CHAPTER

26

Syntax for the SIGNON and the SIGNOFF Commands and Statements

Introduction

The SIGNON and SIGNOFF commands and statements are used to initiate and to terminate a link between a local and a remote host. The syntax for SIGNON and SIGNOFF allows you to specify many of the SAS system options in the command or the statement. See “SIGNON Command and Statement” on page 227, “SIGNOFF Command and Statement” on page 233, “FILENAME Statement” on page 236.

Dictionary

SIGNON Command and Statement

Initiates a link between a local SAS session and a remote SAS session.

Local

Syntax

SIGNON <*remote-session-id*><CONNECTWAIT=YES | NO>
<MACVAR=*value*><CONNECTSTATUS=*value*> <SCRIPT=*value*> <*fileref*>
<’*filespec*’> <NOSCRIPT> <USER=*username* | _PROMPT_>
<PASSWORD=*password* | _PROMPT_>
<SASCMD=*SAS-command1*...*SAS-commandn*’>;

Syntax Description

The options are used for initiating the link and interactions between the local and the remote sessions. The following options or any combination of the following options may be used in a SIGNON command:

remote-session-id
CONNECTREMOTE=*remote-session-id*
REMOTE=*remote-session-id*
PROCESS=*remote-session-id*

specifies the name of the remote session in which you want to sign on. After the *remote-session-id* has been specified by either the SIGNON command or as a system option, subsequent RSUBMIT, SIGNON, or SIGNOFF commands or statements that omit *remote-session-id* default to the specified session id.

PROCESS= was made an alias for REMOTE= in order to give you the option of differentiating between a SIGNON to a remote session on a local host (MP CONNECT) and a SIGNON to a remote session on a remote host. REMOTE= and PROCESS= can be used interchangeably.

CONNECTWAIT=*value*
WAIT=*value*

specifies whether remote submits are to be executed synchronously or asynchronously by default during a SAS session. The default setting can be overridden by specifying the CONNECTWAIT= option in subsequent RSUBMIT statements for a specific remote submit.

In synchronous processing, you must wait for the remote processing to complete before control in the local SAS session is returned to you. This is the default processing technique for RSUBMIT if the WAIT option is not specified. In asynchronous processing, after the RSUBMIT block begins to execute on the remote host, you will regain control of your local SAS session to continue local processing or to continue to RSUBMIT to other remote sessions.

The *value* for the WAIT= option must be one of these:

YES | Y indicates to execute RSUBMITs synchronously.

NO | N indicates to execute RSUBMITs asynchronously.

If WAIT=NO is specified, it is also useful to specify the MACVAR= option in the RSUBMIT statement. This allows you to test the status of the current asynchronous RSUBMIT by determining whether processing has completed or is still in progress.

When %SYSRPUT executes within a synchronous WAIT=YES) remote submit, the macro variable is defined to the local SAS session as soon as it executes.

When %SYSRPUT is executed within an asynchronous (WAIT=NO) remote submit, the macro variable, by default, is not set in the local session until a synchronization point. See “%SYSRPUT Statement” on page 29 for more information about synchronization points.

MACVAR=*value*

specifies the name of the macro variable to associate with this remote session. The macro variable will NOT be set if the SIGNON command fails due to incorrect syntax. Other than this one exception, the macro variable is set when the SIGNON command is completed. The variable will have one of the following values:

0 indicates that the SIGNON was successful.

1 indicates that the SIGNON failed.

2 indicates that you have already signed on to this remote session.

If the SIGNON is successful, the macro variable is set, and it becomes the default macro variable for this remote session. This default can only be overridden by a subsequent successful RSUBMIT command that has the MACVAR= option specified. There is no other way to change the default for this remote session. Therefore, if the user attempts to sign on to the same remote session a second time and specifies a different value for MACVAR=, the macro variable associated with the second sign on attempt does not become the default, but the macro variable will be set to indicate that the user had already signed on.

CONNECTSTATUS=*value*
STATUS=*value*

specifies the default setting for the display of the status window. The *value* for this option must be one of these:

YES | Y	Status window is displayed during file transfers.
NO | N	Status window is not displayed during file transfers.

If this option is omitted from the SIGNON statement, the Status window is displayed during transfers by default. This option can also be specified on subsequent RSUBMIT and PROC UPLOAD and DOWNLOAD statements so that the default may be overridden for specific transfers.

SCRIPT=*value*

specifies the script file to be used during sign on. It may either be a fileref or a quoted, fully-qualified pathname. If the fileref, the filespec, and the SCRIPT= option are specified, the last specification overrides and takes precedence over the others.

When the SIGNON command executes, the usual SAS log messages for the remote SAS System display in your local LOG window. When the link has been successfully established, the following message is displayed:

```
NOTE: REMOTE SIGNON TO remote-session-id
COMPLETE.
```

fileref

is the name of the reference file associated with the script that starts the link. The fileref must be defined by a previously executed FILENAME statement.

If the fileref you define for the script is the default fileref RLINK, omit this specification in the SIGNON command.

'filespec'

is the name of the script you want to execute. Use the filespec in the SIGNON command when no fileref has been defined for the script that you want to execute. The filespec can be either a fully qualified filename or the name of a file in the current working directory. Do not specify both a fileref and a filespec.

NOSCRIPT
NOCSCRIPT

specifies that no script should be used to sign on. This is useful if you have already invoked the SAS session on the remote host, and you do not need any additional script functionality. When you use NOSCRIPT, the SIGNON command executes faster and requires less memory, but no special functions invoke a SAS session on the remote host or perform any special functions such as error processing or logging on to the remote host.

USER | USERNAME | USERID | UID=*username* | _PROMPT_

valid values that can be assigned to USER are

username

For details about a valid username, see "Username and Password Naming Conventions" on page 230.

PROMPT

a secure method, specifies that SAS prompt the user for a valid username.

PASSWORD | PASSWD | PWD | PW=*password* | _PROMPT_

valid values that can be assigned to PASSWORD are:

password
For details about a valid password, see “Username and Password Naming Conventions” on page 230.

PROMPT
a secure method, specifies that SAS prompt the user for a valid password.

SASCMD=“*SAS-command1*” ... “*SAS-commandn*”
For MP CONNECT signon only, is the command that invokes the remote SAS session.

Note: SASCMD= is also a global SAS system option. If the SASCMD= global system option is already set, a locally set SASCMD= option in the SIGNON command would take precedence over the globally set option. See “SASCMD=” on page 223 for more information. △

Here is a typical SASCMD option setting:

```
SASCMD="sas"
```

Details

The primary difference between the SIGNON command and the statement is that the command can be used only from the command line of the windowing environment or within the DM statement. The SIGNON statement must be followed by a semicolon (;) and can be used in any SAS session on the local host.

Username and Password Naming Conventions

Each username and password is limited to 256 characters that follow these conventions:

□ Mixed case is allowed.

□ A null value, which is no value, that is delimited with quotation marks is allowed.

□ Quotation marks must surround values that contain one or more spaces.

□ Quotation marks must surround values that contain one or more special characters.

□ Quotation marks must surround values that contain one or more quotation marks.

Examples:

```
user=joe password=Born2run
user=joe password='' # null space specified by contiguous quotation marks
user='joe black' password='Born 2 run'
user='joe?black' password='Born 2 run'
user='apexdomain\joe' password=born2run # Win NT username
user='"crazy joe"' pw=_prompt_;
user=_prompt_;
```

Examples of the SIGNON Command and the SIGNON Statement

These examples use the SIGNON statement, which requires a semicolon (;). The SIGNON command does not use the semicolon.

Suppose you specify the name of the session in an OPTIONS statement:

```
options remote=rhost;
```

You then assign the fileref RLINK to your script by using the following FILENAME statement, where *external-file-name* is your script:

```
filename rlink 'external-file-name';
```

Now you can use the short form of the SIGNON command or statement to initiate a link, assuming that the default access method for your host is TCP/IP.

```
signon;
```

If you assign the fileref RLINK to your script, you can specify the remote-session-id in the SIGNON statement without having to issue an OPTIONS statement:

```
signon rhost;
```

You can assign some other fileref to the script, as in the following example:

```
filename startup 'external-file-name';
```

You can now issue this SIGNON command or statement:

```
signon startup cremote=ahost;
```

In Version 8, the ability to provide USER= and PASSWORD= options directly to the SIGNON statement and command supercedes the use of the Version 7 host-specific security option.

For example, in Version 7, if using the TCP/IP access method to make a secure connection between a local host that runs on a Windows NT and a secure spawner running on a remote UNIX host that runs onUNIX, you set this security option:

```
options set tcpsec=_prompt_;
```

In Version 8, the preferred method for making a secure connection is to specify security options directly to the SIGNON command or statement. Here is a Version 8 example:

```
signon user=_prompt_;
```

At signon, the user is prompted for username and password.

If you are using RLINK or any other fileref in the SIGNON command or statement, remember that you can define the fileref for the script in a FILENAME statement in the SAS autoexec file.

The SIGNON statement is particularly useful for running SAS/CONNECT from an interactive line-mode session or a non-interactive job. For line-mode and non-interactive sessions, use the SIGNON statement in combination with the SIGNOFF, RSUBMIT, and ENDRSUBMIT statements.

Examples of SIGNON, RSUBMIT, ENDRSUBMIT, and SIGNOFF Statements

Suppose you want to use two remote hosts to execute SAS programs and download data to a local host. This example assigns a fileref for script files on remote hosts; one on a on a Windows NT remote host that runs an unsecured spawner , and another for an OS/390 remote host. The local host is a UNIX system, so the remote-session-ids are the names of the host machines at the site. In this example, the two remote hosts are named TSO and WNT.

The following program can be submitted on a local UNIX host from the Program Editor window, interactive, or non-interactive line-mode session:

```
/*****************************************/
/* set communications access method      */
```

```
   /*****************************************/
options comamid=tcp;

   /*****************************************/
   /* initiates link to a Window NT remote   */
   /* host                                   */
   /*****************************************/
signon wnt;

   /*****************************************/
   /* initiates link to an OS/390 remote host*/
   /*****************************************/
filename tsoscr '!sasroot/misc/connect/tcptso.scr';
signon tso script=tsoscr;

   /*****************************************/
   /* submit statements to a Windows NT      */
   /* remote host                            */
   /*****************************************/
rsubmit wnt wait=no;
   statements to be processed by Windows NT
    remote host
endrsubmit;

   /*****************************************/
   /* submit statements to OS/390 remote host*/
   /*****************************************/
rsubmit tso wait=no;
   statements to be processed by OS/390 remote host
endrsubmit;
waitfor _ALL_ wnt tso;
   /*****************************************/
   /* terminates both links                  */
   /*****************************************/
signoff tso cscript=tsoscr;
signoff wnt cscript=winscr;
```

The statements enclosed by the RSUBMIT and ENDRSUBMIT statements are processed by the remote host. The remaining statements are processed by the local host.

The following example illustrates that the macro variable from a successful SIGNON will be used if an unsuccessful attempt is made.

```
   /*****************************************/
   /* sign on successful, rhost1 will be     */
   /* set to 0 to indicate success.          */
signon rhost macvar=rhost1;

   /*****************************************/
   /* sign on fails because we have already  */
   /* signed on to this remote session,      */
   /* so rhost2 will be set to 2 to          */
   /* indicate this, but rhost1 will         */
   /* still be the MACVAR associated         */
   /* with rhost.                            */
   /******************************************/
signon rhost macvar=rhost2;

rsubmit rhost wait=no;
   data a;
   x=1;
   run;
endrsubmit;

   /******************************************/
   /* rhost1 is still the default and        */
   /* will indicate the progress of any      */
   /* subsequent RSUBMITs.                   */
   /******************************************/
%put &rhost1;
```

SIGNOFF Command and Statement

Terminates the link between a local SAS session and a remote SAS session.

Local

Syntax

SIGNOFF <*remote-session-id*> <MACVAR=*value*> <CSCRIPT=*value*> *fileref*<'*filespec*'> <NOSCRIPT>;

Syntax Description

Options specify defaults for terminating the link between the local and the remote sessions.

Details

The SIGNOFF command and the SIGNOFF statement terminate a link between a local SAS session and a remote SAS session and execute a script if you are using an access method that requires a script file. Issue the SIGNOFF command from the command line of any local SAS windowing environment window or in a DM statement. You can also issue a SIGNOFF statement from the local SAS session, which is especially useful for interactive line-mode sessions or non-interactive jobs.

The following options or any combination of these options may be used in a SIGNOFF command:

remote-session-id
CONNECTREMOTE=*remote-session-id*
REMOTE=*remote-session-id*
PROCESS=*remote-session-id*

specifies the name of the remote session from which you want to sign off. After the *remote-session-id* has been specified by either the SIGNON command or as a system option, subsequent RSUBMIT, SIGNON, or SIGNOFF commands or statements that omit *remote-session-id* default to the specified session id.

When you specify *remote-session-id*, the script that is associated with the default fileref RLINK is executed. In this case, you must have a valid script file associated with the fileref RLINK.

PROCESS= was made an alias for REMOTE= in order to give you the option of differentiating between a SIGNOFF from a remote session on a local host (MP CONNECT) and a SIGNOFF from a remote session on a remote host. REMOTE= and PROCESS= can be used interchangeably.

MACVAR=*value*

specifies the name of the macro variable to be associated with this remote session. The macro variable is NOT set if the SIGNOFF command fails due to incorrect syntax. Other than this one exception, the macro variable is set when the SIGNOFF command is completed. The variable will have one of the following values:

0	indicates that the SIGNOFF was successful.
1	indicates that the SIGNOFF failed.

If the SIGNOFF is successful, the macro variable is set to 0. However, because the link is now terminated, the macro variable is no longer associated with that remote session. If the SIGNOFF fails, the macro variable that is specified in the SIGNOFF is set to 1. Even though it was an unsuccessful SIGNOFF, the macro variable is not associated with that remote session; instead the default MACVAR= (if any) is used.

CSCRIPT=*value*

specifies the script file to be used during SIGNOFF. CSCRIPT may be a fileref or a quoted, fully-qualified pathname. If the *fileref*, the *filespec*, and the CSCRIPT= options are specified, the last specification takes precedence over the others.

fileref

is a filename associated with the script that terminates the link. The fileref must be defined by a previously executed FILENAME statement.

At many computing installations, you use the same script to start and terminate a link. If you use one script to do both, you need to assign only one fileref.

If the fileref you define for the script is the default fileref RLINK, omit this specification in the SIGNOFF command.

’*filespec*’

is the filename of the script that you want to execute. Use the filespec in the SIGNOFF command when no fileref has been defined for the script that you want to execute. The filespec can be either a fully qualified filename or the name of a file in the current working directory. Do not specify both a fileref and a filespec.

NOSCRIPT
NOCSCRIPT

specifies that no script should be used to sign off. This is useful if you do not need any additional script functionality. When you use NOSCRIPT, the SIGNOFF command executes faster and requires less memory, but no special functions such as logging off the remote host or error processing, are performed.

Examples

The following example demonstrates that the MACVAR variable is used if the sign off is unsuccessful:

```
   /* Sign on successful, rhost1 will be */
   /* set to 0 to indicate success, and  */
   /* macro variable rhost1 is now       */
   /* associated with this remote        */
   /* session.                           */
signon rhost macvar=rhost1;

   /* Sign off will fail, and rhost2     */
   /* will be set to 1 to indicate this, */
   /* but because it was unsuccessful,   */
   /* rhost1 is still the default macro  */
   /* variable associated with this      */
   /* remote session.                    */
signoff rhost macvar=rhost2
   script='noexist.scr';
```

In this example, no macro variable was assigned at sign on, therefore if the sign off is not successful, no macro variable will be set for this remote session.

```
   /* No macro variable associated with  */
   /* remote session                     */
signon rhost;

   /* Sign off will fail, and ABC will   */
   /* be set to 1 to indicate this,      */
   /* but because it was unsuccessful,   */
   /* the default of no macro variable   */
   /* will go into effect for this       */
   /* remote session.                    */
signoff rhost macvar=abc
   cscript='noexist.scr';
```

When the SIGNOFF command executes, the usual SAS log messages for the remote SAS System appear in the Log window on your local host. When the link has been terminated, the following message is displayed:

```
NOTE: REMOTE SIGNOFF TO remote-session-id
      COMPLETE.
```

Suppose you assign the fileref RLINK to your script with the following FILENAME statement, where *external-file-name* is the name of your script:

```
filename rlink 'external-file-name';
```

Then you can use the short form of the SIGNOFF command or statement to terminate the link:

```
signoff;
```

If you have multiple remote sessions executing, you can specify which session to sign off by using the *remote-session-id*.

```
signoff ahost;
```

Suppose that you assign some other fileref to the script:

```
filename endit 'external-file-name';
```

Then you must specify the fileref in the SIGNOFF command or statement because it is not the default script fileref:

```
signoff cscript=endit;
```

When you are using a fileref other than RLINK for the signoff script and you have multiple remote sessions executing, use the CREMOTE= option to specify which session should be ended.

```
signoff cremote=ahost cscript=endit;
```

If you are using RLINK or any other fileref in the SIGNOFF statement, remember that you can define the script's fileref in a FILENAME statement in the SAS autoexec file. Then you do not have to type and execute the FILENAME statement before you use the SIGNOFF command.

If you do not assign any fileref to the script, you must specify the filespec in the SIGNOFF command. For example, you can use the following command:

```
signoff 'external-file-name';
```

If you do not want to perform any special processing when you sign off, you can omit the script that is used for signing off, as in this example:

```
signoff noscript;
```

FILENAME Statement

Associates a SAS fileref with an external file.

Local and remote

Syntax

FILENAME *fileref* '*filespec*' <*host-options*>;

Syntax Description

fileref
specifies any SAS file reference name.

filespec
specifies the physical name of an external file. The physical name is specified so that the external file is recognized by the operating environment.

host-options
specify details, such as file attributes and processing attributes that are specific to the operating environment.

Details

The FILENAME statement associates a SAS *fileref* (a file reference name) with a *filespec* (a host-dependent external filename). The fileref must conform to the rules for

SAS names. The form of the filespec differs from system to system. On some systems you must include a fully-qualified filename. Other systems may permit partial pathnames.

Filerefs are a shorthand method for specifying a file in SAS statements and commands. After you define a fileref, you can use the fileref in place of the longer file specification to reference the file throughout a SAS session or program. The FILENAME statement and filerefs are useful in many SAS programs, not just with SAS/CONNECT.

Note: A fileref remains associated with an external file for the duration of the SAS session only. The association is not permanent. Also, a fileref must be defined before a SAS statement or command that uses the fileref can execute. Therefore, the FILENAME statement must execute before any statement or command that uses the fileref. △

With Script Files

A common use of the FILENAME statement in SAS/CONNECT is to define filerefs for script files.* A script's fileref can then be specified in SIGNON and SIGNOFF commands to identify the script that starts or terminates the link.

You can define a default fileref for a script in a FILENAME statement. The default script fileref is RLINK. If you specify RLINK as the fileref for your script, you do not need to specify a fileref or a filespec in SIGNON and SIGNOFF commands or statements; you simply use the command or the statement. When SAS executes a SIGNON or a SIGNOFF command without a fileref or a filespec, SAS automatically searches for a file that is defined with RLINK as the fileref. If RLINK has been defined, SAS executes the corresponding script.

Examples

Suppose your SAS Support Consultant writes a script and copies it to a directory on your local system. If you want to define the default fileref RLINK for your script, use the following FILENAME statement:

```
filename rlink 'external-file-name';
```

Because you defined RLINK as the script's fileref, you can use the shortest form of the SIGNON and SIGNOFF commands or statements. For example, to start the link, enter:

```
signon;
```

To terminate the link, enter:

```
signoff;
```

If you use one script to start the link and a different script to terminate the link, you must define a unique fileref for each script. For example, you can use the following FILENAME statements:

```
filename rlink 'start-link-script-file';
filename endit 'end-link-script-file';
```

Subsequently, to start the link, enter the following command or statement, which uses the default fileref RLINK for the sign-on script.

```
signon;
```

* You can also use the FILENAME statement to specify an alternate 3270 keyboard translation table.

To terminate the link, enter:

```
signoff endit;
```

In the SAS Autoexec File

You can make starting and ending SAS/CONNECT even easier by adding the FILENAME statement that defines the script's fileref to an autoexec file. An *autoexec file* is a file of SAS statements and commands that you can set up to execute automatically each time you invoke the SAS System. Its purpose is to automate the execution of statements, commands, and entire programs that you use routinely in SAS processing. If you use an autoexec file that contains a FILENAME statement that defines your script's fileref, you do not have to type and execute the FILENAME statement each time you want to invoke the link.

Refer to the SAS Companion documentation for your environment and *SAS Language Reference: Concepts* for more information about setting up an autoexec file.

With the DOWNLOAD and UPLOAD Procedures

You can also use the FILENAME statement with the DOWNLOAD and UPLOAD procedures when copying external files. When you submit the FILENAME statement to define the fileref for the external file on the local host, you remote-submit the FILENAME statement to define the fileref for the external file on the remote host.

Example

Suppose you want to download an external file from the remote host to a directory-based local system. You submit the following FILENAME statement to assign the fileref on the local host:

```
filename lhost 'local-file-name';
```

Then remote-submit these statements to assign the fileref on the remote host and perform the download:

```
filename rhost 'remote-file-name';

   proc download infile=rhost outfile=lhost;
   run;
```

See Chapter 16, "Using Data Transfer Services," on page 103 for more examples of the FILENAME statement and the DOWNLOAD and UPLOAD procedures.

The correct bibliographic citation for this manual is as follows: SAS Institute Inc., *SAS/ CONNECT User's Guide, Version 8*, Cary, NC: SAS Institute Inc., 1999. pp. 537.

SAS/CONNECT User's Guide, Version 8

ISBN 1–58025–477–2

SAS Institute Inc., SAS Campus Drive, Cary, North Carolina 27513.

1st printing, September 1999

The Institute is a private company devoted to the support and further development of its software and related services.