Leveraging Common Functions

ILE has benefits to offer both RPG and SQL allowing both languages to share common functions. Service programs can be created from procedures which may be used in both SQL and RPG. Re-usable code leverages existing processes to speed future application development. A simple example is a procedure that performs a simple conversion on a column/field from one format to another format.

```
Dcl-Proc Convert_Date_to_Dec  Export;
   Dcl-PI Convert_Date_to_Dec zoned(8:0);
      in_Date      date       Const;
      in_Format  varChar(5) Const;
   end-PI;
```

The code above describes a procedure in a service program. It was created to accept a date-type variable as input as well as a date format (*ISO, USA, etc.) which will be used to format the decimal value that the procedure returns. (Yes, there is a BIF to perform this, but the procedure contains different rules and validation regarding what is passed back. For example, the procedure may construct a 6-digit date format that is outside the IBM's standard date window.)

The same function can be made available for use in SQL by creating a User Defined Function (UDF). UDF's are not limited to SQL code (though they may be created with SQL). A service program procedure written in RPG can quite easily become an SQL UDF. All that needs to be done in this case is to register the service program procedure as a function. Create Function supplies the necessary argument and defines the parameter style for the new function. (A row will be entered in QSYS2.SYSROUTINE)

```
CREATE FUNCTION CONVERT_DATE_TO_DEC (
  in_date DATE, in_from_format VarChar(5) )
RETURNS NUMERIC(8,0)
LANGUAGE rpgle
DETERMINISTIC
NO SQL
EXTERNAL NAME 'TTCODA/DATE_SP(CONVERT_DATE_TO_DEC)'
PARAMETER STYLE General;
```

An RPG program (bound to the DATE_SP service program, where the procedure is defined) can access the procedure to convert a date into a zoned decimal value (ISO format) in a manner consistent with the date rules coded. The RPGLE code looks exactly as expected, using the EVAL to assign the numeric date variable to the specified date format.

```
today = %date();
dateR1 = Convert_Date_To_Dec( Today      : '*ISO' );
yesterday = today - %days(1);
dateR3 = Convert_Date_To_Dec( yesterday : '*MDY' )  ;
```

Since a User Defined Function (UDF) was constructed to invoke the same procedure, an SQL statement may be issued—in a SET or SELECT operation, for example, to access the same procedure. The variable returned is consistent with the same format and rules applied to the request in RPG code.

```
tomorrow = today + %days(1);

Exec SQL
  set :Date2 = Convert_Date_to_Dec( :tomorrow , :ISOVAR );

SELECT * FROM oas_docHead WHERE CONVERT_DATE_TO_DEC(docDate,'*USA') = 10032014
```

The following example illustrates the use of a service program procedure to supply text in place of an integer code (for a display, or a report). In RPG the bound procedure accepts the argument list and after performing some table look-ups to find the area and element associated with the code, returns the text definition of the code.

```
Dcl-PR getCODAdefinition varchar(200) ;
    p_Area        varChar(15)  CONST    ;
    p_Name        varChar(25)  CONST    ;
    p_ID            int(10)    CONST    ;
End-PR                                  ;

elementText = getCodaDefinition(areaName:elementName:elementInt);
```

The same translation process can be defined as a UDF, allowing SQL to provide the same conversion from the integer code to a text definition.

```
CREATE FUNCTION TTCODA/CVTTO_TEXT
(
  areaName       varChar(15),
  elementName    varChar(25),
  codeValue      Int
)

RETURNS VARCHAR(200)
LANGUAGE RPGLE
DETERMINISTIC
MODIFIES SQL DATA
EXTERNAL NAME 'TTCODA/XRF000SV(GETCODADEFINITION)'
PARAMETER STYLE General;


        Exec SQL
          select CVTTO_TEXT(:areaName,:elementName, :elementInt)
            into :elementText2
            from sysibm/sysdummy1 ;
```

The examples are very simplistic, but illustrate an easy method of sharing common functions between SQL and RPG via service programs. Check out IBM's Knowledge Center for more information

http://www-01.ibm.com/support/knowledgecenter/ssw_ibm_i_71/sqlp/rbafykickoff.htm