eBook Gratuit

APPRENEZ sas

eBook gratuit non affilié créé à partir des **contributeurs de Stack Overflow.**

#sas

Table des matières

À propos

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: sas

It is an unofficial and free sas ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official sas.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

Chapitre 1: Démarrer avec sas

Examples

Installation ou configuration

SAS peut être exécuté dans un modèle client-serveur, en utilisant soit le client lourd Enterprise Guide, soit le client léger SAS Studio (Web-enabled), soit en mode "serveur local" où un système SAS entièrement fonctionnel est présent sur un ordinateur local (Windows ou un bureau ou serveur Unix / Linux s'exécutant en mode interactif) et s'exécutent soit en mode Display Manager (le client local), soit via l'un des clients client-serveur ci-dessus (connecté au serveur installé localement).

L'installation SAS est généralement effectuée par un administrateur SAS, qui installera le logiciel à partir d'un dépôt de logiciel personnalisé pour le site (et souvent fourni directement par SAS Institute).

Pour les besoins de l'apprentissage de SAS, il existe également la version gratuite de SAS University Edition, qui peut être installée gratuitement à des fins éducatives sur tout ordinateur Windows, Mac ou Unix / Linux. Il est disponible directement auprès de SAS, actuellement sur la page SAS University Edition , en exécutant une instance AWS (sur le niveau gratuit) ou en téléchargeant une machine virtuelle localement. Consultez le guide d'installation sur SAS.com pour les instructions à jour ou ci-dessous pour les instructions actuelles (juillet 2016).

Pour l'installer localement, vous devez d'abord télécharger et installer Oracle Virtualbox 5.0 (Windows / Mac / Linux) . Ensuite, téléchargez l' image disque la plus récente de SAS University Edition , qui fait environ 2 Go et nécessite la création d'un profil SAS.com.

Une fois cela fait, vous devez configurer la machine virtuelle dans VirtualBox. Importez la machine virtuelle SAS en tant qu'appliance ("Import Appliance" dans VirtualBox). Créez un dossier que SAS utilisera comme stockage local (pour que vous puissiez placer les fichiers dans un emplacement où SAS peut les voir) et définissez-le en tant que dossier partagé dans la boîte de dialogue des paramètres de l'ordinateur. Configurez-le pour le montage automatique.

Ensuite, démarrez la machine virtuelle SAS et une fois démarrée, vous pouvez vous connecter via votre navigateur Web, en vous connectant à http: // localhost: 10080 / si vous avez utilisé les paramètres par défaut.

Si vous rencontrez des problèmes, les forums de la communauté SAS - Analytics U sont les forums des fournisseurs pour obtenir une assistance, ou posez une question sur le débordement de la pile .

Vue d'ensemble de la base SAS

SAS est un système intégré de solutions logicielles qui vous permet d'effectuer les tâches suivantes:

- saisie, extraction et gestion des données
- rédaction de rapports et conception graphique
- analyse statistique et mathématique
- prévision des activités et aide à la décision
- recherche opérationnelle et gestion de projet
- développement d'applications

Comment vous utilisez SAS dépend de ce que vous voulez accomplir. Certaines personnes utilisent de nombreuses fonctionnalités du système SAS et d'autres n'en utilisent que quelques-unes.

Bonjour le monde!

En raison de la structure de SAS, il existe trois manières principales de créer "Hello World!" exemples:

1. Dans une étape de données pour placer un message dans le journal SAS (`_null_` indique qu'aucun jeu de données en sortie ne doit être créé):

```
data _null_;
    put "Hell" "o World!";
run;
```

2. Dans une étape de données pour stocker "Hello World!" dans une variable (`foo` indique qu'un ensemble de données de sortie appelé `foo` doit être créé que a) ne contient qu'un seul enregistrement et b) ne contient qu'une seule variable: `bar` , qui a une valeur de `Hello World!` tout le `Hello World!`):

```
data foo ;
    bar="Hello" ;
    put bar= "World!";
run ;
```

3. Via la langue SAS Macro (en "code ouvert" en dehors des étapes de données). `&` identifie un appel à une variable macro et `.` identifie la fin de la variable (si un espace blanc n'est pas souhaité):

```
%let foo=Hello;
%put &foo.o World!;
```

4. Hybride: utilisation d'une variable macro dans une étape de données:

```
%let foo=Hello;

data _null_ ;
  put "&foo World!";
run ;
```

Architecture du serveur SAS

Présentation : Il existe généralement deux types de déploiement SAS:

1. Installation de SAS Foundation uniquement (BASE SAS). Ceci est généralement installé sur un PC. Il n'exécute aucun logiciel serveur.

2. Déploiement planifié SAS pour leur architecture de serveur qui installera l'environnement du serveur SAS avec éventuellement tous les logiciels clients SAS.

Lequel de ceux que vous possédez sera indiqué dans votre courriel SAS Software Order en indiquant la planification ou la non-planification. Si vous effectuez une installation planifiée, vous aurez besoin d'un fichier de plan pour votre commande, d'abord votre topologie.

Note d'installation 44320: Utilisation de plans de déploiement lors d'une installation SAS®

Architecture du serveur SAS

L'environnement du serveur SAS est divisé en trois niveaux différents:

1. Serveur (s) de métadonnées SAS - Le serveur de métadonnées SAS est chargé de gérer l'environnement du serveur SAS, y compris les bibliothèques, les utilisateurs et la configuration du serveur.

2. Serveur (s) d' application SAS - Le serveur d'application SAS est principalement un serveur de calcul sur lequel vos clients lancent généralement des travaux.

3. SAS Middle Tier (s) = Le niveau intermédiaire de SAS est principalement votre niveau Web qui exécute vos applications Web.

4. Niveau client - Le niveau client correspond aux applications client de vos utilisateurs qu'elles utilisent pour se connecter à l'environnement, telles que SAS Enterprise Guide.

Papier 363-2011 | Comprendre l'anatomie d'un déploiement SAS®: Qu'y a-t-il dans la soupe de serveur? Mark Schneider, Donna Bennett et Connie Robison, SAS Institute Inc., Cary, Caroline du Nord

Topologie:

Les niveaux de métadonnées SAS, SAS Application Server et SAS Middle Tier peuvent être installés sur un seul serveur ou répartis sur plusieurs serveurs. Ceci est déterminé par le fichier de plan que vous avez, il doit correspondre à la topologie souhaitée pour votre déploiement.

Généralement, la plupart des clients, sinon tous, sont des applications Windows, de sorte que le niveau client se trouve sur les postes de travail de vos utilisateurs SAS. En option, ils pourraient probablement être installés sur le ou les serveurs s'ils sont basés sur Windows.

Systèmes d'exploitation pris en charge par SAS

Versioning

Les principales versions actuelles de SAS sont les versions 9.4 et 9.3. Ce sont les versions du

moteur SAS de base le plus couramment utilisé aujourd'hui. Le lien vers les notes de version pour les versions 9.1 + et d'autres documents connexes est inclus ci-dessous.

Veuillez également noter qu'il existe divers packages et fonctions qui étendent les fonctionnalités de SAS et que ceux-ci ont leur propre documentation et fonctionnalité .

- SAS 9.4 - Documentation clé et notes de mise à jour
- SAS 9.3 - Documentation clé et notes de mise à jour
- SAS 9.2 - Documentation clé et notes de mise à jour
- SAS 9.1.x - Documentation clé et notes de mise à jour

Lire Démarrer avec sas en ligne: https://riptutorial.com/fr/sas/topic/2108/demarrer-avec-sas

Chapitre 2: Copier un fichier, octet pour octet

Introduction

Si vous utilisez SAS pour générer des rapports de quelque sorte, vous allez devoir copier un fichier à un moment donné. J'ai principalement utilisé cette méthode pour copier un modèle Excel, puis pour transférer des données via PROC EXPORT dans le nouveau fichier que j'ai créé.

C'est un excellent exemple que j'ai trouvé de Chris Hemedinger (http://blogs.sas.com/content/sasdummy/2011/06/17/how-to-use-sas-data-step-to-copy-a) -file-from-everywhere /) .

Examples

Copier un fichier, octet par octet

```
/* these IN and OUT filerefs can point to anything */
filename in "anyfilehere.xlsx";
filename out "anyfilehere.xlsx";


/* copy the file byte-for-byte  */
data _null_;
  length filein 8 fileid 8;
  filein = fopen('in','I',1,'B');
  fileid = fopen('out','O',1,'B');
  rec = '20'x;
  do while(fread(filein)=0);
     rc = fget(filein,rec,1);
     rc = fput(fileid, rec);

     rc =fwrite(fileid);
  end;
  rc = fclose(filein);
  rc = fclose(fileid);
run;

filename in clear;
filename out clear;
```

Lire Copier un fichier, octet pour octet en ligne: https://riptutorial.com/fr/sas/topic/9394/copier-un-fichier--octet-pour-octet

Chapitre 3: Création de variables de macro

Introduction

L'utilisation de variables macro dans tous vos programmes SAS est une fonctionnalité de base que tous les programmeurs SAS doivent connaître. L'utilisation de variables macro peut vous aider à garder votre code simple et générique. Le code générique est un code réutilisable.

Examples

Utiliser% LET

Je décrirais% LET comme le moyen le plus simple de créer une variable macro dans SAS.

```
%LET variableName = variableValue;
```

Désormais, où que vous utilisiez `&variableName` , cela se résoudra à `variableValue` .

> REMARQUE: vous pouvez considérer que `variableValue` seul peut vous amener des erreurs de syntaxe, en fonction de la valeur et de son utilisation. Par exemple, s'il s'agit d'une date et que vous l'utilisez dans le cadre d'une instruction PROC SQL, vous devrez l'écrire sous la forme `"&variableName"d` pour fonctionner correctement.

Utilisation de PROC SQL

Utiliser PROC SQL est un bon moyen d'obtenir des résultats rapides à partir d'une table et de les lancer dans des variables. J'ai l'habitude de constater que lorsque je veux compter le nombre d'enregistrements que je viens de charger dans une table, je peux obtenir ce nombre dans une variable avec un appel rapide PROC SQL.

```
PROC SQL;
SELECT
    COUNT(*) INTO:aVariable
FROM
    MyTable

;QUIT;
```

Dans l'exemple ci-dessus, `aVariable` représentera le nombre d'enregistrements existant dans `MyTable` .

Vous pouvez également utiliser PROC SQL pour créer plusieurs variables de macro.

```
PROC SQL;
SELECT
    a,
    b,
```

```
    c INTO:aVariable, :bVariable, :cVariable
FROM
    MyTable


;QUIT;
```

Dans l'exemple ci-dessus, les variables créées dans l'instruction INTO correspondent aux colonnes extraites dans l'ordre dans lequel elles sont renvoyées par l'instruction SELECT. Cependant, seule la première ligne de résultats sera utilisée pour remplir ces 3 variables.

Si vous souhaitez stocker plusieurs lignes et que vous utilisez la version 6.11 ou supérieure, utilisez l'exemple suivant:

```
PROC SQL;
    SELECT DISTINCT
        a,
        b,
        c INTO :aVariable1 - :aVariable5,
              :bVariable1 - :bVariable5,
              :cVariable1 - :cVariable5
    FROM
        MyTable
;QUIT;
```

Les mots-clés `THROUGH` et `THRU` peuvent-ils être utilisés en lieu et place du tiret `-`

Utilisation de Call Symput () dans une étape DATA

```
DATA _null_;
            CALL SYMPUT('testVariable','testValueText');
;RUN;
```

Dans l'exemple ci-dessus, `%PUT &testVariable;` va résoudre à `testvalueText` .

Vous pouvez trouver le besoin de formater votre variable dans l'appel SYMPUT ().

```
DATA _null_;
            CALL SYMPUT('testDate',COMPRESS(PUT(today(),date9.)));
;RUN;
```

Dans l'exemple ci-dessus, `%PUT &testDate;` va résoudre à `10MAR2017`

Lire Création de variables de macro en ligne: https://riptutorial.com/fr/sas/topic/9403/creation-de-variables-de-macro

Chapitre 4: DO Loop

Examples

DO Loop

```
DATA salary;
    /*define variables*/
    raise=0.1;
    salary=50000;
    year=1;
    /*do loop*/
    DO year=1 to 20 by 2;
        salary + salary*raise;
        output; /*generates an observation for each iteration of the do loop, optional*/
    END;
RUN;
```

Macro do loop

```
%macro doloop;
  %do age=11 %to 15 %by 2;
    title Age=&age.;
    proc print data=sashelp.class(where=(age=&age.));
    run;
  %end;
%mend;
%doloop;
```

Lire DO Loop en ligne: https://riptutorial.com/fr/sas/topic/7919/do-loop

Chapitre 5: Envoi d'un email avec SAS

Introduction

Il existe plusieurs raisons pour lesquelles vous pourriez avoir besoin de fonctionnalités de messagerie dans SAS. Vous pourriez envoyer un courrier électronique pour informer une personne qu'un processus a réussi / échoué, vous pourriez envoyer un courrier électronique contenant des variables de macro qui indiquent combien d'enregistrements ont été chargés à la fin de votre flux de données, ou peut-être contenir des rapports. Quel que soit votre besoin, il existe plusieurs manières d'envoyer des e-mails et des fichiers dans SAS.

Paramètres

Tag / Attribut	Valeur
LRECL	Ce paramètre permet de définir la longueur de l'enregistrement lors de la lecture et de l'écriture de fichiers. J'ai résolu de nombreux problèmes en réglant simplement cette valeur à la valeur maximale, à savoir 32767. Il est très possible que définir une valeur similaire à sa valeur maximale soit moins efficace, mais au bout du compte, le travail est fait sans toute perte de performance ressentie. (la gamme pour LRECL est 1-32767)

Examples

Envoi d'un email texte de base avec SAS

```
Filename myEmail EMAIL
    Subject = "My Email Subject"
    From    = "myFromAddress@email.com"
    To      = 'toAddress@email.com'
    CC      = 'ccAddress@email.com'
    Type    = 'Text/Plain';


Data _null_; File myEmail;
    PUT "Email content";
    PUT "&recordsCount loaded to your favorite table today!";
RUN;
```

Joindre un fichier Excel à votre messagerie SAS

```
Filename myEmail EMAIL
    Subject = "My Email Subject"
    From    = "myFromAddress@email.com"
    To      = 'toAddress@email.com'
```

```
    CC      = 'ccAddress@email.com'
    Type    = 'Text/Plain'
    ATTACH = ("my/excel/file/path/file.extension" content_type="application/vnd.ms-excel"
LRECL= 32767);


Data _null_; File myEmail;
    PUT "Email contentent";
    PUT "&recordsCount loaded to your favorite table today!";
RUN;
```

Envoi d'un email SAS avec un corps HTML

Prenez note du type d'e-mail: Type = 'text / html';

```
Filename myEmail EMAIL
    Subject = "My Email Subject"
    From    = "myFromAddress@email.com"
    To      = 'toAddress@email.com'
    CC      = 'ccAddress@email.com'
    Type    = 'text/html';

Data _null_; File myEmail;
PUT "
<html>
    <head>
        <style>
            table, th, td {
                border: 1px solid black;
                   border-collapse: collapse;
            }
        </style>
    </head>
    <body>
        <p>Here is your email</p>
        <p>Go ahead, organize your data within an HTML table tag here!</p>
        <table>
            <tr>
                <th>
                    column 1
                </th>
                <th>
                    column 2
                </th>
            </tr>
            <tr>
                <td>
                    &countOfRecords1
                </td>
                <td>
                    &countOfRecords2
                </td>
            </tr>
        </table>
    </body>
</html>
";
RUN;
```

Il est très possible qu'après avoir créé un message HTML dans SAS, vous constatez que le code HTML est déformé lorsque vous recevez le courrier électronique. Ceci est le résultat de la mise en pause de SAS à la ligne suivante du texte de votre PUT. Une pause a probablement été placée au milieu d'un texte de votre tag. *Si cela vous arrive, essayez de déplacer vos balises HTML. Ce n'est peut-être pas joli, mais vous devrez peut-être avoir des balises pour partager une ligne pour éviter que cela se produise.* Cela m'est arrivé, et c'est exactement comme ça que j'ai résolu ce problème.

Lire Envoi d'un email avec SAS en ligne: https://riptutorial.com/fr/sas/topic/9398/envoi-d-un-email-avec-sas

Chapitre 6: étape de données

Examples

obtenir des données avec le setp de données

```
data newclass(keep=first_name sex weight yearborn);
  set sashelp.class(drop=height rename=(name=first_name));
  yearborn=year(date())-age;
  if yearborn >2002;
run;
```

Data spécifie le jeu de données cible. L'option Conserver spécifie les colonnes à imprimer sur la cible.

Définit le jeu de données source. Drop spécifie les colonnes à ne pas prendre. Renommer renomme le nom en prénom.

Yearborn est une variable numérique implicite calculée (colonne).

Filtrer et données de sortie implicites avec `if` pour les élèves nés après 2002.

Lire étape de données en ligne: https://riptutorial.com/fr/sas/topic/10673/etape-de-donnees

Chapitre 7: Étiquettes SAS

Remarques

Les étiquettes peuvent être utilisées pour décrire une variable qui aide à améliorer la lisibilité de vos sorties. Les étiquettes peuvent être créées de manière permanente dans l'étape `DATA` ou créées temporairement dans une étape `PROC` .

Examples

Créer des étiquettes de variables permanentes dans l'étape DATA

```
data table;
    set table;
    label variable1 = 'label1'
            variable2 = 'label2'
            variable3 = 'label3';
run;
```

Lire Étiquettes SAS en ligne: https://riptutorial.com/fr/sas/topic/7877/etiquettes-sas

Chapitre 8: Formats SAS

Introduction

Informats et formats sont utilisés pour indiquer à SAS comment lire et écrire les données respectivement. Les informats sont couramment utilisés dans un datastep lors de la lecture de données à partir d'un fichier externe. Les informats sont rarement utilisés dans les PROC. Les formats sont couramment utilisés dans les étapes de données et les processus.

Remarques

Les formats SAS convertissent les valeurs numériques ou de caractères en valeurs de caractère. Un format peut être appliqué à l'aide d'un `format` ou d'une instruction `put` , ce qui modifie l'affichage d'une valeur ou utilise la fonction `put` pour stocker la valeur mise en forme dans une nouvelle variable.

Il existe quatre catégories de formats:

- Caractère: indique à SAS d'écrire les valeurs de données de caractère à partir des variables de caractère.
- Date et heure - indique à SAS d'écrire des valeurs de données à partir de variables représentant des dates, des heures et des dates.
- ISO 8601 - demande à SAS d'écrire les valeurs de date, d'heure et de date / heure en utilisant la norme ISO 8601.
- Numérique - indique à SAS d'écrire des valeurs de données numériques à partir de variables numériques.

Les formats prennent généralement la forme `<formatname><w>.<d>;` , `w` étant la largeur (y compris les décimales et le point), `d` étant le nombre de décimales.

Formats de date courants (appliqués aux valeurs de date SAS):

- `date9.` par exemple 02AUG2016
- `ddmmyyn8.` par exemple 02082016
- `ddmmyy8.` par exemple 02/08/16
- `yymmdd10.` par exemple 20160802
- `year4.` par exemple 2016

Formats numériques communs (appliqués aux nombres):

- `comma11.0` eg `comma11.0`
- `comma12.2` eg `comma12.2`
- `dollar11.2` par exemple 5 789,12 $
- `nlmnlgbp11.2`

par exemple £ 2,468.02

Autres formats:

- `$hex8.` , convertir une chaîne en hexadécimal
- `$upcase.` , convertir une chaîne en majuscule
- `$quote.` , mettez une chaîne entre guillemets

Une liste complète des formats peut être trouvée ici>
https://support.sas.com/documentation/cdl/en/lrdict/64316/HTML/default/viewer.htm#a001263753.htm

Examples

Utiliser la déclaration de format

L'instruction de `format` applique le format donné à la variable spécifiée *à des fins d'affichage uniquement* , c'est-à-dire que la valeur sous-jacente ne change pas.

```
data example1 ;
  Date  = '02AUG2016'd ; /* stored as a SAS date, i.e. a number */
  Date2 = '31AUG2016'd ;
  format Date monyy7. Date2 yymmddn8. ;
run ;
```

Rendez-vous amoureux	Date2
AUG2016	20160831

Utilisation de l'instruction de format pour grouper des données

Vous pouvez appliquer des formats dans une procédure, par exemple pour modifier les regroupements dans un `proc summary` `proc freq` ou une `proc summary` `proc freq` .

Grouper les dates SAS

```
data example2 ;
  do Date = '01JUN2016'dt to '31AUG2016'dt ;
    Days = 1 ;
    output ;
  end ;
run ;

/* Summarise by year & month */
proc summary data=example2 nway ;
  class Date ;
  var Days ;
  output out=example2_sum (drop=_TYPE_ _FREQ_) sum= ;
  format Date yymmn6. ; /* e.g. 201606 */
run ;
```

Rendez-vous amoureux	Journées
201606	30
201607	31
201608	31

```
/* Summarise by month & year */
proc summary data=example2 nway ;
  class Date ;
  var Days ;
  output out=example2_sum2 (drop=_TYPE_ _FREQ_) sum= ;
  format Date monyy7. ; /* e.g. JUN2016 */
run ;
```

Rendez-vous amoureux	Journées
JUN2016	30
JUL2016	31
AUG2016	31

L'avantage d'utiliser un format est que l'ordre de tri naturel est conservé.

En utilisant `sashelp.class` comme exemple, dites que vous vouliez comparer la fréquence de la première lettre de chaque nom. Vous pouvez utiliser la fonction `substr()` pour trouver la première lettre et exécuter un programme `proc freq` sur la nouvelle variable. Alternativement, vous pouvez appliquer le `$1.` format à la variable `Name` :

```
proc freq data=sashelp.class ;
  table Name ;
  format Name $1. ;
run ;
```

prénom	COMPTER
UNE	7
B	4
C	2
etc.	

Formats personnalisés

Les formats personnalisés, également appelés formats définis par l'utilisateur, peuvent être créés

et utilisés comme tous les autres formats par défaut.

```
/*Create new character format for state variables*/
PROC FORMAT;
VALUE $statef      'CA' = 'California'
                   'MA' = 'Massachusetts'
                   'NY' = 'New York';

/*Once created, you can use your custom format in PROC and DATA steps*/
PROC PRINT DATA=table;
FORMAT state-var $statef.;
RUN;
```

La variable `state-var` sera imprimée selon le nouveau format. Par exemple, la valeur `'CA'` sera imprimée en tant que `'California'` . Si une valeur n'a pas été formatée, telle que `'CT'` , cette valeur sera imprimée telle qu'elle apparaît dans le jeu de données.

Utiliser des informats pour lire des données

Les informats sont utilisés pour indiquer à SAS comment lire les données et sont identifiés par une déclaration `informat` .

```
data test;
 infile test.csv;
 informat     id $6.
              date mmddyy10.
              cost comma10.2
 ;
 input @1 id
       @7 date
       @20 cost
 ;
run;
```

Informats et Formats peuvent également être utilisés ensemble pour lire les données et les écrire dans un format différent, par exemple avec la variable de salaire ci-dessous:

```
DATA workers;
  informat first last $16.;
  informat salary 12.1;
  informat birthdate 8.;
  input
    first $
    last $
    birthdate
    salary;
  format salary dollar10.;
datalines;
John Smith 19810505 54998.5
Jane Doe 19950925 45884.5
Frank James 19600222 70000.5
Jamie Love 19630530 292000.5
;
run;
```

Lire Formats SAS en ligne: https://riptutorial.com/fr/sas/topic/5010/formats-sas

Chapitre 9: Informats dans SAS

Introduction

SAS `informats` indique à SAS comment lire les données de tout emplacement d'entrée (tel qu'un fichier, une feuille de calcul Excel, un canal nommé ou même une autre variable SAS, etc.) dans une variable.

SAS ne comporte que deux types de données - caractère et numérique - et chaque informat est spécifique au stockage de la valeur dans une variable numérique ou caractère. Si la variable de destination est un caractère, alors l'informat commencera par un symbole `$` , tout le reste sera un informat numérique.

Remarques

Les informats sont très importants, en particulier lorsque nous importons des données d'autres ensembles de données. Par exemple, la plupart du temps, lorsque vous travaillez sur des données en temps réel, nous extrayons des données de diverses sources de données (Oracle, Mysql, Teradata, etc.). Chaque fois que nous importons des données, nous devons spécifier la déclaration informat afin que SAS puisse lire les données correctement.

Examples

Importation de données Excel dans SAS

Par exemple, disons ci-dessous les exemples de données dans un «test» Excel,

```
Purchase_Date    Customer_Name    Price
05-05-2017    Adam    1075
06-05-2017    Noah    1093
07-05-2017    Peter    1072
08-05-2017    Louis    1101
09-05-2017    Zoe    1248
10-05-2017    Kevin    1045
11-05-2017    Messiah    1072
12-05-2017    John    1046
13-05-2017    Stephen    1043
14-05-2017    Solly    1113
15-05-2017    Jeevan    1137
```

Vous devez utiliser le code ci-dessous pour importer ceci avec succès,

```
Data Test;
Infile 'D:\Test.csv';
Delimiter=',' Missover DSD Getnames=Yes;
Informat Purchase_Date date9.;
Informat Price dollarx10.2;
Format Purchase_Date date9.;
```

```
Format Price dollarx10.2;
run;
```

```
Informat in the above code helps SAS to read the data from Excel.
Format in the above code helps to write the data properly into SAS Data set.
```

Importer le caractère vs numérique

L'exemple ci-dessous utilise l'instruction input pour lire une valeur provenant d'une source (dans ce cas, la chaîne `123`) dans une destination de caractère et une destination numérique.

```
data test;
   source = '123';
   numeric_destination = input(source, best.);
   character_destination = input(source, $3.);
run;
```

Lire Informats dans SAS en ligne: https://riptutorial.com/fr/sas/topic/9888/informats-dans-sas

Chapitre 10: Lecture des données

Introduction

La lecture des données dans un jeu de données SAS peut être effectuée à l'aide de plusieurs approches, notamment l'instruction `datalines` , à partir d'un fichier externe utilisant une instruction `infile` dans l'étape de données ou la lecture des données d'un fichier externe à l'aide de `proc import` . De plus, vous pouvez lire des données provenant de sources externes compatibles avec odbc (par exemple, des bases de données SQL) en utilisant les pilotes odbc.

Examples

Lire un fichier texte avec un séparateur de virgule

```
DATA table-name;
    INFILE "file-path/file-name.csv" dsd;
    INPUT Name $ City $ Age;
RUN;
```

Lire les données du fichier Excel

```
PROC IMPORT DATAFILE = "file-path/file-name.xlsx" OUT=data_set DBMS=XLSX REPLACE;
```

PROC IMPORT pour Excel, importation d'une feuille spécifique

Il y aura des moments où vous voulez seulement importer une feuille spécifique à partir d'un fichier Excel avec plusieurs feuilles. Pour ce faire, nous utiliserons " **SHEET =** ".

```
PROC IMPORT
    OUT= YourNewTable
    DATAFILE= "myfolder/excelfilename.xlsx"
    DBMS=xlsx
    REPLACE;
    SHEET="Sheet1";
    GETNAMES=YES;
RUN;
```

> Notez également la possibilité de spécifier si la ligne supérieure importée contient ou non des noms de colonne (**GETNAMES = YES** (ou NO)).

Lire Lecture des données en ligne: https://riptutorial.com/fr/sas/topic/7989/lecture-des-donnees

Chapitre 11: Longueur variable

Syntaxe

- LONGUEUR variable (s) <$> longueur;

Paramètres

Paramètre	Détails
variable (s)	variable (s) que vous souhaitez attribuer à une longueur
$	paramètre facultatif qui spécifie si votre variable est une variable de caractère
longueur	entier qui spécifie la longueur de la variable

Examples

Affectation de la longueur à une variable de caractère

```
data table;
set table;
length state_full $8;
if state = 'KS' then state_full = 'Kansas';
else if state = 'CO' then state_full = 'Colorado';
else state_full = 'Other';
run;
```

Lire Longueur variable en ligne: https://riptutorial.com/fr/sas/topic/7883/longueur-variable

Chapitre 12: Proc SQL

Examples

Créer un ensemble de données vide basé sur un jeu de données existant

Méthode 1:

```
 proc sql;
    create table foo like sashelp.class;
    quit;
```

Méthode 2:

```
proc sql;
create table bar as
    select * from sashelp.class (obs=0);
quit;
```

La méthode 1 devrait être l'option préférée

SELECT Syntaxe

```
PROC SQL options;
  SELECT column(s)
FROM table-name | view-name
   WHERE expression
   GROUP BY column(s)
   HAVING expression
ORDER BYcolumn(s);
QUIT;
```

Exemple 1:

```
proc sql;
select name
       ,sex
    from sashelp.class ;
quit;
```

L'instruction SELECT est spécifiée dans cet ordre:

```
1.select;
2.from;
3.where;
4.group by;
5.having;
6.order by.
```

"select" et "from" sont requis. Les autres clauses sont facultatives.

Lire Proc SQL en ligne: https://riptutorial.com/fr/sas/topic/5870/proc-sql

Chapitre 13: Résoudre les variables de macro entre guillemets dans les traversées PROC SQL

Introduction

L'un des défis auxquels j'ai été confronté lorsque j'ai commencé à utiliser SAS consistait non seulement à transmettre des données de variables de macro dans un transfert SQL PROC, mais à les résoudre correctement si des citations étaient nécessaires. Lors de la transmission d'une chaîne comme value ou date / datetime dans un pass SQL direct, il est probable que des guillemets simples soient présents lors de la résolution.

J'ai trouvé les meilleurs résultats en utilisant la fonction% BQUOTE pour y parvenir.

Remarques

Plus d'informations sur la fonction% BQUOTE peuvent être trouvées ici: https://v8doc.sas.com/sashtml/macro/z4bquote.htm

Examples

Pass-through avec macro variable qui est une date

Tout d'abord, je placerai ma date dans une variable macro.

> NOTE: Je trouve cette date9. fonctionne parfaitement avec IBM® Netezza® SQL et Transact-SQL. Utilisez le format qui fonctionne pour le type de SQL que vous exécutez.

```
data _null_;
            call symput('testDate',COMPRESS(put(today(),date9.)));
;RUN;
%PUT &testDate;
```

Mon énoncé% PUT est résolu comme suit: 10MAR2017

Ensuite, je veux exécuter un passage SQL PROC et résoudre cette variable de macro pour spécifier une date.

```
PROC SQL;
CONNECT TO odbc AS alias (dsn=myServer user=userName password= pass);
CREATE TABLE TableName AS
SELECT *
FROM connection to alias
```

```
    (
        SELECT *
        FROM
            Database.schema.MyTable
        WHERE
            DateColumn = %bquote('&testDate')
    );
QUIT;
```

% bquote ('& testDate') se résoudra à '10MAR2017' lorsque le code sera exécuté.

Lire Résoudre les variables de macro entre guillemets dans les traversées PROC SQL en ligne: https://riptutorial.com/fr/sas/topic/9396/resoudre-les-variables-de-macro-entre-guillemets-dans-les-traversees-proc-sql

Chapitre 14: Utilisation de jointures dans SAS

Introduction

Chaque base de données est une collection de différentes tables et chaque table contient des données différentes de manière organisée. Lorsque vous travaillez avec des données, la plupart des informations dont nous avons besoin sont dispersées dans plus d'une table. Nous avons besoin de jointures / fusions pour obtenir le résultat souhaité.

Dans SAS, nous utilisons des jointures lorsque nous travaillons avec `Proc SQL` et utilisons la fusion tout en travaillant avec l' `Data step` . Nous ne parlerons plus que des jointures dans `Proc SQL` .

Paramètres

Type de jointure	Sortie
Proc Sql	Procédure SQL dans SAS
Créer une table	Crée un jeu de données SAS
Sélectionner	Sélectionne les variables requises des ensembles de données respectifs
Où	Spécifie une condition particulière
Quitter	Mettre fin à la procédure

Remarques

Comme mentionné dans l'introduction, nous pouvons également utiliser `Merge` dans une `data step` qui sera discutée dans un sujet distinct. Les jointures jouent un rôle très important pour mélanger et unifier les données en fonction des besoins.

Examples

Jointure verticale

La jointure verticale ajoute le jeu de données B au jeu de données A, à condition que les deux aient des variables similaires. Par exemple, nous avons des ventes pour le mois de janvier 17 dans le jeu de données A et les ventes pour le mois de février 17 dans le jeu de données B. Pour créer un jeu de données avec des ventes de janvier et de février, nous utilisons Vertical Join.

```
PROC SQL;
CREATE TABLE C AS
SELECT *
```

```
FROM A
UNION
SELECT *
FROM B;
QUIT;
```

Maintenant, le jeu de données C contient des observations à la fois de A et de B et est ajouté verticalement.

Jointure interne

La jointure interne crée un ensemble de données qui contient des enregistrements dont les valeurs correspondent à celles des deux tables. Par exemple, nous avons un jeu de données A contenant des informations sur les clients et un jeu de données B contenant les détails de la carte de crédit. Pour obtenir les détails de carte de crédit des clients dans le jeu de données A, créons un jeu de données C

```
PROC SQL;
CREATE TABLE C AS
SELECT A.*, B.CC_NUM
FROM CUSTOMER A, CC_DETAILS B
WHERE A.CUSTOMERID=B.CUSTOMERID
QUIT;
```

Le jeu de données C n'aura que des observations correspondantes provenant des deux jeux de données.

Joint gauche

La jointure à gauche renvoie toutes les observations dans le jeu de données de gauche, indépendamment de leurs valeurs de clé, mais uniquement des observations avec des valeurs de clé correspondantes provenant du jeu de données correct. Considérant le même exemple que ci-dessus,

```
PROC SQL;
CREATE TABLE C AS
SELECT A.*, B.CC_NUMBER, B.START_DATE
FROM CUSTOMER A LEFT JOIN CC_DETAILS B
ON A.CUSTOMERID=B.CUSTOMERID
QUIT;
```

Le jeu de données C contient toutes les valeurs de la table de gauche, ainsi que les valeurs correspondantes de la table de droite ou les valeurs manquantes en cas de non-correspondance.

Droit rejoindre

Comme la jointure gauche, la jointure droite sélectionne toutes les observations du jeu de données droit et les enregistrements correspondants de la table gauche.

```
PROC SQL;
```

```
CREATE TABLE C AS
SELECT A.*, B.CC_NUMBER, B.START_DATE
FROM CUSTOMER A RIGHT JOIN CC_DETAILS B
ON A.CUSTOMERID=B.CUSTOMERID
QUIT;
```

Le jeu de données C contient toutes les valeurs de la table de droite, ainsi que les valeurs correspondantes de la table de gauche ou les valeurs manquantes en cas de non-correspondance.

Full Join

La jointure complète sélectionne toutes les observations des deux ensembles de données, mais il manque des valeurs pour lesquelles la valeur clé de chaque observation se trouve dans une seule table.

```
PROC SQL;
CREATE TABLE C AS
SELECT A.*, B.CC_NUMBER, B.START_DATE
FROM CUSTOMER A FULL JOIN CC_DETAILS B
ON A.CUSTOMERID=B.CUSTOMERID
QUIT;
```

Le jeu de données C contiendra tous les enregistrements des deux tables et les remplira . pour les matchs manquants de chaque côté.

Lire Utilisation de jointures dans SAS en ligne: https://riptutorial.com/fr/sas/topic/9900/utilisation-de-jointures-dans-sas

Crédits

S. No	Chapitres	Contributeurs
1	Démarrer avec sas	Bendy, brusso, Chris J, Community, dcudonk, fl0r3k, Jay Stevens, Joe
2	Copier un fichier, octet pour octet	Joshua Schlichting
3	Création de variables de macro	Joshua Schlichting
4	DO Loop	heyydrien, zuluk
5	Envoi d'un email avec SAS	Joshua Schlichting
6	étape de données	zuluk
7	Étiquettes SAS	heyydrien
8	Formats SAS	Chris J, GForce, heyydrien, Robert Penridge
9	Informats dans SAS	Praneeth Rachumallu, Robert Penridge
10	Lecture des données	GForce, heyydrien, Joshua Schlichting
11	Longueur variable	heyydrien
12	Proc SQL	Altons, D. O., Jay Stevens
13	Résoudre les variables de macro entre guillemets dans les traversées PROC SQL	Joshua Schlichting
14	Utilisation de jointures dans SAS	Praneeth Rachumallu