SESUG Paper 137-2023

Implementing GIT for Change Control of Delegated Tables Owned by End Users

Angye Rivero Rodríguez, Osmel Brito Bigott, DatAnalítica;
Stephanie Inostroza Arratia, Universidad Católica de Chile

ABSTRACT

Applications commonly use data tables that are maintained by users, also known as delegated tables. Delegated tables enable business users to maintain master or reference data in spreadsheets or comma-delimited files and load that data directly into the application. Multi-copy files, no control over updating them, inability to retrieve previous information, and other uncomfortable situations are very common in these situations.

In this paper, we will use the functions of Git in SAS to present an option, which will allow users to solve in an easy and practical way how to keep these delegated tables used by their business applications updated, integrating this version control system and all its benefits with SAS programs; enabling them to maintain their data the way they have always done, but in a controlled manner.

INTRODUCTION

In organizations, data is always generated and shared, often recorded in desktop application files (Excel, CSV, Access) that are inputs to operational systems and their update process involves several people, several physical copies of the same file and even different directories that contain the same repeated file, which makes control and maintenance difficult, generating problems of duplicity, inconsistency or even loss of valuable information. In this paper we will show how to better manage these sources of information, eliminating the aforementioned problems by making use of the GIT functions embedded in SAS, so that the data can be made available faster, with higher quality and providing agility to the processes. of business..

THE FIRST THING TO KNOW

GIT is a distributed version control system. Generally when you hear Git it is related to version control of a project's scheduled source code, but we can also use it to control changes to files such as spreadsheets, documents, and others. Git is a system that guarantees the efficiency and reliability of maintaining complete file versions. With Git we can compare the changes made over time, see who modified something and know when the change was made, having the option to revert it. Git manages remote repositories and local clones of them where each business end user can make changes collaboratively and share it with other users by committing their changes locally and then synchronizing them with the remote repository.

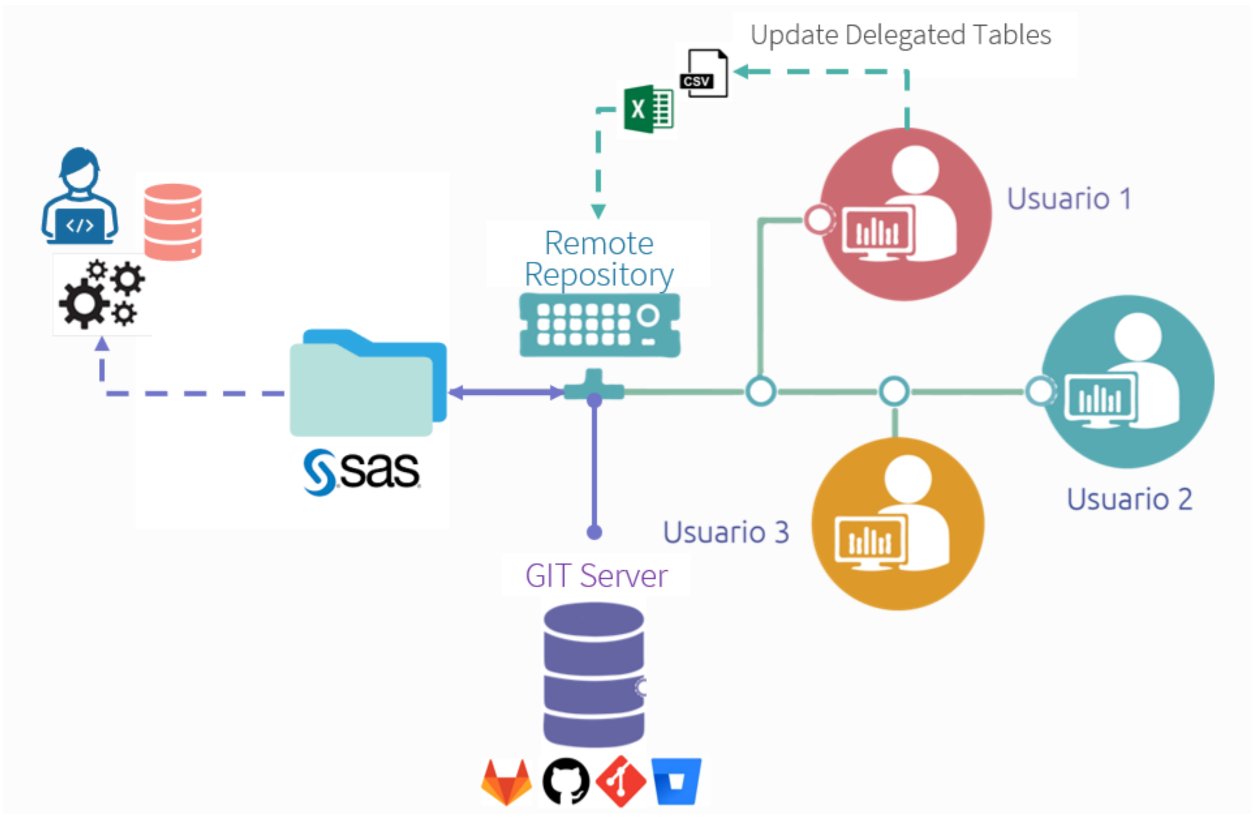
AND, WHAT ARE DELEGATED TABLES?

They are files that contain reference or master data that are customer-mantained. These files can be spreadsheets, comma-delimited files, among others, and are input sources of information for the organization's processes and applications.

Start working with a distributed version control system like Git would change the way in which business teams administer and mantain delegated tables, obtaining great benefits from this implementation such as:

   - • Easy maintenance of reference data in formats already known to users, such as Excel, CSV, etc.
   - • Change management of delegated tables.
   - • Greater access control to delegated tables.
   - • Compliance with auditing standards since there is an auditable change trace. This is usually an internal requirement of the organization or regulatory compliance

- Podemos construir flujos de trabajo que realicen la actualización de las tablas y automatizarlos para que puedan ejecutarse de manera automática.



NOW, WE WILL SHOW HOW TO IMPLEMENT GIT TO UPDATE DELEGATED TABLES WITH SAS

SAS language has functions that we can use directly in our programs to take advantage of all the benefits and functionality of Git. In this paper we will only mention the ones we used for the implementation, which are:

GIT CLONE

Git clone basically makes an identical copy of the latest version of files that are in a remote repository in our local repository.

```
data _null_;
   rc = git_clone("remote-repository-url",
                  "target-directory",
                  "user-name",
                  "password",
                  "ssh-public-key",
                  "ssh-private-key");
run;
```

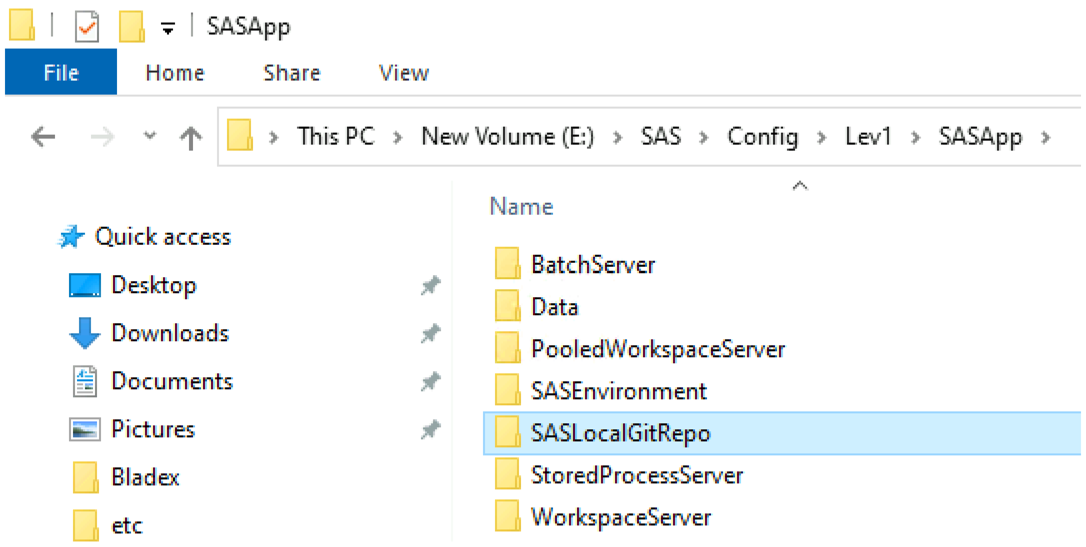
GIT PULL

It is used to receive updates from the remote repository and these latest changes are immediately applied to our local repository.

```
data _null_;
   rc= git_pull("your-local-repository",
                "user-name",
                "password",
                "ssh-public-key",
                "ssh-private-key");
run;
```

INITIAL STEPS

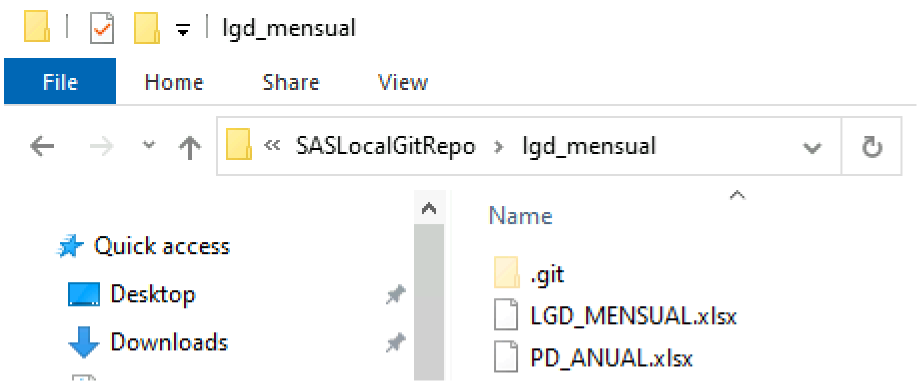
Create a folder where the local repositories(s) will be stored on the SAS server. It is advisable to keep our directories organized within the SAS server, in addition to this way business users will not have direct access to the delegated tables, providing better control, they will only be accessed by the SAS programs responsible for importing the data and creating the tables in SAS libraries. :



Then we must clone the remote repository, for this example we will call it lgd_mensual. This can be done directly by the Git command line or by running a SAS program by using the clone function. For our example we assume that the remote repository requires authentication username and password, being as follows:

```
data _null_;
  rc = git_clone(
               "https://github.com/datanalitica-hub/lgd_mensual.git",
               " E:\SAS\Config\Lev1\SASApp\SASLocalGitRepo",
               "user-git",
               "{SAS002}DA9A0A5C20629B7F34D2C88A165E5530",
               );
  put rc=;
run;
```

Creating the local repository within the folder that we defined for this purpose on the server



Display 2. Local Repository

UPDATE DELEGATED TABLES WITH SAS

To update the delegated tables with SAS we only need to create a program that executes a Pull to extract the changes from the remote repository that have been sent by the different users responsible for the maintenance of the reference tables in their different formats such as .xlsx, .csv, among others. In our example:

```
%global rc_git;

data _null_;
   rc= git_pull(" E:\SAS\Config\Lev1\SASApp\SASLocalGitRepo",
                "user-git",
                "{SAS002}DA9A0A5C20629B7F34D2C88A165E5530");

   call symput("rc_git",rc);
run;

%put NOTE: Result of GitPull: &=rc_git;
```

It is very important to note that the return code generated by the git_pull function is the one that tells us if the Pull has been done correctly and will be the one that helps us define the logic of action and decision of our SAS programs according to what we want to do. So we capture the value in a macro variable of global scope which we define as rc_git.

We will highlight some of the return codes, which we will use in the process of updating the delegated tables in SAS:

Return Code	**Description**
0	The pull operation was successful and the local repository was updated.
1	The local repository is up-to-date and there are no new changes to pull from the remote repository.
Other	The pull operation failed.

Table 1. Git_Pull function Return Codes

En la sección de referencias pueden encontrar el resto de los valores de return codes generados al ejecutar la función `git_pull`.

Once the Pull operation has been executed we must evaluate the return code and take action based on the result.

Our goal is to update delegate tables, so we need to evaluate three scenarios:

- The delegate tables have changed, that is, the file has been updated in the remote repository and we must update these tables in the SAS library.
- The delegate tables have not changed, which means that there are no new changes in the remote repository and we do not need to update the tables in the SAS library.
- The Pull operation failed while attempting to update the local repository.

Based on these scenarios we can implement a program like the one shown below:

```
%macro UpdateLibGit;

   %if &rc_git = 0 %then %do;

       libname lib_dat base "F:\MyLibraries";

       proc import dbms=xlsx
        file="E:\SAS\Config\Lev1\SASApp\SASLocalGitRepo\Monthly_Data.xlsx"
        out=lib_dat.table_monthly_data
        replace;
      run;

   %end;
   %else %do;

      %if &rc_git = 1 %then %do;

         /* The local repository is up-to-date and there are no new
            changes to pull from the remote repository */

         %put NOTE: UpdateLibGit: The local repository is up-to-date. No
                    need to update the delegated tables in lib_dat;

      %end;
      %else %do;

         /* If rc_git <> (0,1) generate an alert because the Pull was not
            executed correctly
            Check the execution log to see the details of the failure */

         %put WARNING: UpdateLibGit: The pull operation failed;

      %end;

   %end;

%mend UpdateLibGit;
%UpdateLibGit;
```

We can supplement our program in many ways, such as sending an email notification that there is an update of the information or performing other types of tasks according to our needs.

A significant value that we can add to a business process like this is that we can schedule it in SAS Management Console to execute automatically and in this way the data in the SAS libraries are kept updated in an agile way thanks to the integration of SAS process flows with the technology and benefits offered by Git.

CONCLUSION

Updating tables maintained by business users from files such as spreadsheets among others through technologies such as SAS and GIT allows multidisciplinary organizational environments where a data source can be updated by one or more business units and people, to maintain greater control and audit over their delegated tables, which are frequently imported into SAS libraries and then this data is used in daily processes to generate valuable information through different SAS workflows.

Carrying out this implementation improves the way in which work teams manage their data, creating more confidence in the information generated, benefits every aspect of the organization's processes and encourages the development of agile businesses which translates into competitive advantages.

REFERENCES

Burns, Larry Data Model Storytelling: An Agile Approach to Maximizing the Value of Data Management. (2021). (n.p.): Technics Publications.

SAS® Help Center. “Using Git Functions in SAS”. SAS® 9.4 and SAS® Viya® 3.5 Programming Documentation. August 15, 2023. SAS Help Center: Using Git Functions in SAS

SAS® Help Center. “GIT_CLONE Function”. SAS® 9.4 and SAS® Viya® 3.5 Programming Documentation. August 15, 2023. SAS Help Center: GIT_CLONE Function

SAS® Help Center. “GIT_PULL Function”. SAS® 9.4 and SAS® Viya® 3.5 Programming Documentation. August 15, 2023. SAS Help Center: GIT_PULL Function

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Angye Rivero Rodríguez (angye.r@datanalitica.com)
Osmel Brito Bigott (osmel.b@datanalitica.com)
DatAnalítica

info@datanalitica.com

7801 NW 37TH ST
DO80Q75172N
DORAL,
FLORIDA 33195-6503