

## SQL Query Window Classes: How To Incorporate the SQL Where Expression Builder Class In Custom SAS/AF® Applications

Christopher A. Roper, Qualex Consulting Services, Inc. Fairfax, Va.

### Abstract

SQL (Structured Query Language) is the standard language used for querying all major databases, including SYBASE, ORACLE, RED BRICK, DB2, and others. The SAS Query window creates SQL expressions using SAS/AF Classes that are available to the SAS/AF developer. One of these class entries is the Where Clause Builder Class, QWWHERE.CLASS in the SASHELP.SQL catalog. This is the class used by the Query window to build SQL where clauses. This class may be called by any SAS/AF entry to present the same interface the SAS Query window uses to build where clauses for SQL queries. This paper will cover what the Where Clause Builder Class is, how to incorporate it into a SAS/AF application, and the nuances of using this class effectively.

### Introduction

The SQL Where Clause Expression Builder class is essentially a frame called by the SAS Query window when a user has selected the "Where Conditions for Subset" option under the Locals pmenu. Many users of the SAS System have used this option in the Query window, and are very comfortable with it. It has a clear, simple interface that is easy to use and well thought out. Little is to be gained by taking the time to develop a different SQL where clause builder for a custom AF application. This makes it an obvious choice for the AF developer to include in a custom application. For the remainder of this paper, this class will be referenced by the name QWWHERE.CLASS.

### Default Settings Used For Examples

The examples shown in this paper will use the following defaults. The data set used is SASHELP.RETAIL, available to anyone licensing Base SAS. All other settings and options will be left at the default settings. Note that the SCL list id values are set upon execution of the methods and code, and may differ from those shown in the following examples and figures in this paper.

### Methods of the SQL Where Clause Expression Builder Class (QWWHERE.CLASS)

The SQL Where Clause Expression Builder class (QWWHERE.CLASS) provides four methods: WHERE\_BLD, GET\_TEXT, GET\_TEXT\_LIST, and WHERE\_CLEAN. These methods provide access to the lists that control the behavior of this class. A brief description of each of these methods follows, plus the full descriptions provided by the SAS online help facility are available by selecting the Index tab from the Help Topics: SAS System Help window, and typing QWWHERE.

The WHERE\_BLD method makes an SCL list of all the columns available for subsetting. It also opens the frame interface, or window, used by the end user to build the where clause. See the SAS Online Help documentation for a brief description of this method and the method arguments.

The GET\_TEXT method creates a text string that is the where clause built in the WHERE window. This text string is limited to 200 characters. See the SAS Online Help documentation for a brief description of this method and the method arguments.

The GET\_TEXT\_LIST method is very similar to the GET\_TEXT method, except that the results are returned to an SCL list, and thus the 200 character limit does not apply. Both the GET\_TEXT and GET\_TEXT\_LIST methods optionally return a text string reporting the status of the text string built in the WHERE window. See the SAS Online Help documentation for a brief description of this method and the method arguments.

The WHERE\_CLEAN method performs the housekeeping chores for the QWWHERE.CLASS. It will recursively delete the SCL lists used by the QWWHERE.CLASS, thus freeing up system resources used by the execution of this class. See the SAS Online Help documentation for a brief description of this method and the method arguments.

### Using the QWWHERE.CLASS

The QWWHERE.CLASS can be made available to an application by loading the class, and then instantiating it. These two steps are often nested into one SCL statement in the following manner:

```
QW_ID =  
instance(loadclass('SASHELP.SQL.QWWHERE.CLASS'));
```

Where QW\_ID is the identifier of this instance of the QWWHERE.CLASS. Now, a CALL SEND statement may reference this identifier to execute any of the four methods available to this class.

### Using The WHERE\_BLD Method

The WHERE\_BLD method performs two functions. It builds a list of the columns available to the Where Clause Builder, plus it displays the Where Clause Builder window. This is the interface the user will use to create an SQL where clause. The WHERE\_BLD method accepts nine arguments, and the syntax is:

```
CALL SEND  
(QW_ID,'WHERE_BLD','SQLLIST',dsname,wherels  
t,whitle,avtitle,'NOPROMPT','NOLOOKUP',  
'AUTOLOOK',wheremsg);
```

The key to effective use of the WHERE\_BLD method lies in understanding how to use the first two of these arguments, 'SQLLIST' and dsname.

SQLLIST is the master list for the QWWHERE.CLASS. It contains the names and list id's for the eighteen lists and one numeric item used by the QWWHERE.CLASS. Of these nineteen items, only the first six are of interest to the application developer. The other thirteen are used only during the execution of the class, and do not contain any information after the class has executed. These first six lists are: COLWHLST, CINFOLST, LBMEMLST, COLLIST, WHERELST, and WHDSPLST. See Figure 1 for an example of the master list SQLLIST.

COLWHLST contains the available columns for the QWWHERE.CLASS. Each column is an item in the list and the column label is used as the name of the item. If no label exists for the column, then the column name is used. See Figure 2 for an example.

CINFOLST tracks the column type for the QWWHERE.CLASS. Character data will be noted as 'C' and numeric will be noted as 'N'. See Figure 3 for an example. This list is used by the QWWHERE.CLASS to determine if invalid operations are requested (i.e. comparing a numeric column to a character column), if constant text entered by the user should be quoted (numeric columns will not have quoted text), and whether to present a numeric or character 'Enter Values Screen' to the user (See Figures 4 & 5).

LBMEMLST keeps track of the data sets contributing the columns to the QWWHERE.CLASS. The format of this list is TABLE=LIBREF.TABLE (See Figure 6).

COLLIST contains the labels of the columns available to the QWWHERE.CLASS. The values from this list are displayed in the WHERE window as they are selected by the user. The format of this list is TABLE=COLUMN LABEL (See Figure 7).

WHERELST contains the components of the where clause created by the end user. Each component of the clause is an item in the list, and these items are named according to the function they serve ( See Figure 8).

WHDSPLST is a list identical to the WHERELST list, except that it uses column labels instead of column names where appropriate. This list is used by the QWWHERE.CLASS when an existing where clause is loaded into the QWWHERE.CLASS (See Figure 9)

### Saving a Where Clause

Since the QWWHERE.CLASS uses SCL lists to manipulate and display where clauses in the Where Clause window, a where clause built by the QWWHERE.CLASS may be saved just like any other SCL list. In the case of the where clause, the master list and all sublists are saved to the same directory structure to simplify retrieval using the SCL SAVELIST function. In practice, a scheme must be devised to create unique directory structures for each saved where clause, so that previously saved where clauses are not over written. There are any number of valid naming schemes, and it is up to the developer to determine one that will work in any given situation. (See Example 1: SCL, section SAVE: ).

**Loading an Existing Where Clause**

Loading the QWWHERE.CLASS with existing lists built and/or modified in a previous session is simply the opposite of the save technique. By using the SCL function FILLIST, the master list and all the sublists can be recreated. Then they may be passed to the QWWHERE.CLASS instead of data sets, so that they can be reviewed or modified. This allows the end user to proceed with the where clause construction just as if the end user had created it from scratch. This can be especially useful for building a library of standard queries that can be modified by an end user to meet his/her specific needs at a later date.

In practice, an interface is needed by the end user to select from the library of stored queries. The discussion of developing this interface is beyond the scope of this paper.

Note: if a data set name were passed along with the master list, the QWWHERE.CLASS would overwrite the contents of the master list and all the sublists with information from the data set. (See Example 1: SCL, section LOAD: )

**Viewing the Results of the Where Clause Builder**

When the SQL statements are created in a SUBMIT block of SCL code, they are stored in the SAS Preview buffer. This buffer is available to be viewed until the buffer is cleared by either a direct CLEAR command, or by sending the code to the SAS supervisor through a SUBMIT CONTINUE block. Therefore, the SQL statements and where clause may be viewed through the use of the SCL Preview command, which grants access to the SAS preview buffer. (See Example 1: SCL, section VIEW: )

**A Simple Example**

Example 1 is a simple example of the use of the QWWHERE.CLASS in a Frame entry. The Frame entry has four command push buttons, New Where, Save Where, Load Where, and View Where. The SCL entry has the labeled sections NEW, SAVE, LOAD, and VIEW respectively. This example is far too simple to be robust enough for a practical application, it is intended to simply show the basics of how to use the QWWHERE.CLASS.

**Conclusion**

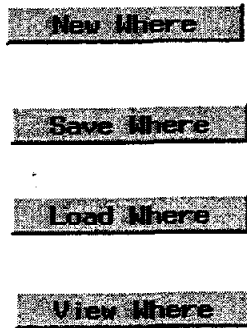
SAS Institute has spend a great deal of effort in creating very elegant object classes to support it's various modules.

SASHELP.SQL.QWWHERE.CLASS is just one of these classes, and it is a great advantage for the SAS developer to be able to use this class. Using this existing class greatly reduces the amount of time necessary to develop an application needing this functionality. Testing is much simpler since SAS has already performed a great deal of testing on this class. And finally, the end users have likely been using this class for some time and thus end user acceptance of the application is made much easier since they are already familiar with this class.

**Author Contact Information**

Christopher A. Roper  
 Qualex Consulting Services, Inc.  
 Rt. 2 Box 224C  
 Hillsville, Va. 24343  
 Chris.Roper@QLX.COM  
<http://www.qlx.com>

### Example 1: Frame



### Example 1: SCL

```
INIT:
  /* Prevent compile time warnings */
  rc = rc;

  /* Load and instantiate the WHERE
  CLAUSE BUILDER class */
  qw_id = instance(loadclass(
    'SASHELP.SQL.QWWHERE.CLASS'));

  /* Create an empty dummy data set with
  no variables */
  dsid = open('WORK.DUMMY','N');
  rc = close(dsid);

  /* Create the master list used by the
  QWWHERE.CLASS */
  sqllist = makelist();

  /* Save indentation of PROC SQL step in
  Preview window */
  control asis;

  return; /* INIT */

TERM:

  /* Housekeeping */
  Call send(qw_id,'WHERE_CLEAN',sqllist);

  Return;

LOAD:
```

```
/* Start with a clean list, prevents
errors */
rc = clearlist(sqllist,'Y');

/* Repopulate the master list SQLLIST */
rc =
filllist('CATALOG','Q.QWWHERE.QUERY.SLIST',sqllist);

/* Repopulate the sublists */
do i = 1 to listlen(sqllist);

  sublist = nameitem(sqllist,i);
  sublistid = makelist();

  target =
compress('Q.QWWHERE.'!!sublist!!'.SLIST'
);

  if sublist ne 'WHPROMPTNO' then
    rc =
filllist('CATALOG',target,sublistid);

    rc = setitemn(sqllist,sublistid,i);

end; /* do i = 1 to listlen(sqllist); */

/* Pass the rebuilt lists to the Where
Clause Builder */
call
send(qw_id,'WHERE_BLD',sqllist,'');

return; /* LOAD */

SAVE:
  /* Save the master list to a SAS
  catalog */
  rc = savelist(
    'CATALOG','Q.QWWHERE.QUERY.SLIST',sqllis
t);

  /* Save the sublists to a SAS catalog */
  do i = 1 to listlen(sqllist);

    sublist = nameitem(sqllist,i);
    sublistid = getitemn(sqllist,i);

    target = compress(
      'Q.QWWHERE.'!!sublist!!'.SLIST');
```

```

        if sublist ne 'WHPROMPTNO' then
            rc = savelist(
                'CATALOG',target,sublistid);

        end; /* do i = 1 to listlen(sqllist); */

return; /* SAVE */

NEW:

    /* Start with a clean list, prevents
    errors */
    rc = clearlist(sqllist,'Y');

    /* Invoke the Where Clause Builder */
    /* WORK.DUMMY will force the
    QWWHERE.CLASS to use two level names.
    This */
    /* is a work around to prevent
    character columns being treated as
    numeric*/
    call
    send(qw_id,'WHERE_BLD',sqllist,'SASHELP.
    RETAIL WORK.DUMMY');

return; /* NEW */

VIEW:

    /* Build the SQL in a submit block */
    submit;

    proc sql;

        create table NEW as
        select *
        from SASHELP.RETAIL
        where
    endsubmit;

    /* Create a list to contain the Where
    Clause */
    wherelst = makelist();

    /* Retrieve the list id of the sublist
    containing the */
    /* components of the where clause from
    the master list */
    whlst = getnitemn(sqllist,'WHEREELST');

```

```

    /* Copy that into the current list */
    wherelst =
    copylist(whlst,'N',wherelst);

    /* The where clause must be parsed from
    the sublist WHEREELST */

    do i = 1 to listlen(wherelst);

        whseg = getitemc(wherelst,i);

        submit;
            &whseg
        endsubmit;

    end; /* do i = 1 to listlen(wherelst); */
    /* The where clause must be parsed from
    the sublist WHEREELST */

    /* Complete the Proc SQL step */
    submit;
        ;
    quit;
    endsubmit;

    /* Invoke the preview window to see the
    SQL statements */
    rc = preview('BROWSE','Where Clause
    Builder Expression Preview');

    /* Clear the preview window for the
    next use (Commented out) */
    /* OR */
    /* Send the code to the SAS Supervisor
    (Active) */

* rc = preview('CLEAR');

    submit continue;
        /* Demo of QWWHERE.CLASS */ ;
    endsubmit;

return;

```

Figure 1: The master list SQLLIST

```
( COLWHLST=1957
  CINFOLST=1967
  LBMEMLST=1959
  COLLIST=1961
  WHERELST=2007
  WHDSPLST=2009
  WHCURLST=1973
  PROMPLST=2139
  PRSTRLST=2141
  EXNAMLST=2143
  EXASLIST=2145
  EXFMLST=2147
  EXWHLST=2149
  EXMEMLST=2151
  COEXLST=2153
  LKCOLLST=2155
  SLTABLST=2157
  FMLST=2159
  WHPROMPTNO=0
)[1947]
```

Figure 2: COLWHLST

```
( RETAIL.RETAIL SALES IN MILLIONS OF
$='RETAIL.SALES'
RETAIL.DATE='RETAIL.DATE'
RETAIL.YEAR='RETAIL.YEAR'
RETAIL.MONTH='RETAIL.MONTH'
RETAIL.DAY='RETAIL.DAY'
)[1957]
```

Figure 3: CINFOLST

```
( RETAIL.SALES='N'
RETAIL.DATE='N'
RETAIL.YEAR='N'
RETAIL.MONTH='N'
RETAIL.DAY='N'
)[1967]
```

Figure 4: Numeric "Enter Values Screen"

Figure 5: Character "Enter Values Screen"

Figure 6: LBMEMLST

```
( DUMMY='WORK.DUMMY'
RETAIL='SASHELP.RETAIL'
)[2047]
```

Figure 7: COLLIST

```
(RETAIL='RETAIL.Retail sales in millions of $'
RETAIL='RETAIL.DATE'
RETAIL='RETAIL.YEAR'
RETAIL='RETAIL.MONTH'
RETAIL='RETAIL.DAY'
)[2057]
```

Figure 8: WHERELST

```
( OPERAND='RETAIL.SALES'

COMP='GT'
OPERAND='5'
)[2039]
```

Figure 9: WHDSPLS

```
( OPERAND='RETAIL.Retail sales in millions of $'
COMP='GT'
OPERAND='5'
)[2077]
```