LIBNAME et couloir d'instructions SQL: Clés de voûte de la connexion aux BD

Club des Utilisateurs SAS de Québec

Octobre 2023

Jean Hardy

Complexe Jules Dallaire Tour 1 2828, boulevard Laurier Bureau 700 Québec, Qc G1V 0B9
Téléphone: (418) 626-1666 Téléphone: (514) 866-0871 www.schardy.qc.ca

Plan de la présentation

1. Contexte
2. L'énoncé **LIBNAME**
3. Flexibilité dans l'emploi du LIBNAME
4. Qui fait quoi – traçage de la requête
5. Connexion à l'aide de **SQL**
6. Conclusion

Annexe A – Spécificités du LIBNAME

Annexe B – Spécificités du couloir d'instructions SQL

1. Contexte

- Clé de voute: la pierre qui permet de faire tenir en place toutes les autres

1. Contexte

- Multiples méthodes de lecture existent, selon la source de données et selon la flexibilité dont nous avons besoin
- Deux méthodes privilégiées
  - **LIBNAME**
  - **CONNECT TO** (dans la procédure SQL)
- Notre focus: la lecture de données
- Illustrations avec Oracle, DB2, SQL Server, Sybase, My SQL, Microsoft-Access et Excel
- Spécificités propres à chaque source ne seront pas couvertes en détail – des exemples de connexion sont toutefois inclus

2. L'énoncé LIBNAME

- Syntaxe:

LIBNAME *surnom moteur paramètres-connexion*;

- *Surnom*: même caractéristiques que l'énoncé **LIBNAME** pour accéder à des tables SAS
- *Moteur:* nom codé de la source de données (tel *ORACLE*, *DB2* ou *EXCEL*)
- *Paramètres-connexion*: spécifiques à chaque base de données (voir annexe A)

2. L'énoncé LIBNAME (suite)

- Exemple (avec Oracle):

```
LIBNAME BD_VEN  ORACLE  USER="jsmith"
                        PASSWORD="sch"
                        PATH=VENTES;
DATA WORK.EXT(DROP=REGION);
  SET BD_VEN.CLIENTS(KEEP=PROV PAYS
                         REGION);
  WHERE REGION="02";
  LOC = CATX("-",PROV,PAYS);
  … suite-étape-DATA …
```

2. L'énoncé LIBNAME (suite)

- Dans l'exemple précédent:
  - Base de données *VENTES* contient une ou plusieurs tables
  - Table *CLIENTS* est importée pour en faire une table SAS temporaire, en filtrant les rangées et en effectuant des transformations de données complémentaires
  - Paramètres USER=, PASSWORD= , etc. propres à Oracle
  - Valeur de ces paramètres n'a pas besoin d'être codée « en dur » dans le programme

3. Flexibilité dans l'emploi du LIBNAME

- Souvent, les requêtes recourant au **LIBNAME** se limitent à une simple importation massive
- N'exploitent pas la flexibilité de la source de données et de SAS :
  - ✓ Lecture répétitive des données
  - ✓ Filtrage des rangées et choix des colonnes après l'importation
  - ✓ Tri des données après l'importation

3. Flexibilité dans l'emploi … LIBNAME (suite)

```
LIBNAME BD_VEN  moteur  param.-con.;
DATA T1;
  SET BD_VEN.CLIENTS;
RUN;
DATA T2;
  SET BD_VEN.PROSPECTS;
RUN;
DATA ALL(DROP=TYPE_CL);
  SET T1 T2;
  LOC = CATX("-",PROV,PAYS);
RUN;
PROC SORT  DATA=ALL;
  BY NAME;
RUN;
```

3. Flexibilité dans l'emploi … LIBNAME (suite)

- Pourquoi multiplier les étapes si tout le travail peut être réalisé dans la même étape (concaténation et intercalage):

```
LIBNAME BD_VEN  moteur  param.-con.;
DATA ALL(DROP=TYPE_CL);
   SET BD_VEN.CLIENTS;
       BD_VEN.PROSPECTS;
   BY NAME;
   LOC = CATX("-",PROV,PAYS);
RUN;
```

3. Flexibilité dans l'emploi … LIBNAME (suite)

- Pourquoi importer les données avant de les soumettre à une procédure ? Utiliser plutôt ceci:

```
LIBNAME BD_VEN  moteur  param.-con.;
PROC SUMMARY  DATA=BD_VEN.CLIENTS  NWAY;
  CLASS PAYS  TYPE_CL;
  VAR CODE_ACTIV;
  OUTPUT  OUT=AGG;
          N=NB_CLIENTS_ACTIFS;
RUN;
```

3. Flexibilité dans l'emploi … LIBNAME (suite)

- Pourquoi multiplier les étapes DATA…

```
LIBNAME BD_VEN  moteur  param.-con.;
DATA ACTUAL;
  SET BD_VEN.CL_ACTUAL;
RUN;
DATA PROSPECTS;
  SET BD_VEN.CL_PROSPECTS;
RUN;
DATA REFERRED;
  SET BD_VEN.CL_REFERRED;
RUN;
```

3. Flexibilité dans l'emploi … LIBNAME (suite)

- … lorsqu'une seule procédure peut faire tout le travail…:

```
LIBNAME BD_VEN  moteur  param.-con.;
PROC DATASETS  LIBRARY=BD_VEN  NOLIST;
  COPY  OUTLIB=WORK;
    SELECT  CL_ACTUAL  CL_PROSPECTS
            CL_REFERRED;
QUIT;
```

3. Flexibilité dans l'emploi … LIBNAME (suite)

- … et qu'on peut même éviter de nommer les tables (ici, toutes celles dont le nom débute par *CL_* sont copiées):

```
LIBNAME BD_VEN  moteur  param.-con.;
PROC DATASETS  LIBRARY=BD_VEN  NOLIST;
  COPY  OUTLIB=WORK;
    SELECT  CL_: ;
QUIT;
```

3. Flexibilité dans l'emploi … LIBNAME (suite)

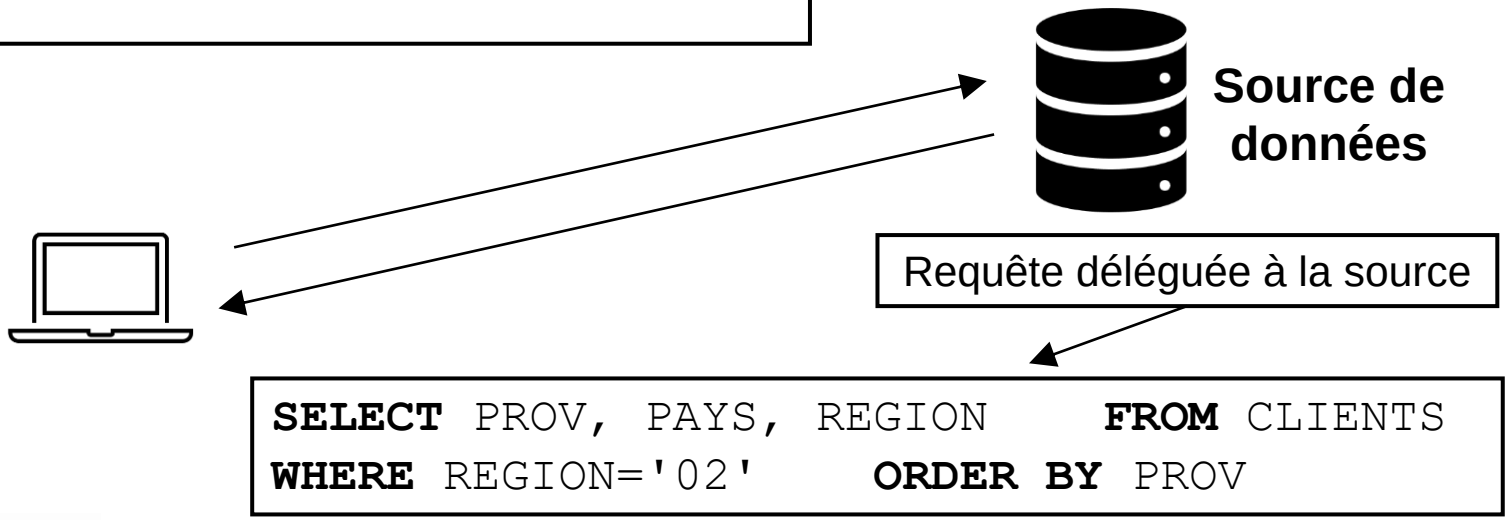
- Toutefois, si la même table est utilisée dans plusieurs étapes ou procédures SAS:
  - Les données sont transférées plusieurs fois de la source de données vers SAS, ce qui n'est pas optimal
  - Mieux vaudrait extraire les données
    - ✓en appliquant un filtre au besoin
    - ✓en réduisant si possible le nombre de colonnes
    - ✓en ajoutant un énoncé BY afin de faire trier les rangées par la source

4. Qui fait quoi – traçage de la requête

- Code SAS soumis → requête SQL exécutée par la source (DB2, Oracle, Excel, etc.)
- Une partie du code SAS soumis est interprété par la source
- Certaines sources sont plus « agiles » à transformer les énoncés SAS soumis en requête:
  - ✓Filtrage des rangées et des colonnes
  - ✓Tri des rangées
  - ✓Jointure
  - ✓Agrégation

4. Qui fait quoi – traçage de la requête (suite)

```
DATA EXT(DROP=REGION);
  SET BD_VEN.CLIENTS(KEEP=PROV PAYS
                         REGION);
  WHERE REGION="02";
  LOC = CATX("-",PROV,PAYS);
  BY PROV;
RUN;
```



4. Qui fait quoi – traçage de la requête (suite)

- Des options SAS permettent de tracer avec précision la requête exécutée par la source de données
- Utiliser pour des résultats optimaux:

```
OPTIONS   SASTRACELOC=SASLOG   SASTRACE=",,,D"
          NOSTSUFFIX;
```

- Diapo suivante: trace laissée dans le journal par l'exécution de la requête de la diapo précédente (source Ms-Access):

4. Qui fait quoi – traçage de la requête (suite)

```
23     DATA EXT(DROP=REGION);
24       SET MY_DB.CLIENTS(KEEP=PROV  PAYS  REGION);
25       WHERE REGION="02";
26       LOC = CATX("-",PROV,PAYS);
27       BY PROV;
28     RUN;

Jet_7: Prepared: on connection 0

SELECT  `PROV`, `PAYS`, `REGION`  FROM `CLIENTS`  WHERE
(`REGION` = '02')  ORDER BY `PROV`

Jet_8: Executed Prepared SQL Statement on connection= 0.
```

4. Qui fait quoi – traçage de la requête (suite)

- Ce qui a été délégué à la source de données ici:
  - ✓Filtrage des rangées
  - ✓Choix des colonnes
  - ✓Tri des rangées

- Ce qui n'a pas été délégué à la source de données ici:
  - ✓Concaténation des chaînes

- Ce qui est peut être délégué varie selon la source et selon la manière de rédiger le code

5. Connexion à l'aide de SQL

- Permet de contrôler très précisément ce qui est délégué à la source de données, mieux encore qu'avec **LIBNAME**

- Exige de connaître:
  - ✓ Syntaxe de la procédure SQL de SAS
  - ✓ Syntaxe du SQL de la source de données

- Connu sous le nom de *Pass-through facility* (« couloir d'instructions » en français)

5. Connexion à l'aide de SQL (suite)

- Syntaxe:

```
PROC SQL;
    CONNECT TO moteur (paramètres-de-connexion);
    < CREATE TABLE … AS >
    SELECT * < , expression . .. >
        FROM CONNECTION TO moteur
            (SELECT col-x, col-y, …
             FROM table
             < WHERE …. | ORDER BY … >)
        < WHERE … | ORDER BY … >;
QUIT;
```

5. Connexion à l'aide de SQL (suite)

- Dans la syntaxe montrée à la diapositive précédente:
  - **CONNECT TO** contient les paramètres de connexion
  - **SELECT** au sein de la parenthèse (aussi nommé *SELECT intérieur* - ici en gras) est exécuté par la source de données et doit suivre la syntaxe du SQL de la source
  - **SELECT** hors de la parenthèse (aussi nommé *SELECT extérieur*) suit la même syntaxe que d'habitude
  - Énoncé **DISCONNECT FROM** peut être ajouté avant de quitter, pour interrompre l'accès à la source
  - Particularités du **CONNECT** pour des sources : annexe B

5. Connexion à l'aide de SQL (suite)

- Exemple (ici même requête que diapo #19, avec MS-Access):

```
PROC SQL;
     CONNECT TO ACCESS (PATH="D:\HRD-2023\HR.accdb");
     CREATE TABLE EXT AS
       SELECT *  FROM CONNECTION TO ACCESS
         (SELECT PROV, PAYS,
                 (PROV & "-" & PAYS) AS LOC
          FROM CLIENTS
          WHERE REGION = "02"
          ORDER BY PROV
         );
      DISCONNECT FROM ACCESS;
QUIT;
```

5. Connexion à l'aide de SQL (suite)

- Certaines sources permettent d'accéder à un dictionnaire de données contenant le nom des tables ou des colonnes contenues dans la BD
- Nombreux paramètres de connexion permettent d'optimiser l'extraction, pour une source de données, que ce soit avec **LIBNAME** ou avec **CONNECT TO**

6. Conclusion

- **LIBNAME** et **CONNECT TO**: deux méthodes d'accès qui permettent d'optimiser l'extraction de données à partir d'une source
- Efforts consentis pour optimiser l'extraction, lorsque des volumes importants de données sont en jeu, valent presque toujours la peine
- Revisiter, à chaque changement majeur de version SAS, les pratiques qui se sont montrées efficaces dans le passé

POUR PLUS D'INFORMATIONS

Jean Hardy
Services Conseils Hardy Inc.
418-626-1666
jhardy@schardy.qc.ca
Web: www.schardy.qc.ca

Annexe A - Spécificités du LIBNAME – DB2 z/OS

- Exemple:

  LIBNAME DBHR *DB2* **SSID=HR AUTHID=JSMITH;**

- *SSID*: sous-système DB2
- *USER:* utilisateur (les paramètres *AUTHID=* ou *SCHEMA=* sont parfois utilisés à la place de *USER*, selon la configuration)
- *SERVER*: serveur DRDA, souvent facultatif
- D'autres paramètres de connexion peuvent et doivent parfois être utilisés, selon la source de données et la configuration de SAS dans votre organisation

Annexe A - Spécificités du LIBNAME – DB2 Unix

- Exemple:

```
LIBNAME DBHR DB2  DATABASE=HR  USER="jsmith"
                  PASSWORD="sch";
```

- *DATABASE*: base de données
- *USER:* utilisateur
- *PASSWORD*: mot de passe

Annexe A - Spécificités du LIBNAME – SQL Server

- Exemple:

```
LIBNAME DBHR SQLSVR  DATASRC=HR   USER="jsmith"
                     PASSWORD="sch";
```

- *DATASRC*: base de données ou source ODBC pour la lire
- *USER:* utilisateur
- *PASSWORD*: mot de passe

Annexe A - Spécificités du LIBNAME – MySQL

- Exemple (jamais testé):

LIBNAME DBHR *MYSQL* **DATABASE=HR USER="jsmith" PASSWORD="sch" PORT=9876 SERVER=PROD;**

- *DATABASE*: base de données
- *USER:* utilisateur
- *PASSWORD*: mot de passe
- *PORT*: port de communication
- *SERVER*: serveur MySQL

Annexe A - Spécificités du LIBNAME – Sybase

- Exemple:

```
LIBNAME DBHR SYBASE  DATABASE=HR USER="jsmith"
                     PASSWORD="sch"
                     SERVER=PROD;
```

- *DATABASE*: base de données
- *USER:* utilisateur
- *PASSWORD*: mot de passe, *SYBPW* et *PW* sont des synonymes
- *SERVER*: serveur Sybase

Annexe A - Spécificités du LIBNAME – ODBC

- Exemple:

```
LIBNAME DBHR ODBC  DATASRC=HR_ORA
                   USER="jsmith"
                   PASSWORD="sch";
```

- *DATASRC*: source de données ODBC (source générique ou spécifique à une base de données)
- *USER:* utilisateur
- *PASSWORD*: mot de passe

Annexe A - Spécificités du … – Ms-Access/Excel

- Exemples:

  LIBNAME DBHR *ACCESS* **"D:\HRD-2023\HR.accdb";**

  LIBNAME DBHR *EXCEL* **"D:\HRD-2023\HR.xlsx";**

- Suppose ici une version 64-bits de SAS et de Office
- D'autres moteurs et/ou paramètres seront requis si l'un ou l'autre des logiciels est en version 32-bits
- Alternative utile à l'habituelle procédure **IMPORT**

Annexe B – Spécificités du couloir d'instructions

- Paramètres comportant des enjeux de sécurité peuvent être encodés, notamment via **PROC PWENCODE** (idem qu'avec **LIBNAME**)

- ORACLE – exemple :

```
PROC SQL;
     CONNECT TO ORACLE (USER="jsmith"  PW="sch"
                        BUFFSIZE=150  PATH="@p:");
```

Annexe B – Spécificités du couloir d'instructions

- DB2 z/OS – exemple :

```
PROC SQL;
     CONNECT TO DB2 (SSID=HR  < SERVER=serveur-DRDA >);
```

- DB2 Unix – exemple :

```
PROC SQL;
     CONNECT TO DB2 (DATABASE=HR  USER="jsmith"
                                  PW="sch");
```

Annexe B – Spécificités du couloir d'instructions

- SYBASE – exemple :

```
PROC SQL;
     CONNECT TO SYBASE (INTERFACE="/sybase11/interfaces"
                        SERVER="sybd001"  DATABASE="hr"
                        USER="jsmith"  PASSWORD="sch");
```

- SQL SERVER – exemple :

```
PROC SQL;
     CONNECT TO SQLSVR (USER="jsmith"  PASSWORD="sch"
                        DATASRC=HR);
```

Annexe B – Spécificités du couloir d'instructions

- My Sql – exemple:

```
PROC SQL;
     CONNECT TO MYSQL (USER="jsmith"  PASSWORD="sch"
                       DATABASE=HR  PORT=9876
                       SERVER=PROD);
```

- ODBC – exemple (*DATASRC=* est un synonyme de *DSN=*):

```
PROC SQL;
     CONNECT TO ODBC (USER="jsmith"  PASSWORD="sch"
                      DSN=HR_ORA);
```

Annexe B – Spécificités du couloir d'instructions

- Excel – exemple (Excel et SAS en 64-bits):

```
PROC SQL;
     CONNECT TO EXCEL (PATH="chemin/classeur.xlsx");
```

- Excel – exemple (Excel 32-bits et SAS en 64-bits):

```
PROC SQL;
     CONNECT TO PCFILES (PATH="chemin/classeur.xlsx"
                         SERVER="serveur-du-PC-FILES");
```

Annexe B – Spécificités du couloir d'instructions

- Ms-Access – exemple (Access et SAS en 64-bits) :

```
PROC SQL;
     CONNECT TO ACCESS (PATH="chemin/nom-bd.accdb");
```

- Ms-Access – exemple (Access 32-bits et SAS en 64-bits):

```
PROC SQL;
     CONNECT TO ACCESS (PATH="chemin/classeur.xlsx"
                        SERVER="serveur-du-PC-FILES");
```