CHAPTER

4

Using the SQL Query Window in SAS/AF Applications

SAS/AF Classes for the SQL Query Window

Several Screen Control Language (SCL) methods are available that enable you to use the SQL Query Window in SAS/AF applications. Refer to *SAS Component Language: Reference* for more information on SCL methods. Examples are shown later in this chapter on using SCL methods with the SQL Query Window.

WHERE Expression Builder Class

QWWHERE is the WHERE expression builder class that builds a WHERE expression by using columns from existing SAS data sets or SAS views. You can add the WHERE expression builder in the SQL Query Window to any SAS/AF application. The QWWHERE class provides the following methods:

WHERE_BLD

WHERE_BLD builds an available columns list and invokes the WHERE window. The WHERE_BLD method takes the following arguments:

SQLLIST type=listid use=IN/OUT
: The master SCL list that is used by the WHERE expression builder. The first time that the WHERE expression is called, the application creates an empty list and passes it as the SQLLIST.

DSNAME type=char use=IN
: The SAS data set or data sets from which the available columns are created. If this argument is blank, WHERE_BLD assumes that the available column list has already been built.

WHERELST type=listid use=OUT
: Contains the WHERE list. If more than one data set has been used as input, the WHERE clause contains MEMNAME.COLUMNNAME. If one data set has been used as input, the WHERE clause contains COLUMNNAME only. This argument is optional.

WHTITLE type=char use=IN
: Defines the title for the WHERE window. This argument is optional.

AVTITLE type=char use=IN
: Defines the title for the available column selection list in the WHERE window. This argument is optional.

NOPROMPT type=char use=IN
: Setting this argument disables <PROMPT at Run Time> as an available column selection choice. If this argument is set, GET_TEXT_LIST is no longer necessary. This argument is optional.

NOLOOKUP type=char use=IN
: Setting this argument disables <LOOKUP distinct values> as an available column selection choice. Enabling <LOOKUP distinct values> allows the user to view

lookup values for a column while in the WHERE window. This argument is optional.

AUTOLOOK type=char use=IN
Specifies the data set that can be included to make use of customized automatic lookup. See “Using the Automatic Lookup Feature” on page 58 for more information on automatic lookups. This argument is optional.

WHEREMSG type=char use=OUT
This argument reports the status of the WHERE build and invocation. This argument is optional.

GET_TEXT

GET_TEXT creates an array that contains the WHERE expression text string, including the values from the **<PROMPT at run time>** window. GET_TEXT takes the following arguments:

SQLLIST type=listid use=IN
The master SCL list that is used by the WHERE expression builder.

TEXTARRAY type=char use=OUT
The text array that contains the text string for the existing WHERE expression. A text array should be defined in the caller program. The length of the array should be 200 characters.

WHEREMSG type=char use=OUT
Reports the status of the text character build. This argument is optional.

GET_TEXT_LIST

GET_TEXT_LIST creates an SCL list that contains the WHERE expression text string, including the values from the **<PROMPT at run time>** window. GET_TEXT_LIST takes the following arguments:

SQLLIST type=listid use=IN
The master list used by the WHERE expression builder.

WHRLST type=listid use=OUT
The SCL list that contains the text string for the existing WHERE expression.

WHEREMSG type=char use=OUT
Reports the status of the text character build. This argument is optional.

WHERE_CLEAN

WHERE_CLEAN removes SCL listids that are needed only for the WHERE expression builder and that are stored in SQLLIST. WHERE_CLEAN takes the following argument:

SQLLIST type=listid use=IN/OUT
The master SCL list that is used by the WHERE expression builder.

Query Invocation Class

QUERY is the query invocation class that invokes the Query window in different ways. The QUERY class invokes the SQL Query window with the following methods.

QW_EDIT

QW_EDIT invokes the SQL Query window. QW_EDIT takes the following arguments:

PROFILE type=char use=IN
The *library.catalog.entry* name of the user-defined profile that will define the parameters of the SQL Query Window session. This is an optional argument.

DATA type=char use=IN
Defines the table or tables that you want to be pre-selected for this query session. This argument is limited to 200 characters. This is an optional argument.

INCLUDE type=char use=IN
Contains the *library.catalog.entry* name of a previously saved query to include as the initial query. This is an optional argument.

ACCESS type=char use=IN
Contains the access mode for the SQL Query Window session. This is an optional argument.

NOEND type=char use=IN
Specifying the value "NOEND" turns off the confirmation window when exiting the SQL Query Window. This argument is optional.

INCLMSG type=char use=OUT
A message that reports the status of QW startup.

QW_EDIT_INCLUDE

QW_EDIT_INCLUDE invokes the SQL Query window with a previously saved query. QW_EDIT_INCLUDE takes the following arguments:

INCLUDE type=char use=IN
The *library.catalog.entry* name of a previously saved query to be included as your initial query.

NOEND type=char use=IN
Specifying the value "NOEND" turns off the confirmation window when exiting the SQL Query Window. This argument is optional.

INCLMSG type=char use=OUT
A message that reports the status of the initial include.

QW_EDIT_SELECTED

QW_EDIT_SELECTED invokes the SQL Query Window with a pre-selected table or tables. QW_EDIT_SELECTED takes the following argument:

DATA type=char use=IN
Defines the table or tables to be selected for this SQL Query Window session. This argument is limited to 200 characters.

NOEND type=char use=IN
Specifying the value "NOEND" turns off the confirmation window when exiting the SQL Query Window. This argument is optional.

QW_LIST_QUERIES

QW_LIST_QUERIES returns a list of previously saved queries to use for populating a list box. List box selections can be passed to the QW_EDIT_INCLUDE_RUN method. QW_LIST_QUERIES takes the following arguments:

LIBNAME type=char use=IN
The name of the SAS library.

CATALOG type=char use=IN
The catalog within the SAS library.

QUERYLST type=listid use=IN/OUT
Defines the SCL listid that contains the QUERY entries found in the catalog of the SAS library.

The items in QUERYLST contain the query name and query description separated by blanks.

QUERYMSG type=char use=OUT
Reports the status of QW_LIST queries. This is an optional argument.

QW_RUN

QW_RUN runs a query that has been previously saved and displays the output or passes the query to PROC REPORT. It does not invoke an SQL Query Window session. QW_RUN takes the following arguments:

INCLUDE type=char use=IN
The name of a previously saved query to be included.

RUNMODE type=char use=IN
The run query option. The values for RUNMODE are:

IMMEDIATE
The query is run and displayed in the output window.

REPORT
PROC REPORT is invoked with the results of the query.

If this query was last saved with a REPORT definition, the saved definition will be used. This argument is optional.

ACCESS type=char use=IN
The access mode for this query. Prior to Release 6.11 of the SAS System, saved queries did not include the access mode. If the saved query used an access mode other than SAS, the access mode must be supplied. An access mode value is not required for queries saved in Release 6.11 or later.

This is an optional argument.

PROFILE type=char use=IN
The name of the user-defined profile that defines the parameters of the Query. This argument is optional.

INCLMSG type=char use=OUT
A message that reports the status of QW startup. This argument is optional.

QW_EDIT_INCLUDE_RUN

QW_EDIT_INCLUDE_RUN invokes the SQL Query with a saved query and runs the query. QW_EDIT_INCLUDE_RUN takes the following arguments:

INCLUDE type=char use=IN
The name of a previously saved query to be included.

RUNMODE type=char use=IN
The run query option. This argument is optional. The values for this argument are:

IMMEDIATE
The query is run and displayed in the OUTPUT window.

REPORT
PROC REPORT is invoked with the results of the query. If this query was last saved with a REPORT definition, the saved definition is used.

This argument is optional.

ACCESS type=char use=IN
The access mode for this query. Prior to Release 6.11 of the SAS System, saved queries did not include the access mode. If the saved query used an access mode other than SAS, the access mode must be supplied. An access mode value is not required for queries saved in Release 6.11 or later. This argument is optional.

PROFILE type=char use=IN
The name of the user-defined profile that defines the parameters of the query. This argument is optional.

NOEND type=char use=IN
Specifying the value "NOEND" turns off the confirmation window when exiting the SQL Query Window. This argument is optional.

INCLMSG type=char use=OUT
A message that reports the status of QW startup. This argument is optional.

Column Expression Builder Class

QWCLEXPR is a column expression builder class that enables you to add the column expression builder in the SQL Query Window to any SAS/AF application. QWCLEXPR contains the following methods:

EDIT_ADD_EXPR

EDIT_ADD_EXPR builds an SCL list that contains columns that can be selected to build a column expression for a data set or data sets. The column expression builder is invoked, and the user can create a calculated column. After an expression is built, the expression is added to the EXPRMAST, the expression master SCL list.

If EXPRMAST argument is blank, EXPRMAST is created and the expression is added. EXPRMAST contains all sublists that are needed to modify that expression, such as available columns used, column expression type, available formats and expression attributes.

EDIT_ADD_EXPR takes the following arguments:

DSNAME type=char use=IN
The SAS data set or data sets from which the column is created. This argument is optional.

EXPRMAST type=listid use=IN/OUT
The master SCL expression list.

EXPRNAME type=char use=IN/OUT
The name of the expression. If EXPRNAME is blank, a new column expression is built and the new EXPRNAME is returned to the caller. If EXPRNAME is not blank, the column expression builder is invoked and updated or reset with the existing expression.

This argument is optional.

EXPRLABEL type=char use=IN
The label of the expression. This argument is optional.

EXPRFORMT type=char use=IN
The format of the expression. This argument is optional.

EXPRMSG type=char use=OUT
The status of the call for the expression builder. This argument is optional.

GET_TEXT

GET_TEXT creates a text array that contains the text string of an existing column expression, including format, label and expression name. GET_TEXT accepts the following arguments:

EXPRMAST type=listid use=IN
The master expression list.

TEXTARRAY type=char use=IN/OUT
The text array that contains the text string for an existing column expression.

EXPRNAME type=char use=IN
The name of an existing expression. If EXPRNAME is blank, the last expression created in the master expression list is used. This argument is optional.

GET_TEXT_LIST

GET_TEXT_LIST creates an SCL list that contains the text string of an existing column expression including format, label and expression name. GET_TEXT_LIST takes the following arguments:

EXPRMAST type=listid use=IN
The master expression list.

STRINGLST type=listid use=IN/OUT
The SCL list that contains the text string for an existing column expression.

EXPRNAME type=char use=IN
The name of an existing expression. This argument is optional.

GET_VALUE_LIST

GET_VALUE_LIST creates an SCL list with the text string of an existing column expression. The text string does not contain any of the column attributes such as label, alias, or format. If EXPRNAME is blank, the last expression created in the master expression list is used. GET_VALUE_LIST takes the following arguments:

EXPRMAST type=listid use=IN
The master expression list.

STRINGLST type=listid use=IN/OUT
The SCL list that contains the text string for an existing column expression.

EXPRNAME type=char use=IN
The name of an existing expression. This argument is optional.

SUBMIT_EXPR

SUBMIT_EXPR submits the text string of an existing column expression including format, label and expression name. If EXPRNAME is blank, the last expression created in the master expression list is used. SUBMIT_EXPR takes the following arguments:

EXPRMAST type=listid use=IN
The master expression list.

EXPRNAME type=char use=IN
The name of an existing expression. This argument is optional.

GET_ATTRIB

GET_ATTRIB extracts column expression attributes. If EXPRNAME is blank, the last expression created in the master expression list is used. GET_ATTRIB takes the following arguments:

EXPRMAST type=listid use=IN/OUT
The master expression list.

EXPRNAME type=char use=IN
The name of an existing expression. This argument is optional.

TYPE type=char use=OUT
The data type of an existing expression. This argument is optional.

FORMAT type=char use=OUT
The format of an existing expression. This argument is optional.

LABEL type=char use=OUT
The label of an existing expression. This argument is optional.

CLEAN_UP

CLEAN_UP removes the sublists of a specific expression that is stored in the expression master list. If EXPRNAME is blank, the last expression created in the master expression list is used. If EXPRNAME is _ALL_, all expressions are cleaned up. CLEAN_UP takes the following arguments:

EXPRMAST type=listid use=IN/OUT
The master expression list.

EXPRNAME type=char use=IN
The name of an existing expression. This argument is optional.

SCL Class Method Examples

This section contains examples for calling the QUERY, WHERE, and QWCLEXPR class methods from SAS Screen Control Language (SCL) along with possible returned error messages. The LOADCLASS and INSTANCE SCL functions only needed to be specified once.

QW_EDIT

The following example invokes the Query Window with a user-defined profile, a preselected query included, and the confirmation end window turned off.

```
classid    = loadclass('sashelp.sql.query.class');
objectid   = instance(classid);
querymsg   = _blank_;
call send(objectid, 'QW_EDIT', 'SASUSER.PROFILE.ORACLE ',
          ' ','SASUSER.PROFILE.SALES',' ','noend',querymsg);
if ( querymsg not = _blank_ ) then
     _msg_      = querymsg;
```

Error Messages

Query included does not exist
An included query does not exist.

Profile selected does not exist
The profile does not exist.

QW_EDIT_INCLUDE

The following example invokes the Query Window with the query SASUSER.PROFILE.SALES included.

```
classid    = loadclass('sashelp.sql.query.class');
objectid   = instance(classid);
querymsg   = _blank_;
call send(objectid, 'QW_EDIT_INCLUDE', 'SASUSER.PROFILE.SALES',
          ' ',querymsg);
if ( querymsg not = _blank_ ) then
   _msg_      = querymsg;
```

Error Messages

Query included does not exist
An included query does not exist.

INCLUDE parameter is missing
QW_EDIT_INCLUDE has been passed a blank INCLUDE parameter.

QW_EDIT_SELECTED

The following example invokes the SQL Query Window with two tables, SASUSER.FITNESS and SASUSER.CLASS, selected, and begins in the SQL Query Column window.

```
clasid     = loadclass('sashelp.sql.query.class');
objectid   = instance(clasid);
call send(objectid, 'QW_EDIT_SELECTED', 'SASUSER.FITNESS, SASUSER.CLASS');
```

QW_LIST_QUERIES

The following example searches the SQL DICTIONARY.CATALOG rows for the SASUSER library and PROFILE catalog to find all catalog types of QUERY (saved SQL Query window queries). The name and description of the saved queries are stored in the SCL list QUERYLST with blanks between the name and the description.

```
classid    = loadclass('sashelp.sql.query.class');
objectid   = instance(classid);
querymsg   = _blank_;
querylst   = makelist();
call send(objectid, 'QW_LIST_QUERIES', 'SASUSER','PROFILE',
          querylst, querymsg);
if ( querymsg not = _blank_ ) then
   _msg_      = querymsg;
```

Error Messages

LIBNAME parameter is missing
QW_LIST_QUERIES was passed a blank LIBNAME parameter.

CATALOG parameter is missing
QW_LIST_QUERIES was passed a blank CATALOG parameter.

Libname does not exist
The library specified in the LIBNAME parameter does not exist.

Catalog does not exist
The catalog specified in the CATALOG parameter does not exist.

No queries were found
No QUERY entries were found in the specified library and catalog.

QW_RUN

The following example runs the query stored in SASUSER.PROFILE.SALES and displays results in the SAS Display Manager Output window without invoking the Query window.

```
classid    = loadclass('sashelp.sql.query.class');
objectid   = instance(classid);
querymsg   = _blank_;
call send(objectid, 'QW_RUN', 'SASUSER.PROFILE.SALES',
         'IMMEDIATE',' ',' ',querymsg);
if ( querymsg not = _blank_ ) then
   _msg_      = querymsg;
```

The following example runs the query stored in SASUSER.PROFILE.SALES and displays results in the REPORT procedure window without invoking the Query window.

```
classid    = loadclass('sashelp.sql.query.class');
objectid   = instance(classid);
querymsg   = _blank_;
call send(objectid, 'QW_RUN', 'SASUSER.PROFILE.SALES',
         ' ',' ',' ',querymsg));
if ( querymsg not = _blank_ ) then
   _msg_      = querymsg;
```

Error Messages

Query included does not exist
An included query does not exist.

Profile selected does not exist
The profile does not exist.

INCLUDE parameter is missing
QW_RUN was passed a blank INCLUDE parameter.

Signon canceled
The user canceled out of a DBMS signon window.

QW_EDIT_INCLUDE_RUN

The following example runs the query stored in SASUSER.PROFILE.SALES, displays results in the REPORT procedure window, and invokes the Query window with this query included.

```
classid    = loadclass('sashelp.sql.query.class');
objectid   = instance(classid);
querymsg   = _blank_;
call send(objectid, 'QW_EDIT_INCLUDE_RUN', 'SASUSER.PROFILE.SALES');
if ( querymsg not = _blank_ ) then
   _msg_      = querymsg;
```

Error Messages

Query included does not exist
An included query does not exist.

Profile selected does not exist
The profile does not exist.

INCLUDE parameter is missing
QW_RUN was passed a blank INCLUDE parameter.

WHERE_BLD

The following example builds SCL lists that contain available columns and column information from the data set SASUSER.CRIME. The example invokes the WHERE expression window titled **Record Subset**. This is independent of the SQL Query Window.

```
classid    = loadclass('sashelp.sql.qwwhere.class');
objectid   = instance(classid);
wherelst   = makelist();
sqllist    = makelist();
wheremsg   = _blank_;
call send(objectid, 'WHERE_BLD',  sqllist, 'SASUSER.CRIME',
          wherelst, 'Record Subset', ' ', ' ', ' ', ' ', wheremsg);
if ( wheremsg not = _blank_ ) then
   _msg_      = wheremsg;
```

The following example invokes the WHERE expression window and enables the user to edit the last WHERE expression created and stored in SQLLIST. The WHERE expression window is titled **Edit Record Subset**. This is independent of the SQL Query Window.

```
classid    = loadclass('sashelp.sql.qwwhere.class');
objectid   = instance(classid);
wheremsg   = _blank_;
call send(objectid, 'WHERE_BLD',  sqllist, ' ',
          wherelst, 'Edit Record Subset', ' ', ' ', ' ', ' ', wheremsg);
if ( wheremsg not = _blank_ ) then
   _msg_      = wheremsg;
```

Error Messages

Dataset does not exist
The data set specified in the data set name parameter is not found while building the available columns.

SCL list is empty or missing
The available column SCL lists are empty.

Automatic Lookup Dataset does not exist
A data set name was passed as the AUTOLOOK parameter and the data set is not found.

GET_TEXT

The following example creates a text array that contains an existing WHERE expression character string. The array can extend up to fifty 200-character-length segments. If any prompts were included in the WHERE expression, they are resolved by prompting the user to supply the values.

```
array textarray(50) $200;
classid     = loadclass('sashelp.sql.qwwhere.class');
objectid    = instance(classid);
wheremsg   = _blank_;
call send(objectid, 'GET_TEXT', sqllist, textarray, wheremsg);
if ( wheremsg not = _blank_ ) then
   _msg_      = wheremsg;
```

Error Message

SCL list is empty or missing
The SQLLIST is empty.

GET_TEXT_LIST

The following example builds an SCL list that contains the character string of an existing WHERE expression. If any prompts were included in the WHERE expression, they are resolved by prompting the user to supply the values.

```
classid      = loadclass('sashelp.sql.qwwhere.class');
objectid     = instance(classid);
stringlst   = makelist();
wheremsg   = _blank_;
call send(objectid, 'GET_TEXT_LIST', sqllist, stringlst);
if ( wheremsg not = _blank_ ) then
   _msg_      = wheremsg;
```

Error Message

SCL list is empty or missing
The SQLLIST or STRINGLST is empty.

WHERE_CLEAN

The following example removes any SCL lists that are created and stored in SQLLIST for the WHERE expression window.

```
classid    = loadclass('sashelp.sql.qwwhere.class');
objectid   = instance(classid);
call send(objectid, 'WHERE_CLEAN', sqllist);
```

EDIT_ADD_EXPR

The following example invokes the COLUMN expression window to build the expression that will be named PRICEFT using the available columns from the data set SASUSER.HOUSES. PRICEFT will be added to EXPRMAST (expression master SCL list). This is independent of the SQL Query Window.

```
classid    = loadclass('sashelp.sql.qwclexpr.class');
objectid   = instance(classid);
exprmast   = makelist();
exprmsg    = _blank_;
call send(objectid, 'EDIT_ADD_EXPR', 'SASUSER.HOUSES', exprmast,
          'PRICEFT',' ',' ',exprmsg);
if (  exprmsg  not = _blank_ ) then
    _msg_      =  exprmsg;
```

The following example invokes the COLUMN expression window and allows the user to edit the expression named PRICEFT that was created and stored in EXPRMAST. This is independent of the SQL Query Window.

```
classid    = loadclass('sashelp.sql.qwclexpr.class');
objectid   = instance(classid);
exprmast   = makelist();
exprmsg    = _blank_;
call send(objectid, 'EDIT_ADD_EXPR', ' ', exprmast,
          'PRICEFT',' ',' ',exprmsg);
if (  exprmsg  not = _blank_ ) then
    _msg_      =  exprmsg;
```

Error Messages

Dataset does not exist
The data set specified in the data set name parameter is not found while building the available columns.

SCL list is empty or missing
The user attempted to edit a COLUMN expression and EXPRMAST is empty. This message is also generated if the available column SCL lists are empty.

GET_TEXT

The following example creates a text array that contains the text string of the COLUMN expression named PRICEFT including format, label, and expression name. The array can extend up to fifty 200 character length segments.

```
array textarray(50) $200;
classid    = loadclass('sashelp.sql.qwclexpr.class');
objectid   = instance(classid);
call send(objectid, 'GET_TEXT', exprmast, textarray,'PRICEFT');
```

GET_TEXT_LIST

The following example creates an SCL list that contains the text string of the COLUMN expression named PRICEFT including format, label and expression name.

```
classid    = loadclass('sashelp.sql.qwclexpr.class');
objectid   = instance(classid);
stringlst  = makelist();
call send(objectid, 'GET_TEXT_LIST', exprmast, stringlst,'PRICEFT');
```

GET_ATTRIB

The following example extracts the attributes for the COLUMN expression named PRICEFT: format, data type, label and expression name.

```
classid    = loadclass('sashelp.sql.qwclexpr.class');
objectid   = instance(classid);
type       = _blank_;
format     = _blank_;
label      = _blank_;
call send(objectid, 'GET_ATTRIB', exprmast, 'PRICEFT', type,
          format, label);
```

GET_VALUE_LIST

The following example creates an SCL list that contains only the definition for the COLUMN expression named PRICEFT. The text string does not contain any of these column attributes: format, label and expression name.

```
classid    = loadclass('sashelp.sql.qwclexpr.class');
objectid   = instance(classid);
stringlst  = makelist();
call send(objectid, 'GET_VALUE_LIST', exprmast, stringlist,
          'PRICEFT');
```

CLEAN_UP

The following example removes the sublists created for the COLUMN expression named PRICEFT that were stored in EXPRMAST (expression master list).

```
classid    = loadclass('sashelp.sql.qwclexpr.class');
objectid   = instance(classid);
call send(objectid, 'CLEAN_UP', exprmast, 'PRICEFT');
```

The following example removes all COLUMN expression sublists that were stored in EXPRMAST (expression master list).

```
classid    = loadclass('sashelp.sql.qwclexpr.class');
objectid   = instance(classid);
call send(objectid, 'CLEAN_UP', exprmast, '_ALL_');
```

The correct bibliographic citation for this manual is as follows: SAS Institute Inc., *SAS SQL Query Window User's Guide, Version 8*, Cary, NC: SAS Institute Inc., 1999. pp. 120.

SAS SQL Query Window User's Guide, Version 8

ISBN 1–58025–554–X

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

1st printing, October 1999