sont traitées comme les autres valeurs. Si ces valeurs manquantes sont spécifiées comme des valeurs recherchées, elles sont comptées comme les autres valeurs.

On doit spécifier la variable cible, suivie d'un signe égal ('='), puis la liste des variables tests, suivie de l'ensemble des valeurs à compter entre parenthèses.

Pour séparer les spécifications des variables cibles, on utilise un slash ('/').

Le mot-clé *SYSMIS* est utilisé pour compter les valeurs *system-missing* (cf. section 2.7, page 10) des variables numériques.

Le mot-clé *MISSING* permet de compter à la fois les valeurs *system-missing* et *user-missing*.

Exemple

```
DATA LIST NOTABLE /X1 TO X5 1-10.
BEGIN DATA.
 1-120  13
20-3-54220
1520-6-314
END DATA.
MISSING VALUES X1 TO X5 (LO THRU 0).
COUNT Y1=X1 TO X5(LO THRU 0) /Y2=X1 TO X5(20).
FORMATS Y1 Y2 (F2).
LIST.
```

Les valeurs négatives et la valeur 0 sont déclarées manquantes pour toutes les variables avec la commande MISSING VALUES. Ces valeurs manquantes sont comptées avec la commande COUNT et le résultat est transféré dans la variable Y1. De même, la valeur 20 est comptée et le résultat transféré dans la variable Y2.

FORMATS permet de modifier le format des variables Y1 et Y2 : initialement `F8.2`, le format devient `F2`.

Enfin, LIST affiche les valeurs de toutes les variables :

```
X1 X2 X3 X4 X5 Y1 Y2
-- -- -- -- -- -- --
 1 -1 20  . 13  1  1
20 -3 -5 42 20  2  2
15 20 -6 -3 14  2  1
```

6.3 RECODE

Syntaxe

RECODE
var_list (value_list=dest_value)... [INTO var_list]
/...

'value_list' peut prendre une des formes suivantes :
number
string
num1 THRU num2
MISSING
SYSMIS
ELSE

'num1' et 'num2' peuvent être remplacés respectivement par *LOWEST* (ou *LO*) et *HIGHEST* (ou *HI*), pour désigner la valeur la plus faible et la valeur la plus élevée.

'dest_value' peut prendre une des formes suivantes :
number
string
SYSMIS
COPY

Description

La commande RECODE est utilisée pour modifier, arranger les valeurs d'une liste de variables. Le recodage des données se fait de la manière suivante : on spécifie d'abord la liste des variables concernées par le changement, puis entre parenthèses les valeurs d'origine que l'on va modifier, suivies de la nouvelle valeur qui leur est assignée. Les anciennes valeurs sont séparées des nouvelles par un signe égal ('='). Pour une même liste de variables, plusieurs nouvelles valeurs peuvent être données. Les différents recodages sont séparés par un slash ('/').

Le mot-clé *INTO* est utilisé pour copier les nouvelles données dans de nouvelles variables. Il doit y avoir autant de variables d'origine que de nouvelles variables. Les nouvelles variables de type string doivent être déclarées au préalable (avec la commande *STRING*, par exemple), alors que les nouvelles variables numériques peuvent être créées par la commande.

Quand INTO n'est pas spécifié, les variables d'origine doivent être du même type que les nouvelles variables. Dans la cas contraire, les variables de type string peuvent être recodées en variables numériques et inversement.

Le mot-clé *MISSING* fait référence aux valeurs *user-missing* et *system-missing* et peut être utilisé seulement dans la liste des valeurs d'origine.

Le mot-clé *SYSMIS* fait référence aux valeurs *system-missing* uniquement et peut être utilisé comme une valeur d'origine ou comme une nouvelle valeur.

ELSE représente toutes les valeurs d'origine auxquelles on n'a pas attribué de nouvelles valeurs.

COPY copie les valeurs d'origine dans les nouvelles valeurs sans les recoder.

Exemple 1

```
DATA LIST NOTABLE FILE=exemple21 /SEXE 1-5 (A).
RECODE SEXE ('homme'=1) ('femme'=2) (ELSE=SYSMIS) INTO NUMSEXE.
FORMATS NUMSEXE (F1).
LIST.
```

La variable SEXE de type string est recodée en une variable numérique appelée NUMSEXE dont le format est définit avec la commande FORMATS. Les valeurs 'homme' et 'femme' deviennent respectivement les valeurs 1 et 2. Toutes les valeurs autres que 'homme' et 'femme' deviennent *system-missing*.

LIST affiche les valeurs des variables SEXE et NUMSEXE :

```
 SEXE NUMSEXE
----- -------
homme       1
femme       2
toto        .
femme       2
homme       1
tutu        .
homme       1
femme       2
```

Exemple 2

```
DATA LIST NOTABLE FILE=exemple22 /X1 TO X3 1-3.
RECODE X1 TO X3 (1 THRU 4 = 1) (6 THRU 9 = 2) (ELSE=COPY) INTO Y1 TO Y3.
FORMATS Y1 TO Y3 (F1).
LIST.
```

Les variables X1, X2 et X3 sont recodées en Y1, Y2 et Y3. Les valeurs 1, 2, 3, 4 deviennent la valeur 1 ; les valeurs 6, 7, 8, 9 deviennent la valeur 2 ; les autres valeurs sont inchangées.

On obtient la sortie suivante :

```
X1 X2 X3 Y1 Y2 Y3
-- -- -- -- -- --
 1  1  5  1  1  5
 2  4  6  1  1  2
 9  8  8  2  2  2
 0  1  0  0  1  0
 2  5  5  1  5  5
```

6.4 AUTORECODE

Syntaxe

AUTORECODE
VARIABLES=var_list INTO dest_vars
/DESCENDING.

Description

La commande AUTORECODE transforme les valeurs prises par une variable, numérique ou string, en entiers consécutifs, et crée une variable numérique avec ces nouvelles valeurs. Le recodage se fait dans l'ordre ascendant pour les variables numériques, et dans l'ordre alphabétique pour les chaînes de caractères.

Cette commande est similaire à la commande *RECODE* (cf. section 6.3, page 37), mais le recodage des valeurs se fait de manière automatique.

Le mot-clé *VARIABLES* est obligatoire et suivi d'un signe égal ('='). Il doit y avoir autant de variables d'origine que de variables recodées.

Il est possible de changer l'ordre de recodage en utilisant *DESCENDING* : le recodage des valeurs se fait alors dans l'ordre descendant.

Exemple

```
DATA LIST NOTABLE /COULEUR 1-10 (A).
BEGIN DATA.
rouge
bleu
vert
orange
jaune
violet
END DATA.
AUTORECODE VARIABLES=COULEUR INTO NUMCOUL /DESCENDING.
FORMATS NUMCOUL (F1).
LIST.
```

Les couleurs sont transformées en nombres consécutifs dans la nouvelle variable NUMCOUL. Le recodage se fait dans l'ordre inverse de l'ordre alphabétique.

On obtient la sortie suivante :

```
   COULEUR NUMCOUL
---------- -------
rouge            3
bleu             6
vert             2
orange           4
jaune            5
violet           1
```

6.5 IF

Syntaxe

IF
condition variable=expression.

Description

IF assigne la valeur d'une expression à une variable si la condition spécifiée est vérifiée. Cette condition est définie par une expression qui renvoie une valeur de type booléen (cf. *Les booléens*, section 11.1, page 82). Si la valeur renvoyée est 'VRAI', la condition est satisfaite, et la valeur de la deuxième expression est calculée et assignée à la variable cible. Si la valeur renvoyée est 'FAUX' ou 'missing', la condition n'est pas satisfaite, et la variable cible n'est pas modifiée.

La commande IF permet d'assigner des valeurs aux variables numériques et de type string. Les variables de type string doivent être déclarées (avec la commande *STRING*, par exemple) avant qu'une valeur ne leur soit assignée avec IF. Les types de la variable cible et de l'expression doivent être les mêmes.

Exemple

```
DATA LIST NOTABLE /SEXE 1 AGE 3-4.
BEGIN DATA.
1 35
2 14
2 25
1 16
2 36
END DATA.
STRING GROUPE (A10).
IF (SEXE=1 AND AGE>18) GROUPE='Homme'.
IF (SEXE=2 AND AGE>18) GROUPE='Femme'.
LIST.
```

La première étape est de déclarer la nouvelle variable GROUPE de type string à l'aide de la commande STRING (cf. section 5.8, page 33). Pour chaque observation, on vérifie si la condition spécifiée est satisfaite. Si elle l'est, on assigne une valeur à la variable GROUPE pour cette observation.

LIST affiche les valeurs des variables SEXE, AGE et GROUPE :

```
SEXE AGE     GROUPE
---- --- ----------
   1  35 Homme
   2  14
   2  25 Femme
   1  16
   2  36 Femme
```

6.6 SORT CASES

Syntaxe

```
SORT CASES
        BY var_list.
```

Description

SORT CASES permet de réordonner les observations en fonction des valeurs d'une ou plusieurs variables. Par défaut, les valeurs sont rangées dans l'ordre ascendant (ou alphabétique pour les chaînes de caractères). Il est possible de préciser l'ordre souhaité en spécifiant *(A)*, ou *(UP)*, pour ascendant, et *(D)*, ou *(DOWN)*, pour descendant.

Exemple

```
DATA LIST NOTABLE FILE=exemple25 /NOM (A10) PRENOM (A10) AGE (F2).
SORT CASES BY NOM (A) PRENOM (A).
LIST.
```

Les valeurs de la variable NOM sont rangées dans l'ordre alphabétique. Si deux noms sont identiques, les valeurs de la variable PRENOM sont aussi rangées dans l'ordre alphabétique.

La sortie est la suivante :

```
       NOM     PRENOM AGE
---------- ---------- ---
Dupont     Bernard     42
Dupont     Pierre      23
Dupont     Sophie      32
Lambert    Danielle    52
Martin     Romain      41
```

6.7 FLIP

Syntaxe

FLIP
/VARIABLES=var_list
/NEWNAMES=var_name.

Description

Dans la structure habituelle de PSPP, les colonnes représentent les variables et les lignes les observations. Si un fichier est organisé de sorte que les lignes représentent les variables et les colonnes les observations, il faut transposer les lignes et les colonnes avec la commande FLIP.

Toutes les variables transposées sont numériques. FLIP assigne aux variables de type string la valeur *system-missing*.

Les sous-commandes ne sont pas obligatoires. La sous-commande *VARIABLES* sélectionne les variables à transformer en observations. Si cette sous-commande n'est pas spécifiée, toutes les variables sont transposées.

La sous-commande *NEWNAMES* spécifie la variable dont les valeurs vont être utilisées pour les noms des nouvelles variables. Les noms des variables originales deviennent les valeurs d'une nouvelle variable appelée CASE_LBL. Si NEWNAMES n'est pas spécifiée, les nouvelles variables créées par FLIP sont appelées VAR000, VAR001, ..., VAR999, VAR1000, etc.

Exemple1

```
DATA LIST NOTABLE FILE=exemple26
   /X1 (A6) X2 (F7.2) X3 (F7.2) X4 (F7.2) X5 (F7.2).
LIST.
FLIP.
LIST.
```

FLIP entraine la transposition de toutes les variables. La sortie obtenue présente l'organisation initiale des données (première commande LIST) et le résultat de la transposition (deuxième commande LIST) :

```
    X1       X2       X3       X4       X5
------ -------- -------- -------- --------
annee   1999.00  2000.00  2001.00  2002.00
prix      24.25    26.50    30.15    28.45
hausse        .     9.28    13.77    -5.64
```

```
CASE_LBL   VAR000   VAR001   VAR002
-------- -------- -------- --------
X1            .        .        .
X2         1999.00    24.25      .
X3         2000.00    26.50     9.28
X4         2001.00    30.15    13.77
X5         2002.00    28.45    -5.64
```

Les chaînes de caractères deviennent *system-missing* car la sous-commande NEWNAMES n'est pas spécifiée.

Exemple2

```
DATA LIST NOTABLE FILE=exemple26
   /X1 (A6) X2 (F7.2) X3 (F7.2) X4 (F7.2) X5 (F7.2).
LIST.
FLIP
   /VARIABLES=X1 X3 X4 X5
   /NEWNAMES=X1.
LIST.
```

L'exemple est le même que le précedent mais la variable X1 sert à nommer les nouvelles variables et la variable X2 est exclue de la transposition. La sortie est la suivante :

```
    X1        X2        X3        X4        X5
------ -------- -------- -------- --------
annee   1999.00  2000.00  2001.00  2002.00
prix      24.25    26.50    30.15    28.45
hausse       .     9.28    13.77    -5.64

CASE_LBL    ANNEE     PRIX   HAUSSE
-------- -------- -------- --------
X3         2000.00    26.50     9.28
X4         2001.00    30.15    13.77
X5         2002.00    28.45    -5.64
```

6.8 AGGREGATE

Syntaxe

```
AGGREGATE
        OUTFILE={*,'filename'}
        /PRESORTED
        /MISSING=COLUMNWISE
        /BREAK=var_list
        /agg_var['label']...=function(var[,var,...],args)
        /...
```

AGGREGATE permet d'agréger un ensemble d'observations ayant la même valeur pour la (ou les) variable(s) spécifiée(s) dans la sous-commande *BREAK*. Ainsi, les gros fichiers de données sont transformés en fichiers plus compacts, et donc plus facilement analysables.

La sous-commande *OUTFILE* est obligatoire et doit être spécifiée en premier. Elle est utilisée pour préciser le nom du fichier *système* (fichier binaire propre au logiciel) où sont écrites les observations agrégées. Le fichier système peut être spécifié comme un fichier de type *handle* (cf. commande *FILE HANDLE*, section 3.3, page 20). Pour que les données agrégées remplacent les données du fichier en cours de traitement, on utilise le symbole '*'.

Une ou plusieurs variables doivent être spécifiées dans la sous-commande *BREAK*. Ce sont les valeurs de ces variables qui vont déterminer les groupes d'observations à agréger. Toutes les observations ayant la même valeur pour les variables *break* sont regroupées en une seule observation.

Si le fichier de données qui va être agrégé est déjà ordonné en fonction des valeurs des variables *break* (c'est-à-dire que les observations ayant les mêmes valeurs pour ces variables se suivent dans le fichier), *PRESORTED* peut être spécifié pour éviter d'allonger inutilement le temps d'exécution.

Les variables agrégées sont calculées à partir des fonctions d'agrégation spécifiées par l'utilisateur. Il faut spécifier après un slash ('/') la liste des variables agrégées qui sont calculées à partir de la même fonction. Il doit y avoir autant de variables agrégées que de variables servant d'arguments à la fonction d'agrégation. On peut donner un label à une variable en le spécifiant entre apostrophes immédiatement après son nom. Un signe égal ('=') sépare les variables agrégées de la fonction d'agrégation. Les variables agrégées calculées à partir de fonctions différentes sont séparées par un slash ('/').

Par défaut, les valeurs des variables agrégées sont calculées si les variables-arguments ont au moins une valeur non-manquante dans le groupe. On spécifie

MISSING=COLUMNWISE pour que la valeur d'une variable agrégée soit déclarée manquante pour un groupe dès qu'une valeur de ce groupe est manquante pour la variable-argument de la fonction d'agrégation.

Voici la liste des fonctions d'agrégation disponibles :
- `MIN(var_name)` : renvoie la valeur *minimum* de la variable spécifiée.
- `MAX(var_name)` : renvoie la valeur *maximum* de la variable spécifiée.
- `SUM(var_name)` : renvoie la *somme* des valeurs de la variable spécifiée, seulement pour les variables numériques (format par défaut : `F8.2`).
- `MEAN(var_name)` : renvoie la *moyenne* des valeurs de la variable spécifiée, seulement pour les variables numériques (format par défaut : `F8.2`).
- `SD(var_name)` : renvoie l'*écart-type* des valeurs de la variable spécifiée, seulement pour les variables numériques (format par défaut : `F8.2`).
- `N` : pour chaque groupe, renvoie le *nombre d'observations* qui ont été *agrégées*. Si on n'a pas attribué un poids particulier aux observations (cf. commande *WEIGHT*, section 7.4, page 52), le format par défaut est `F7.0`. Dans le cas contraire, le format par défaut est `F8.2`.
- `NU` : pour chaque groupe, renvoie le *nombre d'observations* qui ont été *agrégées*. Contrairement à `N`, cette fonction ne tient pas compte des poids qui ont été attribués aux observations (poids égal à 1 pour chaque observation). Le format par défaut est `F7.0`.
- `FGT(var_name,value)` : renvoie la *part* des valeurs *supérieures* à la constante spécifiée (format par défaut : `F5.3`).
- `FLT(var_name,value)` : renvoie la *part* des valeurs *inférieures* à la constante spécifiée (format par défaut : `F5.3`).
- `FIN(var_name,low,high)` : renvoie la *part* des valeurs à l'*intérieur* de l'intervalle spécifié (format par défaut : `F5.3`).
- `FOUT(var_name,low,high)` : renvoie la *part* des valeurs à l'*extérieur* de l'intervalle spécifié (format par défaut : `F5.3`).
- `PGT(var_name,value)` : renvoie le *pourcentage* de valeurs *supérieures* à la constante spécifiée (format par défaut : `F5.1`).
- `PLT(var_name,value)` : renvoie le *pourcentage* de valeurs *inférieures* à la constante spécifiée (format par défaut : `F5.1`).
- `FIN(var_name,low,high)` : renvoie le *pourcentage* de valeurs à l'*intérieur* de l'intervalle spécifié (format par défaut : `F5.1`).
- `FOUT(var_name,low,high)` : renvoie le *pourcentage* de valeurs à l'*extérieur* de l'intervalle spécifié (format par défaut : `F5.1`).
- `FIRST(var_name)` : renvoie la *première* valeur *non-manquante* pour chaque groupe.
- `LAST(var_name)` : renvoie la *dernière* valeur *non-manquante* pour chaque groupe.
- `N(var_name)` : renvoie le *nombre* de valeurs *non-manquantes* pour chaque groupe. Si on n'a pas attribué un poids particulier aux observations, le format par défaut est `F7.0`. Dans le cas contraire, le format par défaut est `F8.2`.
- `NU(var_name)` : renvoie le *nombre* de valeurs *non-manquantes* pour chaque

groupe. Contrairement à `N(var_name)`, cette fonction ne tient pas compte des poids qui ont été attribués aux observations (poids égal à 1 pour chaque observation). Le format par défaut est `F7.0`.

- `NMISS(var_name)` : renvoie le *nombre* de valeurs *manquantes* pour chaque groupe. Si on n'a pas attribué un poids particulier aux observations, le format par défaut est `F7.0`. Dans le cas contraire, le format par défaut est `F8.2`.
- `NUMISS(var_name)` : renvoie le *nombre* de valeurs *manquantes* pour chaque groupe. Contrairement à `NMISS(var_name)`, cette fonction ne tient pas compte des poids qui ont été attribués aux observations (poids égal à 1 pour chaque observation). Le format par défaut est `F7.0`.

Ces fonctions d'agrégation excluent des calculs les valeurs déclarées *user-missing*. Pour inclure ces dernières dans les calculs, il faut mettre un point ('.') après le nom de la fonction (exemple : `SUM.(var_name)`).

Exemple

```
DATA LIST FILE=donnees90 /DEP 1-2 (A) POP 8-16.
VARIABLE LABELS DEP "numero de departement"
                POP "nombre d'habitants de la commune"
AGGREGATE OUTFILE=*
  /PRESORTED
  /BREAK=DEP
  /NOMBRE "nombre de communes du departement"=N
  /POPULATION "nombre d'habitants du departement"=SUM(POP)
  /PGV "nombre d'habitants de la plus grande ville du departement"=MAX(POP)
  /PPC "nombre d'habitants de la plus petite commune du departement"=MIN(POP).
WRITE OUTFILE=aggreg90
   /DEP 1-2 (A)  POPULATION 4-12 PGV 14-22 PPC 24-32  NOMBRE 44-46.
EXECUTE.
```

Le fichier 'donnees90', qui comporte 36137 lignes, présente les données INSEE du recensement de la population française de 1990. Chaque ligne est composée par quatre informations décrivant une commune : le code INSEE, le nombre d'habitants, la densité et le nom de la commune.

Pour donner un aperçu du fichier 'donnees90', voici quelques lignes (non consécutives) qui en sont extraites :

```
06003        300       6        Andon
06004      70005    2644        Antibes
06005        123       7        Ascros
06006       1496     158        Aspremont
29301        214      47        Trézilidé
2A001       1726     146        Afa
2A004      58949     719        Ajaccio
31555     358688    3032        Toulouse
31557      16669     917        Tournefeuille
31556        305      25        Tourreilles (Les)
```

Pour chaque commune, on retient avec la commande DATA LIST le numéro du département (DEP), donné par les deux premiers caractères du code INSEE, et le nombre d'habitants (POP). Seulement les informations utiles au calcul des nouvelles variables sont lues avec DATA LIST.

On agrège les observations par département et on définit quatre nouvelles variables explicitées par leurs labels : NOMBRE, POPULATION, PGV et PPC. Les nouvelles observations sont écrites dans le fichier 'aggreg90' avec la commande WRITE. Ce nouveau fichier comporte 95 lignes.

Voici un extrait du fichier 'aggreg90' obtenu suite à l'agrégation des données (les lignes se suivent dans le fichier) :

```
29     838687     147956         84          283
2A     118808      58949         25          124
2B     131563      37845          9          236
30     585049     128471         23          353
31     925962     358688          6          588
32     174587      23136          0          463
```

7 La sélection des données pour l'analyse

Cette section décrit les commandes permettant de sélectionner des enregistrements de données pour l'analyse.

7.1 PROCESS IF

Syntaxe

PROCESS IF
expression.

Description

La commande PROCESS IF permet d'exclure de la prochaine analyse des données certaines observations.

Les observations sont sélectionnées en fonction des valeurs de l'expression spécifiée. Cette expression retourne une valeur de type booléen pour chaque observation (cf. LES EXPRESSIONS MATHEMATIQUES, section 11, page 82). Si elle retourne 'vrai', l'observation sera analysée ; dans le cas contraire ('faux' ou 'missing'), l'observation sera exclue de la prochaine analyse.

L'exécution de la commande *TEMPORARY* (cf. section 7.3, page 51), puis de la commande *SELECT IF* (cf. section 7.2, page 50), produit les mêmes effets que ceux de la commande PROCESS IF.

Exemple

```
DATA LIST NOTABLE FILE=exemple28 /SEXE (A5) AGE (F3).
PROCESS IF (SEXE="homme") AND (AGE GT 40).
DESCRIPTIVES AGE.
DESCRIPTIVES AGE.
```

PROCESS IF sélectionne les observations pour lesquelles SEXE a la valeur "homme" et AGE une valeur supérieure à 40. Cette sélection s'applique à la première commande DESCRIPTIVES. En effet, la deuxième commande DESCRIPTIVES retourne les statistiques concernant toutes les observations.

Voici, la sortie obtenue dans le fichier listing :

```
1.1 DESCRIPTIVES.  Valid cases = 2; cases with missing value(s) = 0.
+--------#-+------+-------+-------+-------+
|Variable#N| Mean |Std Dev|Minimum|Maximum|
#========#=#======#=======#=======#=======#
|AGE     #2|49.000|  9.899| 42.000| 56.000|
+--------#-+------+-------+-------+-------+
```

```
2.1 DESCRIPTIVES.  Valid cases = 11; cases with missing value(s) = 0.
+--------#--+------+-------+-------+-------+
|Variable# N| Mean |Std Dev|Minimum|Maximum|
#========#==#======#=======#=======#=======#
|AGE     #11|31.273| 12.993| 14.000| 56.000|
+--------#--+------+-------+-------+-------+
```

7.2 SELECT IF

Syntaxe

SELECT IF
expression.

Description

La commande SELECT IF est similaire à la commande PROCESS IF (cf. section 7.1, page 49) mais à la différence de celle-ci, SELECT IF sélectionne les observations pour les analyses de manière définitive.

L'expression spécifiée est évaluée pour chaque observation. Si la valeur retournée est 'vrai', l'observation sera analysée ; si la valeur est 'faux' ou 'missing', l'observation est exclue définitivement des analyses.

Si la commande SELECT IF est spécifiée après *TEMPORARY* (cf. section 7.3, page 51), la sélection des observations devient temporaire.

Exemple

```
DATA LIST NOTABLE FILE=exemple28 /SEXE (A5) AGE (F3).
SELECT IF NOT (AGE > 25).
DESCR AGE.
FREQ /VAR=AGE.
```

Les observations pour lesquelles la variable AGE prend une valeur inférieure ou égale à 25 sont sélectionnées pour l'analyse des données. Les commandes DESCRIPTIVES et FREQUENCIES analysent les mêmes données.

Voici, la sortie obtenue dans le fichier listing :

```
1.1 DESCRIPTIVES.  Valid cases = 5; cases with missing value(s) = 0.
+--------#-+------+-------+-------+-------+
|Variable#N| Mean |Std Dev|Minimum|Maximum|
#========#=#======#=======#=======#=======#
|AGE     #5|20.000|  5.292| 14.000| 25.000|
+--------#-+------+-------+-------+-------+
```

```
2.1 FREQUENCIES.  AGE:
+-----------+--------+---------+--------+--------+--------+
|           |        |         |        |  Valid |   Cum  |
|Value Label|  Value |Frequency| Percent| Percent| Percent|
#===========#========#=========#========#========#========#
|           |      14|        1|    20.0|    20.0|    20.0|
|           |      15|        1|    20.0|    20.0|    40.0|
|           |      21|        1|    20.0|    20.0|    60.0|
|           |      25|        2|    40.0|    40.0|   100.0|
#===========#========#=========#========#========#========#
|               Total|        5|   100.0|   100.0|        |
+--------------------+---------+--------+--------+--------+

+---------------+------+
|N       Valid  |     5|
|        Missing|     0|
|Mean           |20.000|
|Std Dev        | 5.292|
|Minimum        |14.000|
|Maximum        |25.000|
+---------------+------+
```

7.3 TEMPORARY

Syntaxe

TEMPORARY.

Description

La commande TEMPORARY est utilisée pour que les effets d'une ou plusieurs transformations soient temporaires. Les commandes effectuant des transformations, sur les données ou les variables, spécifiées après TEMPORARY affectent seulement la première analyse qui suit.

TEMPORARY peut s'appliquer aux commandes suivantes :
- COMPUTE, COUNT, RECODE, IF (cf. Transformation des données, section 6, page 35)
- FORMATS, PRINT FORMATS, WRITE FORMATS (cf. commande FORMATS, section 5.9, page 34)
- SELECT IF, SAMPLE, WEIGHT, SPLIT FILE (cf. Sélection des données pour l'analyse, section 7, page 49)
- NUMERIC, STRING, VARIABLE LABELS, VALUE LABELS, MISSING VALUES (cf. Manipulation des variables, section 5, page 26)

Exemple

```
DATA LIST  /X 1-4 (1).
BEGIN DATA.
2.0
6.0
12.0
9.0
END DATA.
COMPUTE X=X/2.
TEMPORARY.
COMPUTE X=X+3.
DESCR X.
DESCR X.
```

La première commande COMPUTE transforme les valeurs `2.0`, `6.0`, `12.0`, `9.0` en `1.0`, `3.0`, `6.0`, `4.5`. La première commande DESCRIPTIVES lit les données `4.0`, `6.0`, `9.0`, `7.5`, alors que la deuxième lit les données `1.0`, `3.0`, `6.0`, `4.5`. On obtient la sortie suivante :

```
1.1 DATA LIST.  Reading 1 record from the command file.
+--------+------+-------+------+
|Variable|Record|Columns|Format|
#========#======#=======#======#
|X       |     1|  1-  4|F4.1  |
+--------+------+-------+------+

2.1 DESCRIPTIVES.  Valid cases = 4; cases with missing value(s) = 0.
+--------#-+-----+-------+-------+-------+
|Variable#N| Mean|Std Dev|Minimum|Maximum|
#========#=#=====#=======#=======#=======#
|X       #4|6.625|  2.136|  4.000|  9.000|
+--------#-+-----+-------+-------+-------+

3.1 DESCRIPTIVES.  Valid cases = 4; cases with missing value(s) = 0.
+--------#-+-----+-------+-------+-------+
|Variable#N| Mean|Std Dev|Minimum|Maximum|
#========#=#=====#=======#=======#=======#
|X       #4|3.625|  2.136|  1.000|  6.000|
+--------#-+-----+-------+-------+-------+
```

7.4 WEIGHT

Syntaxe

```
WEIGHT
        BY var_name.
WEIGHT OFF.
```

Description

La commande WEIGHT permet de modifier le poids des observations dans les analyses en leur assignant de nouvelles fréquences d'apparitions.

La variable numérique spécifiée après le mot-clé *BY* détermine les nouveaux poids qui sont attribués aux observations. Ainsi, chaque valeur de cette variable détermine le nombre d'apparitions de l'observation correspondante.

La commande WEIGHT est valide pour toutes les analyses de données jusqu'à ce que la commande WEIGHT OFF soit mentionnée : toutes les observations reprennent alors le même poids.

Si la commande WEIGHT est précédée par *TEMPORARY*, elle affecte uniquement la prochaine analyse des données.

Exemple

```
DATA LIST NOTABLE/NOMBRE (F1) NOTE (F2).
BEGIN DATA.
213
410
315
416
118
2 5
END DATA.
WEIGHT BY NOMBRE.
FREQ NOTE /STATISTICS=NONE.
WEIGHT OFF.
FREQ NOTE /STATISTICS=NONE.
```

Ce sont les valeurs de la variable NOMBRE, spécifiée dans la commande WEIGHT, qui déterminent les fréquences d'apparitions des valeurs de la variable NOTE. La première commande FREQUENCIES tient compte des modifications apportées par WEIGHT. La deuxième commande FREQUENCIES n'en tient pas compte car elle est précédée par WEIGHT OFF.

Voici la sortie obtenue dans le fichier listing :

```
1.1 FREQUENCIES.  NOTE:
+-----------+--------+---------+--------+--------+--------+
|           |        |         |        |  Valid |   Cum  |
|Value Label|  Value |Frequency| Percent| Percent| Percent|
#===========#========#=========#========#========#========#
|           |       5|        2|    12.5|    12.5|    12.5|
|           |      10|        4|    25.0|    25.0|    37.5|
|           |      13|        2|    12.5|    12.5|    50.0|
|           |      15|        3|    18.8|    18.8|    68.8|
|           |      16|        4|    25.0|    25.0|    93.8|
|           |      18|        1|     6.2|     6.2|   100.0|
#===========#========#=========#========#========#========#
|               Total|       16|   100.0|   100.0|        |
+--------------------+---------+--------+--------+--------+

2.1 FREQUENCIES.  NOTE:
+-----------+--------+---------+--------+--------+--------+
|           |        |         |        |  Valid |   Cum  |
|Value Label|  Value |Frequency| Percent| Percent| Percent|
#===========#========#=========#========#========#========#
|           |       5|        1|    16.7|    16.7|    16.7|
|           |      10|        1|    16.7|    16.7|    33.3|
|           |      13|        1|    16.7|    16.7|    50.0|
|           |      15|        1|    16.7|    16.7|    66.7|
|           |      16|        1|    16.7|    16.7|    83.3|
|           |      18|        1|    16.7|    16.7|   100.0|
#===========#========#=========#========#========#========#
|               Total|        6|   100.0|   100.0|        |
+--------------------+---------+--------+--------+--------+
```

7.5 SAMPLE

Syntaxe

SAMPLE
decimal_value
ou num1 FROM num2.

Description

SAMPLE permet de sélectionner aléatoirement une proportion d'observations pour l'analyse. Cette sélection est valable pour toutes les analyses, sauf si *TEMPORARY* (cf. section 7.3, page 51) précède la commande SAMPLE.

Il y a deux manières de spécifier le nombre d'observations à extraire. La première consiste à préciser la proportion à sélectionner avec un nombre décimal compris entre 0 et 1. Si `N` est le nombre d'observations valides dans le fichier en cours, la commande '`SAMPLE .k.`' indique qu'approximativement `k*N` observations sont sélectionnées.

La deuxième méthode est l'utilisation de la commande '`SAMPLE m FROM M`', qui précise la proportion `m/M` d'observations à sélectionner. Trois situations peuvent se présenter :

- Si `M` est égal au nombre d'observations valides dans le fichier en cours, alors exactement `m` observations sont sélectionnées.
- Si `M` est plus grand que le nombre d'observations valides dans le fichier en cours, alors une proportion équivalente à `m/M` est sélectionnée.
- Si `M` est inférieur au nombre d'observations valides dans le fichier en cours, alors `m` observations sont sélectionnées parmi les `M` premières observations du fichier.

La commande SAMPLE est toujours exécutée avant *N OF CASES* (cf. section 7.6, page 55), même si elle est spécifiée après dans le fichier syntaxe.

Exemple1

```
DATA LIST FILE=exemple32 /X1 TO X5 1-5.
SAMPLE .25.
```

Approximativement un quart des observations du fichier de données 'exemple32' sont sélectionnées de manière aléatoire pour les prochaines analyses.

Exemple2

```
DATA LIST FILE=exemple32 /X1 TO X5 1-5.
SAMPLE 5 FROM 10.
```

Cinq observations sont sélectionnées aléatoirement parmi les dix premières observations du fichier 'exemple32'.

7.6 N OF CASES

Syntaxe

N [OF CASES]
number_cases.

Description

N OF CASES permet de sélectionner les premières observations du fichier en cours pour les prochaines analyses.

Le nombre d'observations à sélectionner est déterminé après la commande N (ou N OF CASES). Si ce nombre est supérieur au nombre d'observations valides dans le fichier en cours, la commande est ignorée.

Une commande N OF CASES ultérieure peut augmenter ou diminuer le nombre d'observations sélectionnées pour l'analyse des données.

Les commandes *SAMPLE*, *PROCESS IF* et *SELECT IF* sont toujours exécutées avant N OF CASES, même si elles sont spécifiées après.

Si N OF CASES est spécifiée après *TEMPORARY* (cf. section 7.3, page 51), la sélection des données est valable seulement pour la prochaine analyse.

Exemple

```
DATA LIST NOTABLE FILE=exemple32 /X1 TO X5 1-5.
N 5.
SAMPLE .5.
FREQ X1 /STATISTICS=NONE.
```

D'abord, la commande SAMPLE sélectionne aléatoirement la moitié des observations du fichier 'exemple32', puis la commande N sélectionne les cinq premières observations obtenues. FREQ donne les résultats suivants :

```
1.1 FREQUENCIES.  X1:
+-----------+--------+---------+--------+--------+--------+
|           |        |         |        |  Valid |   Cum  |
|Value Label|  Value |Frequency| Percent| Percent| Percent|
#===========#========#=========#========#========#========#
|           |       1|        2|    40.0|    40.0|    40.0|
|           |       3|        1|    20.0|    20.0|    60.0|
|           |       4|        1|    20.0|    20.0|    80.0|
|           |       5|        1|    20.0|    20.0|   100.0|
#===========#========#=========#========#========#========#
|               Total|        5|   100.0|   100.0|        |
+--------------------+---------+--------+--------+--------+
```

7.7 SPLIT FILE

Syntaxe

```
SPLIT FILE
        BY var_list.
SPLIT FILE OFF.
```

Description

La commande SPLIT FILE permet d'analyser séparément des groupes d'observations. Il faut spécifier après le mot-clé *BY* la liste des variables dont les

valeurs vont déterminer l'ensemble des groupes (à une valeur correspond un groupe). Les groupes sont constitués uniquement par des observations se suivant dans le fichier de données. Pour qu'un seul groupe soit associé à chaque valeur de la variable spécifiée, il faut au préalable ordonner les observations avec la commande *SORT CASES* (cf. section 6.6, page 42).

SPLIT FILE OFF permet de mettre fin à l'analyse séparée des groupes. Quand SPLIT FILE est utilisée après *TEMPORARY* (cf. section 7.3, page 51), seule la prochaine analyse des données est affectée par la commande.

Exemple

```
DATA LIST LIST NOTABLE FILE=exemple35
   /SEXE (F1) AGE (F2) TAILLE (F4.2).
SORT CASES BY SEXE.
SPLIT FILE BY SEXE.
DESCR TAILLE.
```

Les observations sont d'abord ordonnées avec SORT CASES selon les valeurs de la variable SEXE, puis séparées en groupes avec SPLIT FILE selon ces mêmes valeurs. La commande DESCRIPTIVES analyse séparément les deux groupes créés :

```
Variable Value Label
SEXE         1

1.1 DESCRIPTIVES.  Valid cases = 5; cases with missing value(s) = 0.
+--------#-+-----+-------+-------+-------+
|Variable#N| Mean|Std Dev|Minimum|Maximum|
#========#=#=====#=======#=======#=======#
|TAILLE  #5|1.784|   .068|  1.690|  1.850|
+--------#-+-----+-------+-------+-------+

Variable Value Label
SEXE         2

1.2 DESCRIPTIVES.  Valid cases = 5; cases with missing value(s) = 0.
+--------#-+-----+-------+-------+-------+
|Variable#N| Mean|Std Dev|Minimum|Maximum|
#========#=#=====#=======#=======#=======#
|TAILLE  #5|1.662|   .039|  1.600|  1.700|
+--------#-+-----+-------+-------+-------+
```

8 Les fichiers binaires

8.1 SAVE

Syntaxe

SAVE
OUTFILE='filename'

Description

La commande SAVE permet de sauvegarder dans un fichier binaire les données et le dictionnaire, contenant toutes les informations relatives aux variables. Le fichier binaire créé, appelé fichier *système*, ne peut être lu que par certains logiciels, dont PSPP, SPSS et R.

La sous-commande *OUTFILE* permet de préciser le nom du fichier binaire créé, qui peut être spécifié comme un fichier *handle* (cf. commande *FILE HANDLE*, section 3.3, page 20).

Exemple

```
DATA LIST LIST FILE=exemple35
   /SEXE (F1) AGE (F2) TAILLE (F4.2).
VALUE LABELS /SEXE 1 "homme" 2 "femme".
SAVE OUTFILE=ages.sav.
```

Les données du fichier 'exemple35' et le dictionnaire sont sauvegardés dans le fichier système 'ages.sav' (*.sav* est l'extension conventionnelle pour les fichiers binaires).

8.2 GET

Syntaxe

GET
FILE='filename'
/DROP=var_list
/KEEP=var_list
/RENAME=(old_names=new_names)...

Description

La commande GET permet de lire un fichier *système*, créé par la commande *SAVE* (cf. section 8.1, page 58), qui contient des données et un dictionnaire.

La seule sous-commande obligatoire est *FILE*; elle permet de préciser le nom du fichier système lu, qui peut être spécifié comme un fichier *handle* (cf. commande *FILE HANDLE*, section 3.3, page 20).

Par défaut, toutes les variables du dictionnaire sont lues. Pour spécifier une liste de variables qui ne doivent pas être lues, on utilise la sous-commande *DROP*. Pour spécifier la liste des variables qui doivent être lues, on utilise la sous-commande *KEEP*.

On peut modifier les noms des variables du dictionnaire en utilisant la sous-commande *RENAME*. On spécifie entre parenthèses la liste des noms à modifier, suivie d'un signe égal ('=') et de la liste des nouveaux noms. Plusieurs groupes de noms entre parenthèses peuvent être spécifiés dans une commande RENAME.

Les modifications apportées par KEEP, DROP et RENAME n'affectent pas le fichier système lu par la commande GET.

Exemple

```
GET FILE=ages.sav
  /DROP=TAILLE
  /RENAME=(SEXE=INF1) (AGE=INF2).
```

GET lit le fichier système 'ages.sav' en excluant la variable TAILLE, et en renommant les variables SEXE et AGE.

8.3 EXPORT

Syntaxe

EXPORT
OUTFILE='filename'
/DROP=var_list
/KEEP=var_list
/RENAME=(old_names=new_names)...

La commande EXPORT permet de sauvegarder dans un fichier *portable* les données et le dictionnaire, contenant toutes les informations relatives aux variables. Le fichier portable ainsi créé peut être lu par d'autres logiciels que PSPP.

La seule sous-commande obligatoire est *OUTFILE*; elle permet de préciser le nom du fichier portable créé, qui peut être spécifié comme un fichier *handle* (cf. commande *FILE HANDLE*, section 3.3, page 20).

Par défaut, toutes les variables du dictionnaire sont sauvegardées dans le fichier portable. Pour spécifier une liste de variables qui ne doivent pas être sauvegardées, on utilise la sous-commande *DROP*. Pour spécifier la liste des variables qui doivent être sauvegardées, on utilise la sous-commande *KEEP*.

On peut sauvegarder les variables du dictionnaire sous des noms différents en utilisant la sous-commande *RENAME*. On spécifie entre parenthèses la liste des noms à modifier, suivie d'un signe égal ('=') et de la liste des nouveaux noms. Plusieurs groupes de noms entre parenthèses peuvent être spécifiés dans une commande RENAME.

Les modifications apportées par KEEP, DROP et RENAME concernent uniquement le fichier portable créé par la commande EXPORT.

Exemple

```
DATA LIST LIST FILE=exemple35
   /SEXE (F1) AGE (F2) TAILLE (F4.2).
VALUE LABELS /SEXE 1 "homme" 2 "femme".
EXPORT OUTFILE=tailles.sav
   /DROP=AGE
   /RENAME=(SEXE TAILLE=INF1 INF2).
```

EXPORT crée le fichier portable 'tailles.sav' sans sauvegarder la variable AGE, et en renommant les variables SEXE et TAILLE.

8.4 IMPORT

Syntaxe

IMPORT
FILE='filename'
/DROP=var_list
/KEEP=var_list
/RENAME=(old_names=new_names)...

Description

La commande IMPORT permet de lire un fichier *portable* contenant des données et un dictionnaire.

La seule sous-commande obligatoire est *FILE*; elle permet de préciser le nom du fichier portable lu, qui peut être spécifié comme un fichier *handle* (cf. commande *FILE HANDLE*, section 3.3, page 20).

La syntaxe et l'utilisation des sous-commandes *DROP*, *KEEP* et *RENAME* sont les mêmes que pour la commande *GET* (cf. section 8.2, page 58).

Exemple

```
IMPORT FILE=tailles.sav.
```

IMPORT lit le fichier portable 'tailles.sav'.

8.5 MATCH FILES

Syntaxe

MATCH FILES
/{FILE,TABLE}={*,'filename'}
/DROP=var_list
/KEEP=var_list
/IN=var_name
/BY var_list

Description

La commande MATCH FILES permet la fusion de un ou plusieurs fichiers *systèmes*, créés par la commande *SAVE* (cf. section 8.1, page 58). Il est possible de fusionner un ou plusieurs fichiers systèmes avec le fichier en cours de traitement.

Il y a deux manières de fusionner des fichiers :
- si la sous-commande *BY* n'est pas spécifiée, les observations sont jointes une à une dans l'ordre où elles apparaissent dans les fichiers ;
- si la sous-commande *BY* spécifie une variable, appelée variable *clé*, la jonction se fait en fonction des valeurs prises par cette variable, et en fonction de l'ordre des observations dans le fichier système.

Attention, pour que le fichier obtenu après la fusion soit cohérent, il est nécessaire d'ordonner au préalable les fichiers qui vont être fusionnés en fonction des valeurs de la variable clé (cf. commande *SORT CASES*, section 6.6, page 42).

La sous-commande *FILE* précise le nom d'un fichier système, qui peut être spécifié comme un fichier *handle* (cf. commande *FILE HANDLE*, section 3.3, page 20). Pour fusionner le fichier en cours, on utilise le symbole '∗'. On utilise autant de commandes FILE qu'il y a de fichiers à fusionner.

TABLE peut être utilisée de la même manière que FILE pour spécifier un fichier système. La différence avec FILE est que chaque valeur de la variable clé spécifiée par BY n'apparaît qu'une seule fois dans le nouveau fichier, comme avec le système traditionnel des bases de données.

Chaque sous-commande FILE ou TABLE peut être suivie d'une sous-commande *IN*, qui crée une variable numérique avec le nom spécifié. Celle-ci prend la valeur 1 si l'observation provient du fichier qui précède, 0 sinon.

Si les sous-commandes TABLE ou IN sont utilisées, BY doit être spécifiée.

Enfin, *DROP* et *KEEP* peuvent être utilisées après la spécification des fichiers pour déterminer les variables à exclure ou à garder. La syntaxe et l'utilisation de ces sous-commandes sont les mêmes que pour la commande *GET* (cf. section 8.2, page 58).

Exemple

```
DATA LIST LIST NOTABLE FILE=exemple41
   /NOM (A8) SEXE (F1) AGE (F2).
SORT CASES BY NOM.
SAVE OUTFILE=pers.sav.
LIST.

DATA LIST LIST NOTABLE FILE=exemple42
   /NOM (A8) TAILLE (F4.2).
SORT CASES BY NOM.
SAVE OUTFILE=tail.sav.
LIST.

MATCH FILES
   /FILE=pers.sav
   /TABLE=tail.sav
   /IN=PROV
   /DROP=SEXE
   /BY NOM.
LIST.
```

Les observations des fichiers systèmes 'pers.sav' et 'tail.sav' sont jointes de manière à former un seul fichier. Les jointures sont faites en fonction des valeurs de la variable NOM, rangées au préalable dans l'ordre alphabétique.

Les commandes LIST donnent les sorties suivantes :

```
    NOM SEXE AGE
-------- ---- ---
DUPONT      1  45
DURAND      1  21
GERARD      1  35
MARTIN      2  32
SORDAN      2  43
```

```
     NOM TAILLE
-------- ------
DUPONT     1.81
GERARD     1.85
MARTIN     1.68
SORDAN     1.63
```

```
     NOM AGE TAILLE PROV
-------- --- ------ ----
DUPONT    45   1.81    1
DURAND    21    .      0
GERARD    35   1.85    1
MARTIN    32   1.68    1
SORDAN    43   1.63    1
```

9 Les statistiques élémentaires

Cette section présente les commandes effectuant des analyses statistiques.

9.1 DESCRIPTIVES

Syntaxe

```
DESCRIPTIVES
        [/VARIABLES=]var_list
        /MISSING={VARIABLE,LISTWISE} {INCLUDE}
        /FORMAT={LINE,SERIAL}
        /SAVE
        /STATISTICS={DEFAULT,MEAN,SEMEAN,STDDEV,VARIANCE,
            KURTOSIS,SKEWNESS,RANGE,MINIMUM,MAXIMUM,
            SUM,SESKEWNESS,SEKURTOSIS,ALL}
        /SORT={MEAN,SEMEAN,STDDEV,VARIANCE,KURTOSIS,
            SKEWNESS,RANGE,MINIMUM,MAXIMUM,SUM,
            SESKEWNESS,SEKURTOSIS} {(A),(D)}.
```

Description

DESCRIPTIVES retourne les résultats des statistiques descriptives demandées par l'utilisateur.

La sous-commande *VARIABLES* spécifie la liste des variables numériques qui doivent être analysées (le mot-clé *VARIABLES* est facultatif). Toutes les sous-commandes qui suivent sont optionnelles.

MISSING détermine la manière dont les valeurs manquantes vont être traitées. La sous-commande active par défaut est *VARIABLE*, qui exclut toutes les valeurs manquantes (*system-missing* et *user-missing*) de l'analyse des données. Si *LISTWISE* est spécifié, les observations qui contiennent une valeur manquante (*system-missing* ou *user-missing*) pour une des variables à analyser sont entièrement exclues de l'analyse. *INCLUDE* permet d'inclure les valeurs *user-missing* dans l'analyse des données. Si INCLUDE est spécifié avec LISTWISE, seules les observations contenant une valeur *system-missing* pour une des variables à analyser sont exclues.

FORMAT modifie la sortie des résultats. Si *SERIAL* est spécifié, le nombre de valeurs manquantes et non manquantes est indiqué pour chaque variable. Dans le cas contraire, *LINE* est active par défaut et seul le nombre de valeurs traitées pour chaque variable est indiqué.

La sous-commande *SAVE* calcule le *Z-score* pour chaque variable spécifiée dans la commande VARIABLES. Les Z-scores sont sauvegardés dans de nouvelles variables nommées à partir du nom de la variable d'origine, auquel on

ajoute le préfixe `Z`. On peut choisir le nom du Z-score associé à une variable en le spécifiant entre parenthèses après le nom de cette variable dans la commande VARIABLES.

Remarque : Pour obtenir le Z-score d'une variable, on transforme les valeurs de cette variable, en conservant la même échelle, de manière à obtenir une variable dont les valeurs ont pour moyenne 0, et pour écart-type 1. Le Z-score est aussi appelé *variable centrée réduite*.

La sous-commande *STATISTICS* détermine les statistiques à afficher dans la sortie :

- `MEAN` : moyenne empirique
- `SEMEAN` : erreur standard de la moyenne
- `STDDEV` : écart-type
- `VARIANCE` : variance
- `KURTOSIS` : kurtosis (mesure l'aplatissement d'une distribution) et erreur standard de la kurtosis
- `SKEWNESS` : skewness (mesure la symétrie d'une distribution) et erreur standard de la skewness
- `RANGE` : écart entre le maximum et le minimum
- `MINIMUM` : valeur minimum
- `MAXIMUM` : valeur maximum
- `SUM` : somme des valeurs
- `DEFAULT` : moyenne, écart-type, minimum et maximum
- `SEKURTOSIS` : erreur standard de la kurtosis
- `SESKEWNESS` : erreur standard de la skewness
- `ALL` : toutes les statistiques ci-dessus

Enfin, *SORT* détermine l'ordre de sortie des résultats en fonction des valeurs de la statistique choisie. On spécifie '`(A)`' (par défaut) pour désigner l'ordre ascendant, et '`(D)`' pour l'ordre descendant.

Exemple

```
DATA LIST NOTABLE FILE=exemple43
   /NOM 1-6 (A) SEXE 8 AGE 10-11 TAILLE 13-16 (2).
DESCRIPTIVES
   /VARIABLES=AGE TAILLE /SAVE /FORMAT=SERIAL.
DESCRIPTIVES
   AGE TAILLE /MISSING=LISTWISE /STATISTICS=DEFAULT RANGE.
```

On calcule des statistiques descriptives pour les variables AGE et TAILLE. La première commande DESCRIPTIVES crée les Z-scores associés aux variables et affiche les statistiques par défaut (moyenne, écart-type, minimum et maximum). La deuxième commande DESCRIPTIVES exclut des calculs les observations comprenant une valeur manquante pour les variables AGE ou TAILLE, et affiche les statistiques DEFAULT et RANGE.

Données du fichier 'exemple43' :

```
DUPONT 1 45 1.86
MARTIN 2    1.72
GERARD 1 35
SORDAN 2 43 1.63
DURAND 1 21 1.75
BOYER  1 22
DUBEC    32 1.66
```

On obtient la sortie suivante :

```
1.1 DESCRIPTIVES.  Mapping of variables to corresponding Z-scores.
+------+-------+
|Source| Target|
#======#=======#
|AGE   |ZAGE   |
|TAILLE|ZTAILLE|
+------+-------+

1.2 DESCRIPTIVES.  Valid cases = 7; cases with missing value(s) = 3.
+--------#-------+---------+------+-------+-------+-------+
|Variable#Valid N|Missing N| Mean |Std Dev|Minimum|Maximum|
#========#=======#=========#======#=======#=======#=======#
|AGE     #      6|        1|33.000| 10.139| 21.000| 45.000|
|TAILLE  #      5|        2| 1.724|   .090|  1.630|  1.860|
+--------#-------+---------+------+-------+-------+-------+

2.1 DESCRIPTIVES.  Valid cases = 4; cases with missing value(s) = 3.
+--------#-+------+-------+------+-------+-------+
|Variable#N| Mean |Std Dev| Range|Minimum|Maximum|
#========#=#======#=======#======#=======#=======#
|AGE     #4|35.250| 11.087|24.000| 21.000| 45.000|
|TAILLE  #4| 1.725|   .103|  .230|  1.630|  1.860|
+--------#-+------+-------+------+-------+-------+
```

9.2 FREQUENCIES

Syntaxe

```
FREQUENCIES
        [/VARIABLES=]var_list
        /FORMAT={TABLE,NOTABLE,LIMIT(limit)}
            {STANDARD,CONDENSE,ONEPAGE[(limit)]}
            {SINGLE,DOUBLE}
        /STATISTICS={DEFAULT,MEAN,SEMEAN,MEDIAN,MODE,
            STDDEV,VARIANCE,KURTOSIS,SKEWNESS,RANGE,
            MINIMUM,MAXIMUM,SUM,SESKEWNESS,
            SEKURTOSIS,ALL,NONE}
        /NTILES=ntiles
        /PERCENTILES=percentiles.
```

Description

FREQUENCIES retourne des tableaux de fréquences pour les variables spécifiées par l'utilisateur. FREQUENCIES peut aussi retourner des statistiques comme la médiane, le mode et les centiles (*percentiles*), en plus des statistiques descriptives proposées par la commande *DESCRIPTIVES* (cf. section 9.1, page 64).

Les valeurs *user-missing* sont exclues de l'analyse des données.

La sous-commande *VARIABLES* spécifie la liste des variables qui doivent être analysées (le mot-clé *VARIABLES* est facultatif). Toutes les sous-commandes qui suivent sont optionnelles.

La sous-commande *FORMAT* détermine les options d'affichage. Par défaut, un tableau de fréquences est affiché pour chaque variable spécifiée dans la sous-commande VARIABLES (*TABLE*). Si *NOTABLE* est spécifié, ces tableaux ne s'affichent pas. *LIMIT* suivi d'un nombre entre parenthèses entraine l'affichage des tableaux seulement si ils contiennent un nombre inférieur ou égal de valeurs.

Si les tableaux sont affichés, on peut choisir leur format en mentionnant une des trois options qui suivent :

- `STANDARD` (option par défaut) : pour que les tableaux contiennent les informations complètes ;
- `CONDENSE` : pour que les tableaux contiennent moins d'informations et occupent moins d'espace ;
- `ONEPAGE(n)` : pour que les tableaux soient affichés en format condensé si ils contiennent plus de valeurs que le nombre spécifié (si le nombre n'est pas spécifié, il est par défaut égal à 50).

Enfin, si *DOUBLE* est spécifié, les valeurs dans les tableaux sont espacées par un saut de ligne (ce n'est pas le cas avec *SINGLE*, option par défaut).

Remarque : les labels des variables apparaissent en haut de chaque tableau de fréquences.

Les statistiques qui peuvent être spécifiées dans la sous-commande *STATISTICS* sont les mêmes que pour la commande *DESCRIPTIVES* (cf. section 9.1, page 64), avec trois nouvelles options possibles :

- `MEDIAN` : médiane ;
- `MODE` : mode (s'il y a plusieurs modes, la plus petite valeur est retournée) ;
- `NONE` : aucune statistique.

La sous-commande *PERCENTILES* permet d'afficher les centiles spécifiés par l'utilisateur. Les centiles doivent être spécifiés comme une liste de nombres compris entre 0 et 100, séparés par des virgules ou des espaces.

Remarque : des bugs peuvent apparaître lors de l'affichage des centiles ; aussi, pour calculer ces derniers, il est plutôt conseillé d'utiliser la commande *EXAMINE* (cf. section 9.3, page 70).

NTILES répartit les valeurs en N parties égales et retourne les quantiles correspondant aux coupures. Par exemple, `NTILES=4` retourne les valeurs des quartiles. Un seul nombre peut être précisé après NTILES, aussi, plusieurs sous-commandes NTILES sont autorisées.

Exemple

```
DATA LIST NOTABLE FILE=exemple44 /X1 8 X2 10-11.
VARIABLE LABELS X1 "Sexe" X2 "Age".
VALUE LABELS X1 1 "Homme" 2 "Femme".
FREQUENCIES X1 /STATISTICS=NONE .
FREQ X2 /FORMAT=LIMIT(5)
        /STAT=MODE
        /NTILES=4
        /PERCENTILES=10,90.
```

La première commande FREQUENCIES produit le tableau de fréquences de la variable X1 ; aucune statistique n'est retournée.

La deuxième commande FREQUENCIES analyse la variable X2 et produit la table si elle ne contient pas plus de 5 valeurs. Les statistiques retournées sont le mode, les quartiles, le premier décile et le dernier décile.

On obtient la sortie suivante :

1.1 FREQUENCIES. X1: Sexe

Value Label	Value	Frequency	Percent	Valid Percent	Cum Percent
Homme	1	4	40.0	40.0	40.0
Femme	2	6	60.0	60.0	100.0
Total		10	100.0	100.0	

2.1 FREQUENCIES. X2: Age

N	Valid	9
	Missing	1
Mean		34.222
Mode		35.000
Percentiles	0	21.000
	10	21.800
	25	26.000
	50	35.000
	75	40.000
	90	44.600
	100	51.000

9.3 EXAMINE

Syntaxe

EXAMINE
[/VARIABLES=]var_list
/STATISTICS={NONE,DESCRIPTIVES,EXTREME[(n)]}
/PLOT={NONE,BOXPLOT,NPPLOT,HISTOGRAM}
/CINTERVAL n
/PERCENTILES=[value_list]={NONE,HAVERAGE,WAVERAGE,
ROUND,AEMPIRICAL,EMPIRICAL}
/MISSING={LISTWISE,PAIRWISE} {EXCLUDE,INCLUDE}.

Description

La commande EXAMINE est utilisée pour obtenir des statistiques descriptives et des graphiques.

La sous-commande *VARIABLES* spécifie la liste des variables qui doivent être analysées (le mot-clé *VARIABLES* est facultatif). Toutes les sous-commandes qui suivent sont optionnelles.

La sous-commande *STATISTICS* détermine les statistiques qui vont être affichées (par défaut, aucune statistique n'est affichée). *DESCRIPTIVES* permet d'afficher une table présentant les statistiques descriptives classiques (médiane, moyenne, variance, écart interquartile, etc.).

EXTREME est utilisée pour afficher dans une table les valeurs extrêmes des variables spécifiées dans la sous-commande VARIABLES. *EXTREME(n)* permet d'afficher les n valeurs les plus faibles et les n valeurs les plus élevées. Si aucun nombre n'est spécifié entre parenthèses, les cinq valeurs extrêmes sont affichées.

La sous-commande *PLOT* permet d'obtenir le graphique demandé par l'utilisateur.

Remarque : on ne peut pas visualiser le graphique dans le fichier listing ASCII *pspp.list*. Il faut donc exécuter le fichier syntaxe en tapant la commande `pspp -o html` *nom_fichier* dans le shell UNIX pour obtenir l'ensemble des sorties dans un fichier listing html (*pspp.html*). On peut ensuite visualiser les sorties en ouvrant le fichier *pspp.html* avec un navigateur. Les sorties autres que le graphique peuvent tout de même apparaître dans le fichier *pspp.list* si on exécute le fichier syntaxe avec la commande classique `pspp` *nom_fichier*.

Voici la liste des graphiques proposés :
- `BOXPLOT` : "boîte à moustaches"
- `NPPLOT` : "doite de Henry" (test de normalité)
- `HISTOGRAM` : histogramme

Si la sous-commande PLOT n'est pas spécifiée, aucun graphique n'est affiché.

CINTERVAL spécifie le niveau de l'intervalle de confiance à utiliser pour le calcul des statistiques descriptives de la sous-commande STATISTICS. Par défaut, le niveau de confiance est égal à 95%.

La sous-commande *PERCENTILES* spécifie les centiles qui doivent être affichés et la méthode utilisée pour les calculer. Par défaut, les centiles 5, 10, 25, 50, 75, 90 et 95 sont affichés. Voici la liste des méthodes proposées pour calculer les centiles :

- `HAVERAGE`
- `WAVERAGE`
- `ROUND`
- `AEMPIRICAL`
- `EMPIRICAL`

La méthode utilisée par défaut est HAVERAGE. Si la sous-commande PERCENTILES n'est pas spécifiée, aucun centile n'est affiché.

La sous-commande *MISSING* détermine la manière de traiter les valeurs manquantes. Par défaut, si une observation contient une valeur manquante pour une des variables à analyser, elle est entièrement exclue des analyses (*LISTWISE*). *PAIRWISE* exclut l'observation contenant une valeur manquante seulement pour l'analyse de la variable concernée par la valeur manquante.

Si *INCLUDE* est spécifié, les valeurs *user-missing* sont considérées comme étant non-manquantes, et sont donc incluses dans les analyses. Par défaut, ces dernières sont exclues des analyses (*EXCLUDE*).

Exemple

```
DATA LIST NOTABLE FILE=exemple45 /X2 10-11 X3 13-16 (2).
VARIABLE LABELS  X2 "Age" X3 "Taille".
EXAMINE X2
   /STATISTICS=DESCRIPTIVES EXTREME(2)
   /PLOT=BOXPLOT
   /PERCENT.
```

La commande EXAMINE analyse la variable X2. STATISTICS affiche les statistiques descriptives et les valeurs extrêmes de la variable. PLOT affiche la "boite à moustaches", et PERCENT affiche les centiles calculés par défaut.

La sortie obtenue dans le fichier listing est la suivante :

```
1.1 EXAMINE.  Case Processing Summary
#===#==============================#
#   #             Cases            #
#   #----------+---------+----------#
#   #   Valid  | Missing |   Total  #
#   #--+-------+-+-------+--+-------#
#   # N|Percent|N|Percent| N|Percent#
#===#==#=======#=#=======#==#=======#
#Age#11|   100%|0|     0%|11|   100%#
#===#==#=======#=#=======#==#=======#


1.2 EXAMINE.  Extreme Values
#==========#==========#=====#
#           #Case Number|Value#
#==========#==========#=====#
#AgeHighest1#          9|51.00#
#          2#         11|50.00#
#   --------#----------+-----#
#    Lowest1#          2|21.00#
#          2#          6|22.00#
#==========#==========#=====#


1.3 EXAMINE.  Descriptives
#============================================#========#==========#
#                                            #Statistic|Std. Error#
#============================================#========#==========#
#AgeMean                                     #  33.82  |   3.349  #
#   95% Confidence Interval for MeanLower Bound#  34.034 |          #
#                                  Upper Bound#  33.603 |          #
#   5% Trimmed Mean                          #  33.58  |          #
#   Median                                   #  35.00  |          #
#   Variance                                 # 123.364 |          #
#   Std. Deviation                           #  11.107 |          #
#   Minimum                                  #  21.000 |          #
#   Maximum                                  #  51.000 |          #
#   Range                                    #  30.000 |          #
#   Interquartile Range                      #  21.00  |          #
#   Skewness                                 #   .368  |   .661   #
#   Kurtosis                                 #  -1.321 |   1.279  #
#============================================#========#==========#
```

```
1.4 EXAMINE.  Percentiles
#==================#=========================================#
#                  #                Percentiles              #
#                  #-----+-----+-----+-----+-----+-----+-----#
#                  #  5  |  10 |  25 |  50 |  75 |  90 |  95 #
#===#==============#=====#=====#=====#=====#=====#=====#=====#
#Age|HAverage      #12.60|21.20|22.00|35.00|43.00|50.80|51.00#
#   |Tukey's Hinges#     |     |24.00|35.00|41.50|     |     #
#===#==============#=====#=====#=====#=====#=====#=====#=====#
```

9.4 CROSSTABS

Syntaxe

CROSSTABS
(mode integer)
[/VARIABLES=var_list (low,high)...]

[/TABLES=]var_list BY var_list [BY var_list]...
/MISSING={TABLE,INCLUDE,REPORT}
/CELLS={COUNT,ROW,COLUMN,TOTAL,EXPECTED,
RESIDUAL,SRESIDUAL,ASRESIDUAL,ALL,NONE}
/STATISTICS={NONE,CHISQ,PHI,CC,LAMBDA,UC,BTAU,
CTAU,RISK,GAMMA,D,KAPPA,ETA,CORR,ALL}

Description

La commande CROSSTABS est utilisée pour afficher des tables de contingence (tableaux croisés) représentant la distribution des variables spécifiées par l'utilisateur.

La sous-commande *TABLES* précise les tables qui vont être affichées. Les tables peuvent être de n'importe quelle dimension. On peut spécifier autant de sous-commandes TABLES qu'on le souhaite. Si on spécifie deux listes de variables séparées par BY, chaque variable de la première liste est croisée avec toutes les variables de la deuxième liste.

En *mode integer*, l'utilisateur précise un intervalle de valeurs pour chaque variable apparaissant dans la sous-commande TABLES. Les valeurs a l'extérieur de cet intervalle sont considérées comme manquantes. En mode integer, les valeurs possibles pour les variables sont les entiers compris dans les intervalles spécifiés. Toutes les combinaisons de valeurs possibles sont affichées dans les tables. Si les valeurs des variables ne sont pas des nombres entiers, alors la partie décimale est tronquée.

Pour utiliser le mode integer, on doit spécifier la sous-commande *VARIABLES* avant la sous-commande TABLES. Après chaque nom de variable, on précise l'intervalle de valeurs que l'on va étudier. Cet intervalle est déterminé par deux entiers spécifiés entre parenthèses.

La sous-commande TABLES est obligatoire en mode général comme en mode integer, mais le mot-clé TABLES n'est obligatoire qu'en mode integer.

MISSING détermine la manière dont les valeurs manquantes vont être traitées. Par défaut, toutes les valeurs manquantes sont exclues des analyses (*TABLE*).

On spécifie *INCLUDE* pour inclure les valeurs *user-missing* dans l'analyse des données.

En mode integer seulement, on peut spécifier *REPORT* pour faire apparaître les valeurs *user-missing* dans les tables, mais marquées avec un 'M'. En revanche, ces valeurs sont exclues des calculs statistiques.

CELLS détermine le contenu des cases des tables de contingence. Voici la liste des options possibles :

- `COUNT` (option par défaut) : effectif observé ;
- `ROW` : pourcentage en lignes ;
- `COLUMN` : pourcentage en colonnes ;
- `TOTAL` : pourcentage total ;
- `EXPECTED` : effectif attendu si les deux variables qui composent la table sont statistiquement indépendantes ;
- `RESIDUAL` : différence entre l'effectif observé et l'effectif attendu (résidu) ;
- `SRESIDUAL` : résidu standardisé ;
- `ASRESIDUAL` : résidu standardisé ajusté ;
- `ALL` : toutes les options ci-dessus ;
- `NONE` : supprime la table de contingence.

Si on spécifie le mot-clé CELLS seul, COUNT, ROW, COLUMN et TOTAL sont sélectionnées. Si CELLS n'est pas spécifié, COUNT est sélectionnée par défaut.

STATISTICS détermine les statistiques à afficher :

- `CHISQ` : test du chi-deux de Pearson (test d'indépendance entre deux variables), test du rapport de vraisemblance, test exact de Fisher (tables 2×2), correction de continuité de Yates (tables 2×2)
- `PHI` : Phi
- `CC` : coefficient de contingence
- `LAMBDA` : Lambda
- `UC` : coefficient d'incertitude
- `BTAU` : Tau-b de Kendall
- `CTAU` : Tau-c de Kendall
- `RISK` : risque relatif (tables 2×2)
- `GAMMA` : Gamma
- `D` : D de Somers

- `KAPPA` : coefficient Kappa de Cohen (tables carrées, valeurs des lignes et des colonnes identiques)
- `ETA` : Eta
- `CORR` : r de Pearson et coefficient de corrélation de Spearman
- `ALL` : toutes les statistiques ci-dessus
- `NONE` : aucune statistique

Si on spécifie le mot-clé STATISTICS seul, la statistique CHISQ est affichée. Si STATISTICS n'est pas spécifié, aucune statistique n'est affichée.

Remarque : ces statistiques présentent quelques bugs ; aussi, le recours systématique à ces dernières n'est pas recommandé.

Exemple

Cf. *Exemple d'utilisation du logiciel : L'analyse des données et Les résultats*, sections 12.3 et 12.4, pages 96 et 106.

9.5 T-TEST

Syntaxe

T-TEST

(test 1)
TESTVAL=test_value
/VARIABLES=var_list

(test 2)
GROUPS=var(value1[,value2])
/VARIABLES=var_list

(test 3)
PAIRS=var_list [WITH var_list [(PAIRED)]]

(tous les tests)
/MISSING={ANALYSIS,LISTWISE} {EXCLUDE,INCLUDE}
/CRITERIA=CIN(confidence).

Description

Test de comparaison de la moyenne à une valeur fixée (test 1)

Ce test est utilisé pour comparer la moyenne de une ou plusieurs variables à la valeur spécifiée dans la sous-commande *TESTVAL*. *VARIABLES* permet de spécifier la liste des variables qui vont être testées.

Test de comparaison des moyennes de deux échantillons indépendants : test de Student (test 2)

Le test de Student est utilisé pour savoir si deux groupes d'observations ont la même moyenne pour les variables spécifiées dans la sous-commande *VARIABLES*.

Ce test s'inscrit dans la suite logique du test de Fisher (test d'égalité des variances). Si l'hypothèse d'égalité des variances n'est pas acceptée, le test de Student reste valable pour des échantillons de grande taille.

Les valeurs de la variable spécifiée dans la sous-commande *GROUPS* déterminent les deux groupes d'observations. Il y a trois manières de déterminer les deux groupes :

- Préciser deux valeurs entre parenthèses : si la variable spécifiée dans la sous-commande GROUPS prend la première valeur, l'observation appartient au premier groupe ; si la variable prend la deuxième valeur, l'observation appartient au deuxième groupe.
- Préciser une valeur entre parenthèses : si la variable spécifiée dans la sous-commande GROUPS prend une valeur strictement inférieure à la valeur spécifiée, l'observation appartient au premier groupe ; si la variable

prend une valeur supérieure ou égale, l'observation appartient au deuxième groupe.
- Si aucune valeur n'est spécifiée entre parenthèses, les valeurs par défaut 1.0 et 2.0 déterminent les deux groupes.

Test de comparaison des moyennes de deux échantillons appariés (test 3)

Ce test est utilisé pour comparer les moyennes de deux variables mesurées sur les mêmes "individus". Il y a trois manières différentes de spécifier les variables à tester :
- Spécifier une liste de variables : chaque variable de la liste est comparée avec toutes les autres variables. Exemple : `PAIRS=X1 X2 X3` (X1 comparée avec X2, X1 comparée avec X3 et X2 comparée avec X3).
- Spécifier deux listes de variables séparées par le mot-clé *WITH* : chaque variable de la première liste est comparée à toutes les variables de la deuxième liste. Exemple : `PAIRS=X1 X2 WITH X3` (X1 comparée avec X3 et X2 comparée avec X3).
- Spécifier deux listes de variables séparées par *WITH*, et suivies par *(PAIRED)* : les variables sont comparées deux à deux dans l'ordre spécifié (les deux listes doivent avoir le même nombre de variables). Exemple : `PAIRS=X1 X2 WITH X3 X4 (PAIRED)` (X1 comparée avec X3 et X2 comparée avec X4).

Commandes communes à tous les tests

MISSING détermine la manière de traiter les observations contenant des valeurs manquantes. Par défaut, les observations contenant une valeur manquante (pour une des variables spécifiées dans les sous-commandes VARIABLES, GROUPS ou PAIRS) sont exclues seulement pour les tests analysant la variable contenant la valeur manquante (*ANALYSIS*). Si *LISTWISE* est spécifié, les observations contenant une valeur manquante pour une des variables à analyser sont exclues de toutes les analyses.

Si *INCLUDE* est spécifié, les valeurs *user-missing* sont considérées comme étant non-manquantes, et sont donc incluses dans les analyses. Par défaut, ces dernières sont exclues des analyses (*EXCLUDE*).

La sous-commande *CRITERIA* précise le niveau de l'intervalle de confiance à utiliser pour les tests. Après le mot-clé *CIN*, il faut spécifier entre parenthèses une valeur comprise entre 0 et 1. Par défaut, le niveau de confiance est égal à 95% (0.95).

Exemple

Cf. *Exemple d'utilisation du logiciel : L'analyse des données et Les résultats*, sections 12.3 et 12.4, pages 96 et 106.

9.6 ONEWAY

Syntaxe

ONEWAY
[/VARIABLES=]var_list BY var
/MISSING={ANALYSIS,LISTWISE} {EXCLUDE,INCLUDE}
/CONTRAST=value1[,value2,...,valueN]
/STATISTICS={DESCRIPTIVES,HOMOGENEITY}

Description

La commande ONEWAY est utilisée pour l'analyse de variance à un facteur, c'est-à-dire pour tester la différence entre les moyennes de plusieurs échantillons. Le principe du test de la variance est le suivant :

H_0 *: les moyennes sont toutes égales*

H_1 *: une moyenne au moins est différente des autres*

Pour pouvoir utiliser ce test, il convient de réaliser au préalable un test d'*homogénéité des variances* (tester l'égalité des variances). Si les variances sont hétérogènes, le test de la variance n'est pas utilisable.

La liste des variables à analyser est spécifiée dans la sous-commande *VARIABLES* (le mot-clé VARIABLES n'est pas obligatoire). On spécifie après *BY* le nom de la variable dont les valeurs vont déterminer les différents groupes d'observations.

MISSING détermine la manière de traiter les observations contenant des valeurs manquantes. Par défaut, les observations contenant une valeur manquante (pour une des variables spécifiées dans la sous-commande VARIABLES) sont exclues seulement pour les tests analysant la variable contenant la valeur manquante (*ANALYSIS*). Si *LISTWISE* est spécifié, les observations contenant une valeur manquante pour une des variables à analyser sont exclues de toutes les analyses.

Si *INCLUDE* est spécifié, les valeurs *user-missing* sont considérées comme étant non-manquantes, et sont donc incluses dans les analyses. Par défaut, ces dernières sont exclues des analyses (*EXCLUDE*).

CONTRAST peut être utilisée lorsqu'on anticipe des différences entre certains groupes, que l'on veut donc comparer entre eux. CONTRAST est suivi d'une liste de valeurs correspondant aux coefficients attribués aux groupes (le nombre de coefficients doit être égal au nombre de groupes). La somme des coefficients attribués doit être égale à 0. Pour obtenir plusieurs tests avec contraste, on peut spécifier jusqu'à dix sous-commandes CONTRAST.

La sous-commande *STATISTICS* permet d'afficher deux types de statistiques :

- `DESCRIPTIVES` : affiche des statistiques descriptives sur les différents groupes (nombre d'observations, moyenne, écart-type, erreur standard, intervalle de confiance à 95%, minimum et maximum) ;
- `HOMOGENEITY` : affiche les résultats du test d'homogénéité des variances.

10 Les utilitaires

10.1 COMMENT

Syntaxe

COMMENT texte.
ou *texte.

Description

COMMENT est utilisée pour écrire des commentaires sur le fichier syntaxe. Le nombre de lignes de commentaires n'est pas limité. On met fin aux commentaires avec un point ('.') ou un saut de ligne.

10.2 ECHO

Syntaxe

ECHO "texte".

Description

ECHO permet à l'utilisateur d'afficher un texte avec la sortie standard. Le texte doit être délimité par des apostrophes.

10.3 TITLE

Syntaxe

TITLE "titre".
ou TITLE titre.

Description

TITLE permet de donner un titre à la sortie obtenue suite à l'exécution du fichier syntaxe. Le titre apparaît en haut de chaque page de la sortie. Si le titre n'est pas entouré d'apostrophes, les lettres sont écrites en majuscules.

10.4 SUBTITLE

Syntaxe

SUBTITLE "sous-titre".
ou SUBTITLE sous-titre.

Description

SUBTITLE permet de donner un sous-titre à la sortie obtenue suite à l'exécution du fichier syntaxe. Le sous-titre apparaît en haut de chaque page de la sortie, sous le titre. Si le sous-titre n'est pas entouré d'apostrophes, les lettres sont écrites en majuscules.

10.5 INCLUDE

Syntaxe

INCLUDE 'nom_fichier'.
ou @nom_fichier.

Description

INCLUDE permet d'inclure un fichier de commandes dans le fichier syntaxe courant. Le fichier est alors lu par PSPP comme une partie du fichier syntaxe initial.

10.6 QUIT

Syntaxe

QUIT.
ou EXIT.

Description

QUIT est utilisée pour mettre fin à une session en mode ligne et retourner dans le shell UNIX. Cette commande ne peut pas être utilisée dans un fichier syntaxe.

11 Les expressions mathématiques

Les expressions mathématiques qui apparaissent dans les commandes partagent une syntaxe commune. Des nombres, des chaînes de caractères, des noms de variables ou des opérateurs peuvent composer des expressions mathématiques. Il existe cinq types d'opérateurs : groupant, arithmétiques, logiques, relationnels, et les fonctions. Les opérateurs nécessitent un ou plusieurs paramètres et renvoient un unique résultat.

11.1 Les booléens

Certains opérateurs fonctionnent avec des booléens, c'est-à-dire des variables qui représentent le fait qu'une condition soit satisfaite ou non. Les booléens peuvent prendre trois valeurs : 0 (faux), 1 (vrai), et system-missing (inconnue).

11.2 Les opérateurs groupant

Ce sont les parenthèses : '`( )`'. L'expression qu'elles contiennent est évaluée en priorité. Les parenthèses sont aussi utilisées pour désigner les arguments des fonctions.

11.3 Les opérateurs arithmétiques

Ces opérateurs ont pour paramètres des valeurs ou des variables numériques, et renvoient des résultats numériques.

- `a + b` : retourne la somme de *a* et *b*.
- `a - b` : soustrait *b* à *a* et retourne la différence.
- `a * b` : retourne le produit de *a* et *b*.
- `a / b` : divise *a* par *b* et retourne le quotient. Si *b* vaut zéro, le résultat est *system-missing*.
- `a ** b` : élève *a* à la puissance *b*. Si *a* est négatif et si *b* n'est pas un entier, le résultat est *system-missing*. Le résultat de '`0**0`' est aussi *system-missing*.

11.4 Les opérateurs logiques

Ces opérateurs renvoient 'vrai', 'faux', ou 'missing' selon la valeur des paramètres.

- `a AND b` ou `a & b` : vrai si *a* et *b* sont vrais, faux dans le cas contraire. Si un paramètre est faux, le résultat est faux, même si l'autre paramètre est *missing*. Si les deux paramètres sont *missing*, le résultat est *missing*.
- `a OR b` ou `a | b` : vrai si *a* est vrai ou si *b* est vrai. Si un paramètre est vrai, le résultat est vrai, même si l'autre paramètre est *missing*. Si les deux paramètres sont *missing*, le résultat est *missing*.
- `NOT a` ou `~ a` : vrai si *a* est faux. Si le paramètre est *missing*, le résultat est *missing*.

11.5 Les opérateurs relationnels

Ces opérateurs comparent des valeurs et retournent un résultat de type booléen. On ne peut pas comparer un nombre à une chaîne de caractères.

- `a EQ b` ou `a = b` : vrai si *a* est égal à *b*.
- `a LE b` ou `a <= b` : vrai si *a* est inférieur ou égal à *b*.
- `a LT b` ou `a < b` : vrai si *a* est strictement inférieur à *b*.
- `a GE b` ou `a >= b` : vrai si *a* est supérieur ou égal à *b*.
- `a GT b` ou `a > b` :vrai si *a* est strictement supérieur à *b*.
- `a NE b` ou `a ~= b` ou `a <> b` : vrai si *a* n'est pas égal à *b*.

11.6 Les fonctions

Les fonctions ont une syntaxe commune : un nom de fonction, suivi de un ou plusieurs arguments entre parenthèses. Les noms des fonctions ne sont pas réservés. Une fonction est traitée comme telle seulement si son nom est suivi par une parenthèse ouvrante. Ainsi, '`EXP`' fait référence à la variable EXP.

11.6.1 Les fonctions mathématiques

Ces fonctions prennent des arguments numériques et retournent des résultats numériques.

- `EXP(nombre)` : fonction exponentielle.
- `LG10(nombre)` : fonction logarithme décimal, si le nombre est négatif, le résultat est *system-missing*.
- `LN(nombre)` : fonction logarithme népérien, si le nombre est négatif, le résultat est *system-missing*.
- `SQRT(nombre)` : fonction racine carrée, si le nombre est négatif, le résultat est *system-missing*.
- `ABS(nombre)` : retourne la valeur absolue d'un nombre.
- `MOD(numérateur, dénominateur)` : retourne le reste de la division du numérateur par le dénominateur. Si le dénominateur vaut 0, le résultat est *system-missing*.
- `MOD10(nombre)` : retourne le reste de la division d'un nombre par 10.
- `RND(nombre)` : retourne la partie entière d'un nombre.

11.6.2 Les fonctions trigonométriques

Ces fonctions prennent des arguments numériques et retournent des résultats numériques.

- `COS(nombre)` : fonction cosinus, le nombre est en radians.
- `SIN(nombre)` : fonction sinus, le nombre est en radians.
- `TAN(nombre)` : fonction tangente, le nombre est en radians. Si le nombre est un multiple de `pi/2`, le résultat est *system-missing*.

- `ARCOS(nombre)` ou `ACOS(nombre)` : fonction arccosinus (réciproque de la fonction cosinus), en radians. Si le nombre n'est pas compris entre -1 et 1, le résultat est *system-missing*.
- `ARSIN(nombre)` ou `ASIN(nombre)` : fonction arcsinus (réciproque de la fonction sinus), en radians. Si le nombre n'est pas compris entre -1 et 1, le résultat est *system-missing*.
- `ARTAN(nombre)` ou `ATAN(nombre)` : fonction arctangente (réciproque de la fonction tangente), en radians.

11.6.3 Les fonctions pour les valeurs manquantes

Ces fonctions prennent des arguments numériques et retournent divers types de résultats. Lors de l'évaluation des expressions, les valeurs *user-missing* pour les variables numériques sont transformées en valeurs *system-missing*.

- `MISSING(expression)` : retourne 1 si l'expression a une valeur *missing*, 0 sinon.
- `SYSMIS(expression)` : retourne 1 si l'expression a une valeur *system-missing*, 0 sinon.
- `NMISS(expr[, expr...])` : retourne le nombre de valeurs *missing* dans la liste, les arguments doivent être des expressions numériques.
- `NVALID(expr[, expr...])` : retourne le nombre de valeurs qui ne sont pas *missing* dans la liste, les arguments doivent être des expressions numériques.
- `VALUE(variable)` : empêche les valeurs *user-missing* de la variable d'être transformées en valeurs *system-missing*, et retourne la valeur réelle de la variable.

11.6.4 Les fonctions pour les chaînes de caractères

- `CONCAT(string, string[, ...])` : retourne une chaîne de caractères correspondant aux arguments concaténés. Par exemple, `CONCAT('abc','def')` retourne la valeur `'abcdef'`.
- `LENGTH(string)` : retourne le nombre de caractères de l'argument.
- `NUMBER(string, format)` : retourne le nombre résultant de la transformation de la chaîne de caractères en un nombre à partir du format spécifié. Si la largeur w précisée dans le format est inférieure à la longueur de la chaîne de caractères, seulement les w premiers caractères sont pris en compte. Par exemple, `NUMBER('123', F3.0)` et `NUMBER('1234', F3.0)` renvoient la valeur `123`.
- `STRING(nombre, format)` : retourne la chaîne de caractères correspondant au nombre auquel on applique le format spécifié. Par exemple, `STRING(123.56, F5.1)` renvoie la valeur `'123.6'`.

11.6.5 Les fonctions statistiques

Ces fonctions peuvent retourner des valeurs non manquantes même si plusieurs de leurs arguments sont *missing*. Les fonctions statistiques nécessitent

au moins une valeur non manquante pour retourner un résultat *non-missing*, excepté les fonctions CFVAR, SD et VARIANCE qui en demandent deux.

Il est possible d'augmenter le nombre minimal de valeurs non *missing* avec la syntaxe suivante : *nom_fonction*[.n](*liste_ arg*), où n désigne ce nombre minimal. Par exemple, '`MEAN.3(X, Y, Z)`' retourne une valeur non *missing* seulement si `X`, `Y` et `Z` sont non *missing*.

- `MIN(nombre, nombre[, ...])` : retourne la valeur minimum.
- `MAX(nombre, nombre[, ...])` : retourne la valeur maximum.
- `SUM(nombre, nombre[, ...])` : retourne la somme des valeurs spécifiées.
- `MEAN(nombre, nombre[, ...])` : retourne la moyenne des valeurs spécifiées.
- `VARIANCE(nombre, nombre[, ...])` : retourne la variance des valeurs spécifiées.
- `SD(nombre, nombre[, ...])` : retourne l'écart-type des valeurs spécifiées.
- `CFVAR(nombre, nombre[, ...])` : retourne le coefficient de variation des valeurs spécifiées (écart-type divisé par la moyenne).

11.6.6 Les fonctions pour les distributions statistiques

PSPP peut calculer plusieurs types de fonctions pour les distributions statistiques usuelles. Ces fonctions sont nommées à partir du nom de la fonction et du nom de la distribution statistique. Voici la liste des fonctions proposées :

- PDF.*dist (x[, param...])* : fonction densité de probabilité. Le domaine de définition de x dépend de la distribution. Pour les distributions discrètes, le résultat est la probabilité de x.
- CDF.*dist (x[, param...])* : fonction de répartition (probabilité qu'une variable aléatoire ayant la distribution spécifiée soit inférieure à x). Le domaine de définition de x dépend de la distribution.
- SIG.*dist (x[, param...])* : probabilité qu'une variable aléatoire ayant la distribution spécifiée soit supérieure à x). Le domaine de définition de x dépend de la distribution. Peu de distributions ont une fonction SIG.
- IDF.*dist (p[, param...])* : inverse de la fonction de répartition (donne la valeur x correspondant à la probabilité p).
- RV.*dist ([, param...])* : génère une variable aléatoire de la distribution spécifiée.

Voici les fonctions proposées pour chacune des distributions suivantes :

Lois discrètes

Loi de Bernoulli de paramètre p $(x \in \{0,1\}, 0 \leq p \leq 1)$:
`PDF.BERNOULLI(x, p)`, `CDF.BERNOULLI(x, p)`, `RV.BERNOULLI(p)`.

Loi binomiale de paramètres n et p $(x \in \{0,1,...,n\}, 0 \leq p \leq 1,$
n entier positif) :
`PDF.BINOMIAL(x, n, p)`, `CDF.BINOMIAL(x, n, p)`, `RV.BINOMIAL(n, p)`.

Loi binomiale négative de paramètres n et p $(x \in \{0,1,...,+\infty\},$
$0 \leq p \leq 1$, *n entier positif)* :
`PDF.NEGBIN(x, n, p)`, `CDF.NEGBIN(x, n, p)`, `RV.NEGBIN(n, p)`.

Loi géométrique de paramètre p $(x \in \{0,1,...,+\infty\}, 0 \leq p \leq 1)$:
`PDF.GEOM(x, p)`, `CDF.GEOM(x, p)`, `RV.GEOM(p)`.

Loi hypergéométrique de paramètres N, n et m $(x \in \{0,1,...,n\},$
N, n et m entiers positifs, $n \leq N, m \leq N$*)* :
`PDF.HYPER(x, N, n, m)`, `CDF.HYPER(x, N, n, m)`, `RV.HYPER(N, n, m)`.

Loi de Poisson de paramètre λ $(x \in \{0,1,...,+\infty\}, \lambda \geq 0)$:
`PDF.POISSON(x, λ)`, `CDF.POISSON(x, λ)`, `RV.POISSON(λ)`.

Loi logarithmique de paramètre p $(x \geq 1, 0 \leq p \leq 1)$:
`PDF.LOG(x, p)`, `RV.LOG(p)`.

Lois continues

Loi uniforme sur [a, b] $(x \in [a,b], 0 \leq p \leq 1)$:
`PDF.UNIFORM(x, a, b)`, `CDF.UNIFORM(x, a, b)`, `IDF.UNIFORM(p, a, b)`,
`RV.UNIFORM(a, b)`.

Loi normale de moyenne μ ***et d'écart-type*** σ *(x réel,* $\sigma > 0, 0 < p < 1)$:
`PDF.NORMAL(x, μ, σ)`, `CDF.NORMAL(x, μ, σ)`, `IDF.NORMAL(p, μ, σ)`,
`RV.NORMAL(μ, σ)`.

Loi log-normale de paramètres μ ***et*** σ $(x > 0, \mu > 0, \sigma > 0, 0 \leq p < 1)$:
`PDF.LNORMAL(x, μ, σ)`, `CDF.LNORMAL(x, μ, σ)`, `IDF.LNORMAL(p, μ, σ)`,
`RV.LNORMAL(μ, σ)`.

Loi Gamma de paramètres a et b $(x \geq 0, a > 0, b > 0, 0 \leq p < 1)$:
`PDF.GAMMA(x, a, b)`, `CDF.GAMMA(x, a, b)`, `IDF.GAMMA(p, a, b)`,
`RV.GAMMA(a, b)`.

Loi Beta de paramètres a et b $(0 \leq x \leq 1, a > 0, b > 0, 0 \leq p \leq 1)$:
`PDF.BETA(x, a, b)`, `CDF.BETA(x, a, b)`, `IDF.BETA(p, a, b)`,
`RV.BETA(a, b)`.

Loi exponentielle de paramètre a $(x > 0, a > 0, 0 \leq p < 1)$:
`PDF.EXP(x, a)`, `CDF.EXP(x, a)`, `IDF.EXP(p, a)`,
`RV.EXP(a)`.

Loi du Chi-2 à n degrés de liberté *(*$x \geq 0, 0 \leq p < 1$*, n entier positif)* :
`PDF.CHISQ(x, n)`, `CDF.CHISQ(x, n)`, `SIG.CHISQ(x, n)`,
`IDF.CHISQ(p, n)`, `RV.CHISQ(n)`.

Loi de Student (ou T) à n degrés de liberté *(*x *réel,* $0 < p < 1$*,*
n entier positif) :
`PDF.T(x, n)`, `CDF.T(x, n)`, `IDF.T(p, n)`,
`RV.T(n)`.

Loi de Fisher (ou F) de paramètres n1 et n2 $(x \geq 0, 0 \leq p < 1$*,*
n1 et n2 entiers positifs) :
`PDF.F(x, n1, n2)`, `CDF.F(x, n1, n2)`, `SIG.F(x, n1, n2)`,
`IDF.F(p, n1, n2)`, `RV.F(n1, n2)`.

Loi logistique de paramètres a et b *(*x *réel,* $b > 0, 0 < p < 1$*)* :
`PDF.LOGISTIC(x, a, b)`, `CDF.LOGISTIC(x, a, b)`,
`IDF.LOGISTIC(p, a, b)`, `RV.LOGISTIC(a, b)`.

Loi de Cauchy de paramètres a et b *(*x *réel,* $b > 0, 0 < p < 1$*)* :
`PDF.CAUCHY(x, a, b)`, `CDF.CAUCHY(x, a, b)`,`IDF.CAUCHY(p, a, b)`,
`RV.CAUCHY(a, b)`.

Loi de Laplace de paramètres a et b *(*x *réel,* $b > 0, 0 < p < 1$*)* :
`PDF.LAPLACE(x, a, b)`, `CDF.LAPLACE(x, a, b)`,
`IDF.LAPLACE(p, a, b)`, `RV.LAPLACE(a, b)`.

Loi de Pareto de paramètres a et b $(x \geq a, a > 0, b > 0, 0 \leq p < 1)$:
`PDF.PARETO(x, a, b)`, `CDF.PARETO(x, a, b)`, `IDF.PARETO(p, a, b)`,
`RV.PARETO(a, b)`.

Loi de Rayleigh de paramètre σ $(x \geq 0, \sigma > 0, 0 \leq p < 1)$:
`PDF.RAYLEIGH(x, σ)`, `CDF.RAYLEIGH(x, σ)`, `IDF.RAYLEIGH(p, σ)`,
`RV.RAYLEIGH(σ)`.

Loi de Weibull de paramètres a et b $(x \geq 0, a > 0, b > 0, 0 \leq p < 1)$:
`PDF.WEIBULL(x, a, b)`, `CDF.WEIBULL(x, a, b)`, `IDF.WEIBULL(p, a, b)`,
`RV.WEIBULL(a, b)`.

12 Exemple d'utilisation du logiciel

L'exemple qui est traité dans cette section a pour seul but d'illustrer une démarche que l'on peut adopter pour lire et analyser des données recueillies après une enquête.

12.1 L'enquête

On dispose des résultats d'une enquête sur les relations extra-conjugales menée par le centre hospitalier de Toulouse-Purpan en 2006. 111 personnes ont répondu de manière anonyme au questionnaire proposé pour l'enquête.

Le questionnaire :

NSP : ne se prononce pas

1 - Etes-vous : Un homme ? Une femme ?
2 - Vivez-vous en couple ? oui non
Marié - PACS - Union libre - Sous le même toit - Pas sous le même toit
3 - Quel âge avez-vous ? .....
4 - Etes-vous croyant ?
Pas du tout - Un peu - Assez - Très - NSP
5 - Vous considérez-vous comme quelqu'un de séduisant ?
Pas du tout - Un peu - Assez - Très - NSP
6 - Avez-vous confiance en vous ?
Pas du tout - Un peu - Assez - Beaucoup
7 - Dans une relation, êtes-vous l'initiateur de la relation ?
Jamais - Parfois - Assez souvent - Souvent - Toujours - NSP
8 - Pensez-vous que pour avoir une relation extra-conjugale il est nécessaire de tomber amoureux d'un autre partenaire ?
Pas du tout - Pas toujours - Souvent - Toujours - NSP
9 - Avez-vous eu des relations extra-conjugales ?
Jamais - Rarement - Assez souvent - Souvent - NSP
10 - Si oui, est-ce plutôt des aventures ? Plutôt des relations suivies ?
Plutôt une vie parallèle ? Ça dépend
11 - Si oui, êtes-vous tombé(e) amoureux(se) de votre nouveau partenaire ?
Jamais - Pas toujours - Souvent - Toujours - NSP
12 - Vous sentiriez-vous coupable si vous aviez des relations extra-conjugales ?
Non - Un peu - Assez - Très - NSP
12bis - Si vous avez eu des relations extra-conjugales, vous êtes-vous senti coupable ?
Jamais - Rarement - Souvent - Toujours - NSP
13 - Cette culpabilité vous semble-t-elle un frein à vos (possibles) relations extra-conjugales ?
Jamais - Rarement - Souvent - Toujours - NSP
14 - L'aspect transgression vous semble-t-il stimulant ?

Pas du tout - Un peu - Assez - Très - NSP

15 - Trouvez-vous que les occasions d'avoir des relations extra-conjugales sont fréquentes ?

Pas du tout - Un peu - Assez - Très - NSP

16 - Ces relations vous attirent-elles ?

Beaucoup - Assez - Pas du tout - "Me dégoûtent" - NSP

17 - Sur le lieu de votre travail, avez-vous subi des pressions pour des relations extra-conjugales ?

Jamais - Parfois - Souvent - Très souvent - NSP

18 - Est ce que vous trouvez votre partenaire attirant(e) et disponible sexuellement ?

Pas du tout - Un peu - Plutôt - Vraiment - NSP

19 - Si vous aviez des relations extra-conjugales, pensez-vous que le retentissement sur votre couple serait :

Très négatif - Négatif - Indifférent - Positif - NSP

20 - Comment réagiriez-vous si vous appreniez que votre partenaire vous trompe ?

Plutôt bien - Indifférent - Plutôt mal - Très mal - NSP

21 - Les relations extra-conjugales semblent en augmentation, pensez-vous que c'est du :

- À l'individualisme ambiant ?

Pas du tout - Un peu - Assez - Certainement - NSP

- À la libération des moeurs ?

Pas du tout - Un peu - Assez - Certainement - NSP

- À la contraception féminine ?

Pas du tout - Un peu - Assez - Certainement - NSP

22 - Qu'est-ce qui peut d'après vous justifier une relation extra-conjugale ? .....

23 - Que pensez-vous de l'échangisme ? .....

24 - Pensez-vous qu'une des causes des relations extra-conjugales est le manque de discussions dans le couple ?

Jamais - Parfois - Souvent - Très souvent - NSP

25 - Comment vos proches voient les relations extra-conjugales ?

Très négativement - Négativement - Indifféremment - Positivement - NSP

26 - Comment les amis de votre âge les voient ? .....

27 - Si votre partenaire a eu des relations extra-conjugales :

- Avez-vous eu l'impression d'avoir inconsciemment favorisé cette infidélité ?

Pas du tout - Un peu - Assez - Certainement - NSP

- Avez-vous eu l'impression d'être coupable (manque d'attention, de désir, etc...) ?

Pas du tout - Un peu - Assez - Certainement - NSP

- Avez-vous remis en cause votre comportement sexuel ?

Pas du tout - Un peu - Assez - Certainement - NSP

28 - Le couple traditionnel était à durée illimitée et ne se terminait qu'à la mort. Notre société semble évoluer vers le couple à durée limitée, chacun ayant dans sa vie plusieurs expériences conjugales. Dans ce cadre-là, l'infidélité ne serait plus

une "faute", seule la tromperie serait alors moralement condamnable, qu'en pensez-vous ? .....

Le questionnaire comporte 33 questions dont les réponses ont été reportées après codage dans le fichier 'donnees_REC'.

Le fichier 'donnees_REC' :

```
1 1 1 40 2 2 2 1 2 1 1 1 1 4 3 1 2 2 2 3 4 4 2 1 1 4 2 3 2 2 2 1
2 2 1 42 1 3 2 1 3 2 1 1 1 1 3 2 2 4 4 3 1 4 4 1 2 2 3 3 4 2 2 1
3 1 1 53 3 3 3 1 3 2 3 1 1 1 3 2 2 4 3 3 1 2 1 1 2 3 3 3 3 1 1 1
...
8 2 3 50 1 2 2 2 2 2 3 1 1 3 1 2 1 2 0 4 4 4 2 1 2 2 2 3 3 3 3 1
9 1 3 36 1 3 4 1 2 3 1 1 1 3 3 2 2 4 3 2 1 2 1 0 2 0 2 3 1 1 1 0
10 1 3 30 1 2 2 3 3 3 3 1 0 4 2 1 1 4 4 3 4 4 3 1 4 4 0 3 4 4 4 2
11 1 1 56 3 3 3 2 3 1 2 1 1 2 2 3 1 3 1 3 2 3 2 1 4 2 2 0 2 2 2 0
...
99 2 1 31 2 2 1 2 1 0 0 4 3 2 2 3 1 4 1 4 2 4 1 2 4 4 2 0 0 0 0 0
100 2 5 44 2 3 1 4 1 0 0 4 4 1 3 3 1 3 2 4 3 4 3 1 3 2 0 0 0 0 0 4
101 2 3 23 2 2 2 1 1 0 0 4 4 3 2 3 1 4 1 4 4 4 1 1 4 3 1 0 1 1 1 0
102 2 2 27 4 3 2 0 1 0 0 4 2 1 3 4 1 3 0 3 1 2 1 4 2 2 1 1 1 1 1 3
...
109 2 2 35 2 3 2 1 2 1 1 3 3 1 3 3 2 4 0 2 3 3 1 1 3 2 1 3 4 4 3 0
110 2 2 59 1 2 1 1 1 0 0 4 4 2 3 3 0 4 1 4 2 2 2 4 3 4 2 3 0 0 0 3
111 2 1 79 2 2 2 2 1 0 0 3 0 0 3 3 1 1 0 4 3 2 3 1 4 1 0 2 1 1 1 2
```

Le premier nombre correspond au numéro de la personne interrogée. Les 31 nombres qui suivent traduisent les réponses aux 33 questions (les réponses aux questions 5 et 6 sont regroupées en une seule, idem pour les questions 12 et 12bis). La section "La lecture des données" donne la signification de ces nombres.

12.2 La lecture des données

Avant de procéder à l'analyse des données, une étape préliminaire est la lecture et le recodage des données. On dispose du fichier 'donnees_REC', à partir duquel on va définir l'ensemble des variables analysables. Pour cela, on crée un premier fichier syntaxe (fichier 'lecture') qui a pour but de définir le dictionnaire des données associé au dépouillement du sondage.

Le fichier 'lecture' :

```
TITLE 'Etude medicale fidelite, Purpan 2006'

** lecture libre des données du fichier 'donnees_REC' **
** la variable Q5_6 regroupe les réponses aux questions 5 et 6 **
** la variable Q12 regroupe les réponses aux questions 12 et 12bis **.
```

```
DATA LIST LIST NOTABLE FILE=donnees_REC
 /patient_N Q1 Q2 Q3 Q4 Q5_6 Q7 Q8 Q9 Q10 Q11 Q12 Q13 Q14 Q15 Q16 Q17
  Q18 Q19 Q20 Q21_1 Q21_2 Q21_3 Q22 Q23 Q24 Q25 Q26 Q27_1 Q27_2 Q27_3
  Q28 .

** on donne le format exact des variables **.
FORMATS patient_N (F3.0)
        Q3 (F2.0)
        Q1 Q2 Q4 TO Q28 (F1.0).

** on attribue un label à chaque variable **
** le label décrit le thème de la question posée **.
VARIABLE LABELS
   Q1 "sexe"
   Q2 "situation maritale"
   Q3 "age"
   Q4 "croyance"
   Q5_6 "confiance en soi"
   Q7 "initiative relationnelle"
   Q8 "necessite amour"
   Q9 "relations extra conjugales"
   Q10 "type REC"
   Q11 "amour si REC"
   Q12 "niveau culpabilite REC"
   Q13 "culpabilite frein REC"
   Q14 "transgression stimulante REC"
   Q15 "frequence occasions REC"
   Q16 "attraction REC"
   Q17 "pression REC lieu de travail"
   Q18 "attirance partenaire"
   Q19 "effet REC sur couple"
   Q20 "reaction tromperie partenaire"
   Q21_1 "individualisme -> REC"
   Q21_2 "liberation moeurs -> REC"
   Q21_3 "contraception feminine -> REC"
   Q22 "justification REC"
   Q23 "avis echangisme"
   Q24 "manque discussion -> REC"
   Q25 "vision proches des REC"
   Q26 "vision amis des REC"
   Q27_1 "favorise REC du partenaire"
   Q27_2 "culpabilite REC du partenaire"
   Q27_3 "remise en cause comportement sexuel si REC"
   Q28 "valeur fidelite".
```

```
** on attribue des labels aux valeurs possibles des variables **
** les labels traduisent le codage des réponses proposées **.
VALUE LABELS
  /Q1
      1 "Homme"
      2 "Femme"
  /Q2
      1 "Mariage"
      2 "PACS"
      3 "Union libre"
      4 "meme toit"
      5 "pas meme toit"
  /Q3
      0 "NSP"
  /Q4 Q5_6
      1 "Pas du tout"
      2 "Un peu"
      3 "Assez"
      4 "Tres"
      0 "NSP"
  /Q7
      1 "Jamais"
      2 "Parfois"
      3 "Assez souvent"
      4 "Souvent"
      5 "Toujours"
      0 "NSP"
  /Q8
      1 "Pas du tout"
      2 "Pas souvent"
      3 "Souvent"
      4 "Toujours"
      0 "NSP"
  /Q9
      1 "Jamais"
      2 "Rarement"
      3 "Assez souvent"
      4 "Souvent"
      0 "NSP"
  /Q10
      1 "aventures"
      2 "ca depend"
      3 "relations suivies"
      4 "vie parallèle"
      0 "NSP"
```

```
/Q11
    1 "Jamais"
    2 "Pas souvent"
    3 "Souvent"
    4 "Toujours"
    0 "NSP"
/Q12
    1 "Non"
    2 "Un peu"
    3 "Assez"
    4 "Tres"
    0 "NSP"
/Q13
    1 "Jamais"
    2 "Rarement"
    3 "Souvent"
    4 "Toujours"
    0 "NSP"
/Q14 Q15
    1 "Pas du tout"
    2 "Un peu"
    3 "Assez"
    4 "Tres"
    0 "NSP"
/Q16
    1 "Beaucoup"
    2 "Assez"
    3 "pas du tout"
    4 "Me degoutent"
    0 "NSP"
/Q17 Q24
    1 "Jamais"
    2 "Parfois"
    3 "Souvent"
    4 "Tres souvent"
    0 "NSP"
/Q18
    1 "Pas du tout"
    2 "Un peu"
    3 "Plutot"
    4 "Vraiment"
    0 "NSP"
```

```
  /Q19
      1 "Tres negatif"
      2 "Negatif"
      3 "Indifferent"
      4 "Positif"
      0 "NSP"
  /Q20
      1 "Plutot bien"
      2 "Indifferent"
      3 "Plutot mal"
      4 "Tres mal"
      0 "NSP"
  /Q21_1 to Q21_3
      1 "Pas du tout"
      2 "Un peu"
      3 "Assez"
      4 "Certainement"
      0 "NSP"
  /Q25 Q26
      1 "Tres negativement"
      2 "Negativement"
      3 "Indifferemment"
      4 "Positivement"
      0 "NSP"
   /Q27_1 to Q27_3
      1 "Pas du tout"
      2 "Un peu"
      3 "Assez"
      4 "Certainement"
      0 "NSP"
   /Q22 Q23 Q28
      1 "Pas rigoriste"
      2 "Peu rigoriste"
      3 "Rigoriste"
      4 "Tres rigoriste"
      0 "NSP".
** les réponses aux questions 22, 23 et 28 sont jugées suivant
   le niveau de rigorisme **.

** on déclare les valeurs manquantes **
** la valeur 0 associée aux réponses "NSP" est declarée manquante **.
MISSING VALUES Q1 TO Q28 (0).
```

```
** on recode les variables pour créer de nouvelles variables
   plus facilement analysables **.
RECODE Q9 (2 3 4=1) (1=2) INTO infidelite.
VALUE LABELS infidelite 1 "oui" 2 "non".

RECODE Q8 (3 4=1) (1 2=2) INTO amour_necessaire.
VALUE LABELS amour_necessaire 1 "oui" 2 "non".

RECODE Q5_6 Q12 (3 4=1) (1 2=2) INTO assurance culpabilite.
VALUE LABELS assurance culpabilite 1 "beaucoup ou assez" 2 "peu ou pas".

FORMATS infidelite amour_necessaire assurance culpabilite (F1.0).

** affichage du dictionnaire contenant toutes les informations
   sur les variables **.
DISPLAY DICT.

** sauvegarde du dictionnaire des données dans un fichier binaire **
** par la suite, c'est le fichier 'fidelite.sav' qui sera utilisé pour
   récupérer le dictionnaire, et procéder à l'analyse des données **.
SAVE OUTFILE=fidelite.sav.
```

Ce fichier syntaxe doit être exécuté une fois pour que le fichier 'fidelite.sav' soit créé. Le fichier 'lecture' devra être exécuté à nouveau seulement si on y apporte des modifications.

Après l'exécution de ce fichier syntaxe, le dictionnaire est affiché dans le fichier listing.

À titre d'exemple, voici le début du dictionnaire :

```
Etude medicale fidelite, Purpan 2006      GNU pspp 0.4.0 - sparc-sun-solaris2.9

1.1(1) DISPLAY.
+----------------+-----------------------------------------+--------+
|Variable        |Description                              |Position|
#================#=========================================#========#
|patient_N       |Format: F3.0                             |       1|
+----------------+-----------------------------------------+--------+
|Q1              |sexe                                     |       2|
|                |Format: F1.0                             |        |
|                |Missing Values: 0                        |        |
|                +-----+-----------------------------------+        |
|                |    1|Homme                              |        |
|                |    2|Femme                              |        |
+----------------+-----+-----------------------------------+--------+
```

```
+---------------+-----+---------------------------------+--------+
|Q2             |situation maritale                     |       3|
|               |Format: F1.0                           |        |
|               |Missing Values: 0                      |        |
|               +-----+---------------------------------+        |
|               |    1|Mariage                          |        |
|               |    2|PACS                             |        |
|               |    3|Union libre                      |        |
|               |    4|meme toit                        |        |
|               |    5|pas meme toit                    |        |
+---------------+-----+---------------------------------+--------+
|Q3             |age                                    |       4|
|               |Format: F2.0                           |        |
|               |Missing Values: 0                      |        |
|               +-----+---------------------------------+        |
|               |    0|NSP                              |        |
+---------------+-----+---------------------------------+--------+
|Q4             |croyance                               |       5|
|               |Format: F1.0                           |        |
|               |Missing Values: 0                      |        |
|               +-----+---------------------------------+        |
|               |    0|NSP                              |        |
|               |    1|Pas du tout                      |        |
|               |    2|Un peu                           |        |
|               |    3|Assez                            |        |
|               |    4|Tres                             |        |
+---------------+-----+---------------------------------+--------+
|Q5_6           |confiance en soi                       |       6|
|               |Format: F1.0                           |        |
|               |Missing Values: 0                      |        |
|               +-----+---------------------------------+        |
|               |    0|NSP                              |        |
|               |    1|Pas du tout                      |        |
|               |    2|Un peu                           |        |
|               |    3|Assez                            |        |
|               |    4|Tres                             |        |
+---------------+-----+---------------------------------+--------+
```

12.3 L'analyse des données

On peut désormais procéder à l'analyse des données en faisant simplement appel au fichier binaire 'fidelite.sav', qui comporte toutes les informations sur les variables. Pour cela, on crée un deuxième fichier syntaxe (fichier 'analyse') qui a pour but de regrouper l'ensemble des commandes effectuant des analyses.

Le fichier 'analyse' :

```
TITLE 'Etude medicale fidelite, Purpan 2006'

** lecture du fichier 'fidelite.sav' **.
GET FILE=fidelite.sav.

** on demande des tableaux de fréquences pour les variables Q1, Q2 et Q9 **
** Q1 : sexe ; Q2 : situation maritale ; Q9 : relations extra conjugales **.
FREQ Q1 Q2 Q9
    /STAT=NONE.

** on demande des statistiques sur la répartition des valeurs de
   la variable Q3 **
** Q3 : age **.
EXAMINE Q3
       /STAT=EXTREME
       /PLOT=BOXPLOT
       /PERCENT.

** on demande 5 tableaux croisés : Q1 * infidelite ; Q1 * amour_necessaire ;
   assurance * culpabilite ; assurance * infidelite ; culpabilite * infidelite **
** par défaut, le test du chi-deux est demandé : on teste l'indépendance
   des deux variables de chaque tableau croisé **.
CROSSTABS /TABLES=Q1 BY infidelite amour_necessaire
          /TABLES=assurance BY culpabilite infidelite
          /TABLES=culpabilite BY infidelite
          /CELLS=COUNT ROW COLUMN EXPECTED SRESIDUAL
          /STATISTICS.

** on détermine deux groupes en fonction des valeurs de la variable Q1 **
** premier groupe : hommes ; deuxième groupe : femmes **
** on demande le test de Student pour la variable Q28 : on teste l'égalité
   des moyennes entre le groupe des hommes et celui des femmes **
** Q28 : valeur fidelite **
** pour la variable Q28, la valeur 0 (NSP) a été déclarée manquante **
** ici, on l'inclut dans l'analyse car on teste le niveau de rigorisme
   des hommes et des femmes à travers une question ouverte **.
T-TEST GROUPS=Q1(1,2)
      /VARIABLE=Q28
      /MISSING=INCLUDE.
```

Après l'exécution du fichier syntaxe 'analyse', la sortie suivante est affichée dans le fichier listing :

```
Etude medicale fidelite, Purpan 2006      GNU pspp 0.4.0 - sparc-sun-solaris2.9

1.1 FREQUENCIES.  Q1: sexe
+-----------+--------+---------+--------+--------+--------+
|           |        |         |        |  Valid |   Cum  |
|Value Label|  Value |Frequency| Percent| Percent| Percent|
#===========#========#=========#========#========#========#
|Homme      |       1|       57|    51.4|    51.4|    51.4|
|Femme      |       2|       54|    48.6|    48.6|   100.0|
#===========#========#=========#========#========#========#
|               Total|      111|   100.0|   100.0|        |
+--------------------+---------+--------+--------+--------+

1.2 FREQUENCIES.  Q2: situation maritale
+-------------+--------+---------+--------+--------+--------+
|             |        |         |        |  Valid |   Cum  |
| Value Label |  Value |Frequency| Percent| Percent| Percent|
#=============#========#=========#========#========#========#
|Mariage      |       1|       33|    29.7|    29.7|    29.7|
|PACS         |       2|       12|    10.8|    10.8|    40.5|
|Union libre  |       3|       25|    22.5|    22.5|    63.1|
|meme toit    |       4|        4|     3.6|     3.6|    66.7|
|pas meme toit|       5|       37|    33.3|    33.3|   100.0|
#=============#========#=========#========#========#========#
|                 Total|      111|   100.0|   100.0|        |
+----------------------+---------+--------+--------+--------+

1.3 FREQUENCIES.  Q9: relations extra conjugales
+-------------+--------+---------+--------+--------+--------+
|             |        |         |        |  Valid |   Cum  |
| Value Label |  Value |Frequency| Percent| Percent| Percent|
#=============#========#=========#========#========#========#
|Jamais       |       1|       51|    45.9|    47.2|    47.2|
|Rarement     |       2|       45|    40.5|    41.7|    88.9|
|Assez souvent|       3|        9|     8.1|     8.3|    97.2|
|Souvent      |       4|        3|     2.7|     2.8|   100.0|
|NSP          |       0|        3|     2.7| Missing|        |
#=============#========#=========#========#========#========#
|                 Total|      111|   100.0|   100.0|        |
+----------------------+---------+--------+--------+--------+
```

2.1 EXAMINE. Case Processing Summary

		Cases				
		Valid		Missing		Total
		N	Percent	N	Percent	N	Percent
age	110	99	1%	1	1%	111	100%

2.2(1) EXAMINE. Extreme Values

		Case Number	Value
age	Highest 1	111	79.00
	2	81	72.00
	3	26	70.00
	4	76	63.00
	5	53	62.00
	Lowest 1	72	22.00
	2	101	23.00
	3	65	23.00
	4	40	23.00
	5	88	24.00

2.3 EXAMINE. Percentiles

		Percentiles						
		5	10	25	50	75	90	95
age	HAverage	24.00	26.00	32.75	40.00	48.25	55.80	62.00
	Tukey's Hinges			33.00	40.00	48.00		

3.1 CROSSTABS. Summary.

	Cases					
	Valid		Missing		Total	
	N	Percent	N	Percent	N	Percent
sexe * infidelite	108	97.3%	3	2.7%	111	100.0%
sexe * amour_necessaire	108	97.3%	3	2.7%	111	100.0%
assurance * culpabilite	107	96.4%	4	3.6%	111	100.0%
assurance * infidelite	107	96.4%	4	3.6%	111	100.0%
culpabilite * infidelite	106	95.5%	5	4.5%	111	100.0%

3.2(1) CROSSTABS. Q1 by infidelite [count,expected,row%,column%,std.resid.].

Q1	infidelite		Total
	oui	non	
Homme	37.0	19.0	56.0
	29.6	26.4	.0
	66.1%	33.9%	100.0%
	64.9%	37.3%	51.9%
	1.4	-1.4	.0%
Femme	20.0	32.0	52.0
	27.4	24.6	.0
	38.5%	61.5%	100.0%
	35.1%	62.7%	48.1%
	-1.4	1.5	.0%
Total	57.0	51.0	108.0
	52.8%	47.2%	100.0%
	100.0%	100.0%	100.0%

3.3 CROSSTABS. Chi-square tests.

Statistic	Value	df	Asymp. Sig. (2-sided)
Pearson Chi-Square	8.247	1	.004
Likelihood Ratio	8.351	1	.004
Continuity Correction	7.176	1	.007
Linear-by-Linear Association	8.171	1	.004
N of Valid Cases	108		

3.4(1) CROSSTABS. Q1 by amour_necessaire [count,expected,row%,column%,std.resid.].

	amour_necessaire		
Q1	oui	non	Total
Homme	6.0	49.0	55.0
	13.8	41.2	.0
	10.9%	89.1%	100.0%
	22.2%	60.5%	50.9%
	-2.1	1.2	.0%
Femme	21.0	32.0	53.0
	13.2	39.8	.0
	39.6%	60.4%	100.0%
	77.8%	39.5%	49.1%
	2.1	-1.2	.0%
Total	27.0	81.0	108.0
	25.0%	75.0%	100.0%
	100.0%	100.0%	100.0%

3.5 CROSSTABS. Chi-square tests.

Statistic	Value	df	Asymp. Sig. (2-sided)
Pearson Chi-Square	11.868	1	.001
Likelihood Ratio	12.383	1	.000
Continuity Correction	10.386	1	.001
Linear-by-Linear Association	11.758	1	.001
N of Valid Cases	108		

3.6 CROSSTABS. assurance by culpabilite [count,expected,row%,column%,std.resid.].

assurance	culpabilite: beaucoup ou assez	culpabilite: peu ou pas	Total
beaucoup ou assez	38.0	27.0	65.0
	41.3	23.7	.0
	58.5%	41.5%	100.0%
	55.9%	69.2%	60.7%
	-.5	.7	.0%
peu ou pas	30.0	12.0	42.0
	26.7	15.3	.0
	71.4%	28.6%	100.0%
	44.1%	30.8%	39.3%
	.6	-.8	.0%
Total	68.0	39.0	107.0
	63.6%	36.4%	100.0%
	100.0%	100.0%	100.0%

3.7(2) CROSSTABS. Chi-square tests.

Statistic	Value	df	Asymp. Sig. (2-sided)
Pearson Chi-Square	1.852	1	.174
Likelihood Ratio	1.881	1	.170
Continuity Correction	1.335	1	.248
Linear-by-Linear Association	1.835	1	.176
N of Valid Cases	107		

3.8 CROSSTABS. assurance by infidelite [count,expected,row%,column%,std.resid.].

		infidelite	
		oui	non	Total
assurance	beaucoup ou assez	40.0	26.0	66.0
		35.2	30.8	.0
		60.6%	39.4%	100.0%
		70.2%	52.0%	61.7%
		.8	-.9	.0%
	peu ou pas	17.0	24.0	41.0
		21.8	19.2	.0
		41.5%	58.5%	100.0%
		29.8%	48.0%	38.3%
		-1.0	1.1	.0%
Total		57.0	50.0	107.0
		53.3%	46.7%	100.0%
		100.0%	100.0%	100.0%

3.9(1) CROSSTABS. Chi-square tests.

Statistic	Value	df	Asymp. Sig. (2-sided)
Pearson Chi-Square	3.723	1	.054
Likelihood Ratio	3.735	1	.053
Continuity Correction	2.994	1	.084
Linear-by-Linear Association	3.688	1	.055
N of Valid Cases	107		

3.10 CROSSTABS. culpabilite by infidelite [count,expected,row%,column%,std.resid.].

culpabilite	infidelite oui	infidelite non	Total
beaucoup ou assez	23.0	44.0	67.0
	36.0	31.0	.0
	34.3%	65.7%	100.0%
	40.4%	89.8%	63.2%
	-2.2	2.3	.0%
peu ou pas	34.0	5.0	39.0
	21.0	18.0	.0
	87.2%	12.8%	100.0%
	59.6%	10.2%	36.8%
	2.8	-3.1	.0%
Total	57.0	49.0	106.0
	53.8%	46.2%	100.0%
	100.0%	100.0%	100.0%

3.11 CROSSTABS. Chi-square tests.

Statistic	Value	df	Asymp. Sig. (2-sided)
Pearson Chi-Square	27.700	1	.000
Likelihood Ratio	30.285	1	.000
Continuity Correction	25.615	1	.000
Linear-by-Linear Association	27.439	1	.000
N of Valid Cases	106		

```
4.1(2) T-TEST.  Group Statistics
#=========#==#====#==============#========#
#      Q1 | N|Mean|Std. Deviation|SE. Mean#
#=========#==#====#==============#========#
#Q28 Homme|57|1.16|         1.293|    .171#
#    Femme|54|1.96|         1.414|    .192#
#=========#==#====#==============#========#
```

```
4.2(1) T-TEST.  Independent Samples Test
#==============================#==========#==============================#
#                              # Levene's | t-test for Equality of Means #
#                              #-----+----+------+-------+--------------#
#                              #     |    |      |       |              #
#                              #     |    |      |       |              #
#                              #  F  |Sig.|   t  |   df  |Sig. (2-tailed)#
#==============================#=====#====#======#=======#==============#
#Q28Equal variances assumed    #1.224|.271|-3.134|     **|          .002#
#   Equal variances not assumed#     |    |-3.126|106.798|          .002#
#==============================#=====#====#======#=======#==============#
```

```
4.2(2) T-TEST.  Independent Samples Test
#================================================#
#                                                #
#---------------+--------------------+-----------#
#               |                    |    95%    #
#               |                    +------+----#
#Mean Difference|Std. Error Difference| Lower|Upper#
#===============#====================#======#=====#
#          -.805|                .258|-1.316|-.295#
#          -.805|                .258|-1.316|-.295#
#===============#====================#======#=====#
```

12.4 Les résultats

Résultats FREQUENCIES

Tableau 1.1 : tableau de fréquences de la variable `Q1` ("sexe").

Tableau 1.2 : tableau de fréquences de la variable `Q2` ("situation maritale").

Tableau 1.3 : tableau de fréquences de la variable `Q9` ("relations extra conjugales").

Résultats EXAMINE

Tableau 2.1 : résume l'analyse des observations (nombre d'observations valides et manquantes).

Tableau 2.2 : affiche les cinq valeurs extrêmes de la distribution de la variable `Q3` ("age").

Tableau 2.3 : affiche les centiles 5, 10, 25, 50, 75, 90 et 95 de la distribution de la variable `Q3` ("age"). Les centiles sont calculés avec la méthode `HAVERAGE`.

Résultats CROSSTABS

Tableau 3.1 : résume les tabulations croisées (nombre d'observations valides et manquantes).

Tableau 3.2 : tableau de contingence des variables `Q1` ("sexe") et `infidelite`. On peut lire dans le tableau que 66.1% des hommes et 38.5% des femmes de l'échantillon ont eu une ou plusieurs relations extra-conjugales.

Tableau 3.3 : résultats du test d'indépendance des variables `Q1` ("sexe") et `infidelite`. Le test est significatif (on rejette l'hypothèse d'indépendance) donc les variables `Q1` ("sexe") et `infidelite` ne sont pas indépendantes.

Tableau 3.4 : tableau de contingence des variables `Q1` ("sexe") et `amour_necessaire`. On peut lire dans le tableau que 77.8% des personnes qui pensent que l'amour est nécessaire dans une relation extra-conjugale sont des femmes.

Tableau 3.5 : résultats du test d'indépendance des variables `Q1` ("sexe") et `amour_necessaire`. Le test est significatif (on rejette l'hypothèse d'indépendance) donc les variables `Q1` ("sexe") et `amour_necessaire` ne sont pas indépendantes.

Tableau 3.6 : tableau de contingence des variables `assurance` et `culpabilite`. On peut lire dans le tableau que 71.4% des personnes qui ont peu ou pas d'assurance se sentiraient très ou assez coupable s'ils avaient une relation extra-conjugale.

Tableau 3.7 : résultats du test d'indépendance des variables `assurance` et `culpabilite`. Le test n'est pas significatif (on accepte l'hypothèse d'indépendance) donc les variables `assurance` et `culpabilite` sont indépendantes.

Tableau 3.8 : tableau de contingence des variables `assurance` et `infidelite`. Parmi les personnes qui sont infidèles, 70.2% déclarent avoir beaucoup ou pas mal d'assurance.

Tableau 3.9 : résultats du test d'indépendance des variables `assurance` et `infidelite`. Le test n'est pas significatif (on accepte l'hypothèse d'indépendance) donc les variables `assurance` et `infidelite` sont indépendantes.

Tableau 3.10 : tableau de contingence des variables `culpabilite` et `infidelite`. Parmi les personnes qui n'ont jamais eu de relation extra-conjugale, 89.8% se sentiraient très ou assez coupable s'ils en avaient.

Tableau 3.11 : résultats du test d'indépendance des variables `culpabilite` et `infidelite`. Le test est significatif (on rejette l'hypothèse d'indépendance) donc les variables `culpabilite` et `infidelite` ne sont pas indépendantes.

Résultats T-TEST

Tableau 4.1 : affiche les statistiques de la variable `Q28` ("valeur fidelite") pour chaque groupe défini (nombre d'observations valides, moyenne, écart-type, erreur standard). La variable `Q28` peut prendre les valeurs 1, 2, 3, 4 suivant le niveau de rigorisme de la réponse donnée. Ainsi, une réponse très rigoriste prend la valeur 4 et une réponse pas rigoriste la valeur 1. La valeur 0 associée à l'absence de réponse est incluse dans l'analyse car elle révèle un manque d'intérêt pour la question, et donc un manque d'attachement aux valeurs morales. Les hommes ont une moyenne égale à 1.16 alors que les femmes ont une moyenne égale à 1.96.

Tableau 4.2 : résultats du test d'égalité des moyennes pour la variable `Q28` ("valeur fidelite"). On accepte l'hypothèse d'égalité des variances (test de Fisher) donc les deux groupes sont comparables. L'hypothèse d'égalité des moyennes est rejetée (test de Student), donc les moyennes des deux groupes sont significativement différentes. On peut dire que dans cet échantillon, les femmes sont plus attachées aux valeurs morales que les hommes (aussi, elles s'expriment plus que les hommes sur les questions ouvertes).

13 Annexes

13.1 GNU Free Documentation License

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject.

(Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.

C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

D. Preserve all the copyright notices of the Document.

E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

H. Include an unaltered copy of this License.

I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone

of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.

N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.

O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties–for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements."

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See 'http://www.gnu.org/copyleft/'.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

13.2 Importer des données de PSPP dans *R*

La version 0.4.0 de PSPP ne permet pas d'effectuer des calculs statistiques avancés. Pour traiter des données de manière plus sophistiquée, on peut utiliser le logiciel libre *R*. Par exemple, on ne peut pas faire de régression linéaire avec PSPP ; mais il est possible d'importer rapidement des données de PSPP dans *R* pour effectuer une régression linéaire.

La première étape est de sauvegarder les données manipulées avec PSPP dans un fichier binaire, en tapant dans le fichier syntaxe de PSPP la commande suivante (cf. commande *SAVE*, section 8.1, page 58) :

```
save outfile=exemple.sav
```

Pour importer des données avec *R*, il faut utiliser le module *foreign* (éventuellement à installer). Après avoir lancé le logiciel *R*, il faut taper les commandes suivantes pour successivement importer les données du fichier 'exemple.sav' et effectuer la régression linéaire sur les variables X et Y déclarées dans PSPP :

```
> library(foreign)
> reglin <- read.spss("exemple.sav")
> attach(reglin)
> lm(Y ~ X)
```

Références

[1] Ben Pfaff, *GNU PSPP, A System for Statistical Analysis, Edition 0.4.0, for PSPP version 0.4.0* (http ://www.gnu.org/software/pspp/manual/)

[2] SPSS Inc., *SPSS 6.1 Syntax Reference Guide*

[3] Duncan Cramer, *Introducing statistics for social research, step-by-step calculations and computer techniques using SPSS*

[4] http ://fr.wikipedia.org/wiki/Logiciel_libre/

[5] http ://www.gnu.org/philosophy/

[6] http ://www.gnu.org/software/pspp/

[7] http ://www.bibmath.net/dico/