```
**FREE
CTL-OPT
    DATFMT(*ISO)
    OPTION(*SRCSTMT : *NODEBUGIO)
    bnddir('QC2LE')
    ALWNULL(*USRCTL)
    EXTBININT(*yes);

    //* This program is offered as is and may be freely copied and modified to fit your needs.
    //* As was discussed during the Gateway/400 presentation, this program should be considered
    //* a work in process, and not a finished product. No guarantee is given or implied.
    //* Please test this program before using it in your environment.

    //*    PROGRAM NAME - SQLtoIFS
    //*    PROGRAMMER   - Modification of an example program found on an IBM site
    //*    DATE         - 02/21/2018
    //*    PURPOSE      - Reads five parameters to run an SQL and write the output to the IFS.
    //*                   This program can be called directly from the CALL statement, but it is
    //*                   easier to call from its stored procedure spSQLtoIFS.
    //*
    //*    PARAMETERS   -
    //*      1. Full IFS location of the output file, including the file name.
    ///          Example: /home/Testdescriptor/testDateStuff.csv
    //*      2. Full SQL statement that you want to execute.
    //*         Example: (in one contiguous string)
    //*                  SELECT CustNo, CAST(SUM(CurBal) AS DEC(11, 2)) AS Current_Balance +
    //*                  FROM   OrderMast +
    //*                  WHERE  status NOT IN (''D'', ''X'') +
    //*                  GROUP BY CustNo +
    //*                  ORDER BY CustNo
    //*      3. Include_Header_YN Values Y and N, defaults to Y.
    //*      4. String_Separator defaults to inch mark (").
    //*      5. Column_Separator defaults to comma (,).


dcl-pr MAIN extPgm('SQLTOIFS');
  Complete_IFS_File_Name         varchar(80);
  SQL_Statement_for_Extract_File varchar(4096);
  Include_Header_YN              char(1) OPTIONS(*NOPASS);
  String_Separator               char(1) OPTIONS(*NOPASS);
  Column_Separator               char(1) OPTIONS(*NOPASS);
end-Pr;

dcl-pi MAIN   ;
  FileName      varchar(80);
  SQLStatement  varchar(4096);
  pInclude_Header char(1) OPTIONS(*NOPASS);
  pString_Separator char(1) OPTIONS(*NOPASS);
  pColumn_Separator char(1) OPTIONS(*NOPASS);
end-Pi;

////////////////////////////////////////////////////
// Used to compare DESCRIPTOR fields
////////////////////////////////////////////////////
dcl-s i                int(10);
dcl-s iCount           int(10);
dcl-s iData_Char       char(  1024);
dcl-s iData_Date_08    date;
dcl-s iData_Time_08    time;
dcl-s iData_Stamp      timestamp;
dcl-s iData_VarChar    varchar( 1024);
```

```
dcl-s iData_Integer    int(10);
dcl-s iDateTimeType    int(10);
dcl-s iLabel           varchar(60);
dcl-s iLength          int(10);
dcl-s iName            varchar(128);
dcl-s iPrecision       int(10);
dcl-s iResult_ind      int(10);
dcl-s iRow             int(10) inz(0);
dcl-s iScale           int(10);
dcl-s iType            int(10);
dcl-s iColumn_Name     varchar(128);


////////////////////////////////////////////////////
// DESCRIPTOR - RETURNED COLUMN TYPE
////////////////////////////////////////////////////
dcl-s tCharacter       int(10) inz( 1);
dcl-s iDate_time_Stamp int(10) inz( 9);
dcl-s tDecimal         int(10) inz( 3);
dcl-s tInteger         int(10) inz( 4);
dcl-s tVarChar         int(10) inz(12);

////////////////////////////////////////////////////
// DESCRIPTOR - SUB TYPES WHEN TYPE IS 9 (DATE/TIME/TIMESTAMP)
////////////////////////////////////////////////////
dcl-s iDateType        int(10) inz( 1);
dcl-s iTimeType        int(10) inz( 2);
dcl-s iTimeStampType   int( 3) inz( 3);

////////////////////////////////////////////////////
// Miscellaneous GLOBAL variables
////////////////////////////////////////////////////
dcl-s dString          varchar(4096);
dcl-s returned_sqlcode int(10);
dcl-s token            varchar(515);
dcl-s var1             int(10);
dcl-s Include_Header   char(1);
dcl-s String_Separator char(1);
dcl-s Column_Separator char(1);

// Maximum # fields in a single file on DATEDW is 1474.
//dcl-ds DS1 DIM(1500) QUALIFIED;
dcl-ds DS1 DIM(1500) QUALIFIED;
  Label               like(iLabel);
  Name                like(iName);
  Data                varchar(1024);
end-Ds;

// ---------------------------------------
// Delete data from prior running of this job.
dcl-s eola      char(2)  inz(X'0d25');
dcl-s vfilename  char(80);
dcl-s oFlag     int(10);
dcl-s oMode     uns(10);
dcl-s outRec    char(1024);
//dcl-s qut       char(1)  inz('"');
dcl-c null      x'00';
dcl-s rc        int(10);
dcl-s buflen    int(10) inz(0);
dcl-s str       int(10);
dcl-s cmd       varchar(1024);
```

```
dcl-s codepage  uns(10)  inz(819);
dcl-s err_flag  int(10);


// ---------------------------------------
// DATA AREA JHAPAR
//dcl-ds File_JHAPar EXTNAME('JHAPAR') end-ds;
//dcl-ds JHAPar DtaAra('JHAPAR') likeds(File_JHAPar);

dcl-pr system extproc(*dclcase);
  envvar pointer value options(*string);
end-pr;

       /copy srccus/qrpglesrc,ifscopy

////////////////////////////////////////////////////
// MAIN PROCEDURE START
//   (BEGINNING OF CALCULATIONS)
////////////////////////////////////////////////////

EXEC SQL
  SET OPTION COMMIT = *NONE,
             CLOSQLCSR = *ENDMOD ;

// Set the default values for the optional parameters.
// As this is now required to be called from the stored procedure, the default
// logic has been moved from here to the procedure.
//if %parms = 2;
//  Include_Header = 'Y';
//  String_Separator = '"';
//else;
Include_Header   = pInclude_Header;
String_Separator = pString_Separator;
Column_Separator = pColumn_Separator;
//endIf;

//if %parms = 3;
//  String_Separator = '"';
//else;
//  String_Separator = pString_Separator;
//endIf;

//if Include_Header = *blank;
//  Include_Header = 'Y';
//endIf;

vFileName = %trim(FileName);


del_file();
crt_File();
GetFileData();
EXEC SQL
  CLOSE c1;

// There are several other descriptor items that you might need to check to determine how to
handle
//  the result data. PRECISION, SCALE, DB2_CCSID, and DATETIME_INTERVAL_CODE are among them. The
//  host variable that has the DATA value read into it must have the same data type and CCSID as
//  the data being read. If the data type is varying length, the host variable can be declared
//  longer than the actual data. For all other data types, the length must match exactly.
```

```
// NAME, DB2_SYSTEM_COLUMN_NAME, and DB2_LABEL can be used to get name-related values for the
result
//  column. See GET DESCRIPTOR for more information about the items returned for a GET DESCRIPTOR
//    https://www.ibm.com/support/knowledgecenter/ssw_ibm_i_72/db2/rbafzgetdescr.htm?view=kc
//  statement and for the definition of the TYPE values.

*inlr = *on;

////////////////////////////////////////
// -----------------------------------------------
// Procedure name:  GetFileData
// Purpose:         Creates the descriptor and loops the data
// Returns:
// Author:          John Derr
// Created:         02/14/2018
//---------------------
DCL-PROC GetFileData  ;

DCL-PI *N   ;
END-PI ;

dString = %trim(SQLStatement);

// The statement is assigned to a host variable. The host variable, in this case named DSTRING,
//  is then processed by using the PREPARE statement as shown:
EXEC SQL
  PREPARE s1 FROM :dstring;


// Next, you need to determine the number of result columns and their data types. To do this,
//  you need to allocate the largest number of entries for an SQL descriptor that you think
//  you will need. Assume that no more than 20 columns are ever expected to be accessed by a
//  single SELECT statement.
EXEC SQL
  ALLOCATE DESCRIPTOR 'myDescriptor' WITH MAX 20;


// Now that the descriptor is allocated, the DESCRIBE statement can be done to get the
//  column information.
EXEC SQL
  DESCRIBE s1 USING DESCRIPTOR 'myDescriptor';

// When the DESCRIBE statement is run, SQL places values that provide information about the
//  statement's select-list into the SQL descriptor area defined by 'myDescriptor'.

// If the DESCRIBE determines that NOT ENOUGH ENTRIES were allocated in the descriptor,
//  SQLCODE +239 is issued. As part of this diagnostic, the second replacement text value
//  indicates the number of entries that are needed. The following code sample shows how
//  this condition can be detected and shows the descriptor allocated with the larger size.

// Determine the returned SQLCODE from the DESCRIBE statement */
EXEC SQL
  GET DIAGNOSTICS CONDITION 1: returned_sqlcode = DB2_RETURNED_SQLCODE;

// Get the second token for the SQLCODE that indicated
//   not enough entries were allocated
if returned_sqlcode = 239;
  EXEC SQL
    GET DIAGNOSTICS CONDITION 1: token = db2_ordinal_token_2;
  //* Move the token variable from a character host variable into an integer host variable */
```

```
  EXEC SQL
    SET :var1 = :token;
  //* Deallocate the descriptor that is too small */
  EXEC SQL
    DEALLOCATE DESCRIPTOR 'myDescriptor';
  //* Allocate the new descriptor to be the size indicated by the retrieved token */
  EXEC SQL
    ALLOCATE DESCRIPTOR 'myDescriptor' WITH MAX :var1;
  //* Perform the describe with the larger descriptor */
  EXEC SQL
    DESCRIBE s1 USING DESCRIPTOR 'myDescriptor';
endif;

// At this point, the descriptor contains the information about the SELECT statement.
//  Now you are ready to retrieve the SELECT statement results. For dynamic SQL, the
//  SELECT INTO statement is not allowed. You must use a cursor.
EXEC SQL
  DECLARE c1 CURSOR FOR s1;


// You will notice that the prepared statement name is used in the cursor declaration instead
//  of the complete SELECT statement. Now you can loop through the selected rows, processing
//  them as you read them. The following code sample shows how this is done.
EXEC SQL
  OPEN c1;

dou sqlcode = 100;       // while not at end of data;

  EXEC SQL
    FETCH c1 INTO SQL DESCRIPTOR 'myDescriptor';
  // SQLSTATE '01534' simply means that a date, time or timestamp is present in the data.
  // SQLSTATE '01548' means Authorization Warning.
  if sqlcode = 100 or (sqlcode <> 0 AND sqlcode <> 100
                       AND sqlstate <>'01534' AND sqlstate <> '01548');
    leave;
  endIf;

  //* process current data returned  (see below for discussion of doing this) */



  // The cursor is opened. The result rows from the SELECT statement are then returned one at a
time
  //  using a FETCH statement. On the FETCH statement, there is no list of output host variables.
  //  Instead, the FETCH statement tells SQL to return results into the descriptor area.

  // After the FETCH has been processed, you can use the GET DESCRIPTOR statement to read the
values
  //  First, you must read the header value that indicates how many descriptor entries were used.
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' :icount = COUNT;
  iRow += 1;


  // Next you can read information about each of the descriptor entries. After you determine the
dat
  //  type of the result column, you can do another GET DESCRIPTOR to return the actual value. To
  //  get the value of the indicator, specify the INDICATOR item. If the value of the INDICATOR
item
  //  is negative, the value of the DATA item is not defined. Until another FETCH is done, the
  //  descriptor items will maintain their values.
```

```
  //  (i = the number of fields that were returned).
  for i = 1 to iCount;
    Ds1(i).Data  = *blank;
    Ds1(i).Name  = *blank;
    Ds1(i).Label = *blank;

    EXEC SQL
      -- set entry number to get */
      -- get the data type */
      -- length value */
      //     :iscale = SCALE,
      //     :iprecision = PRECISION,
      GET DESCRIPTOR 'myDescriptor'
        VALUE :i :itype = TYPE,
                 :ilength = LENGTH,
                 :iresult_ind = INDICATOR,
                 :icolumn_name = DB2_COLUMN_NAME,
                 :ilabel = DB2_LABEL,
                 :iname = DB2_COLUMN_NAME,
                 :idatetimetype = DATETIME_INTERVAL_CODE;
        //     :idata = DATA;

    // Set the column headings. If the column lablel is blank, use the column name.
    if iresult_ind >= 0;
      if iLabel = *blank;
          ds1(i).Label = iName;
          ds1(i).Name  = iName;
        else;
          ds1(i).Label = iLabel;
          ds1(i).Name  = iName;
      endif;

    // Parse the columns. Check each of the column type and translate it into a common
    // field for output.
    SELECT;

// DATE/TIME/TIMESTAMPS
      when iType = iDate_time_Stamp;
        select;
        //when iLength = 8;
          when iDateTimeType = iDateType;
           EXEC SQL
             GET DESCRIPTOR 'myDescriptor'
               VALUE :i :idata_date_08 = DATA;
           ds1(i).Data = %char(iData_Date_08:*usa/);
          when iDateTimeType = iTimeType;
           EXEC SQL
             GET DESCRIPTOR 'myDescriptor'
               VALUE :i :idata_time_08 = DATA;
           ds1(i).Data = %char(iData_Time_08:*usa:);
          when iDateTimeType = iTimeStampType;
           EXEC SQL
             GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_stamp = DATA;
           ds1(i).Data = %char(iData_Stamp);
        endSl;

// CHARACTERS
      when iType = tCharacter;
       EXEC SQL
         GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_char = DATA;
       ds1(i).Data = %trim(iData_Char);
```

```
// VARYING CHARACTERS
      when iType = tVarChar;
       EXEC SQL
         GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_varchar = DATA;
       ds1(i).Data = %trim(iData_VarChar);

// INTEGERS
      when iType = tInteger;
       EXEC SQL
         GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_integer = DATA;
       ds1(i).Data = %editc(iData_Integer : 'Z');

// DECIMALS (KNOWN LENGTHS AND PRECISIONS ONLY)
      when iType = tDecimal;
        ExtractDecimalData();

    endsl; //* continue checking and processing for all data types that might be returned */
 endif;
endfor;



if iRow = 1;     // Open the IFS file and write the header row.
         opn_File();
endIf;

wrt_File();

enddo;
end-proc;


////////////////////////////////////////
//*=========================delete file procedure============================
// -------------------------------------------------
// Procedure name:  del_file
// Purpose:         Deletes the output file before recreating it.
// Returns:
// Author:          John Derr
// Created:         02/14/2018
//---------------------
DCL-PROC del_file  ;

DCL-PI *N   ;
END-PI ;

CMD = 'RMVLNK OBJLNK(''' + %trim(vFileName) + ''')';
SYSTEM (CMD);
END-PROC ;



////////////////////////////////////////
//*=========================create file procedure============================
// -------------------------------------------------
// Procedure name:  crt_File
// Purpose:         Creates the output file
// Returns:
// Author:          John Derr
// Created:         02/14/2018
```

```
//---------------------
DCL-PROC crt_File  ;

DCL-PI *N   ;
END-PI ;

  OFLAG = O_CREAT + O_CODEPAGE + O_RDWR + O_INHERITMODE;
  OMODE = S_IRWXU + S_IROTH;
  vFileName = %trim(vFileName) + NULL;
  ERR_FLAG = OPEN(%addr(vFileName):OFLAG:OMODE:CODEPAGE);
  IF ERR_FLAG < 0;
       return;
  ENDIF;
  RC=CLOSE(ERR_FLAG);

END-PROC ;




////////////////////////////////////////
//*=========================write file procedure============================
// ------------------------------------------------
// Procedure name:  wrt_File
// Purpose:         Writes a row to the output file
// Returns:
// Author:          John Derr
// Created:         02/14/2018
//---------------------

DCL-PROC wrt_File  ;

DCL-PI *N   ;
END-PI ;

  // If this is the first row, then write two rows:
  //   1. Column Headers
  //   2. Column data
  if iRow = 1 AND Include_Header = 'Y';
    for i = 1 to iCount;
      // If this is not the last field, comma separate it.
    if String_Separator <> *blank;
      if i < iCount;
       outrec = %trim(outRec)+String_Separator+%trim(ds1(i).Label)+String_Separator +
                Column_Separator;
      // If it is the last field, insert the end-of-line constant.
      else;
       outrec = %trim(outRec)+String_Separator+%trim(ds1(i).Label)+String_Separator + eola;
      endIf;
    else;
      if i < iCount;
       outrec = %trim(outRec) + %trim(ds1(i).Label   )  + Column_Separator;
      // If it is the last field, insert the end-of-line constant.
      else;
       outrec = %trim(outRec) + %trim(ds1(i).Label   )  + eola;
      endIf;
    endif;
    endFor;
  str    = (%scan(eola:outrec)) + 1;
  bufLen =  STR;
  rc = write(err_flag:%addr(outrec):buflen);
```

```
  clear outrec;

  endIf;

  // Now process the data.
  for i = 1 to iCount;
    if String_Separator <> *blank;
      if i < iCount;
       outrec = %trim(outRec)+String_Separator+%trim(ds1(i).data)+String_Separator +
                Column_Separator;
      else;
       outrec = %trim(outRec)+String_Separator+%trim(ds1(i).data)+String_Separator + eola;
      endIf;
    else;
      if i < iCount;
       outrec = %trim(outRec) + %trim(ds1(i).data) + Column_Separator;
      else;
       outrec = %trim(outRec) + %trim(ds1(i).data) + eola;
      endIf;
    endif;
  endFor;

  str    = (%scan(eola:outrec)) + 1;
  bufLen =  STR;
  rc = write(err_flag:%addr(outrec):buflen);
  clear outrec;

END-PROC ;




//////////////////////////////////////
//*=======================open file procedure============================
// ------------------------------------------------
// Procedure name:  opn_File
// Purpose:         Opens the output file for processing
// Returns:
// Author:          John Derr
// Created:         02/14/2018
//---------------------
DCL-PROC opn_File  ;

DCL-PI *N   ;
END-PI ;

  OFLAG = O_WRONLY + O_TEXTDATA;
  ERR_FLAG = OPEN(%addr(vFileName):OFLAG);

   IF ERR_FLAG < 0;
     return;
   ENDIF;

END-PROC ;




//////////////////////////////////////
//*=======================Extract Decimal Data ============================
```

```
// ------------------------------------------------
// Procedure name:  ExtractNumericData
// Purpose:         Identifies the returned column's decimal length and precision.
// Returns:
// Author:          John Derr
// Created:         02/14/2018
//---------------------
DCL-PROC ExtractDecimalData  ;

DCL-PI *N   ;
END-PI ;

  // The following permutations of length and scale were derived from running a
  // "distinct" query of all data on DATEDW.
  dcl-s iData_Dec01_00   packed(01:00);
  dcl-s iData_Dec02_00   packed(02:00);
  dcl-s iData_Dec02_01   packed(02:01);
  dcl-s iData_Dec02_02   packed(02:02);
  dcl-s iData_Dec03_00   packed(03:00);
  dcl-s iData_Dec03_01   packed(03:01);
  dcl-s iData_Dec03_02   packed(03:02);
  dcl-s iData_Dec03_03   packed(03:03);
  dcl-s iData_Dec04_00   packed(04:00);
  dcl-s iData_Dec04_01   packed(04:01);
  dcl-s iData_Dec04_02   packed(04:02);
  dcl-s iData_Dec04_03   packed(04:03);
  dcl-s iData_Dec04_04   packed(04:04);
  dcl-s iData_Dec05_00   packed(05:00);
  dcl-s iData_Dec05_01   packed(05:01);
  dcl-s iData_Dec05_02   packed(05:02);
  dcl-s iData_Dec05_03   packed(05:03);
  dcl-s iData_Dec05_04   packed(05:04);
  dcl-s iData_Dec05_05   packed(05:05);
  dcl-s iData_Dec06_00   packed(06:00);
  dcl-s iData_Dec06_02   packed(06:02);
  dcl-s iData_Dec06_03   packed(06:03);
  dcl-s iData_Dec06_04   packed(06:04);
  dcl-s iData_Dec06_05   packed(06:05);
  dcl-s iData_Dec07_00   packed(07:00);
  dcl-s iData_Dec07_02   packed(07:02);
  dcl-s iData_Dec07_04   packed(07:04);
  dcl-s iData_Dec07_05   packed(07:05);
  dcl-s iData_Dec07_06   packed(07:06);
  dcl-s iData_Dec07_07   packed(07:07);
  dcl-s iData_Dec08_00   packed(08:00);
  dcl-s iData_Dec08_02   packed(08:02);
  dcl-s iData_Dec08_05   packed(08:05);
  dcl-s iData_Dec08_06   packed(08:06);
  dcl-s iData_Dec08_08   packed(08:08);
  dcl-s iData_Dec09_00   packed(09:00);
  dcl-s iData_Dec09_01   packed(09:01);
  dcl-s iData_Dec09_02   packed(09:02);
  dcl-s iData_Dec09_03   packed(09:03);
  dcl-s iData_Dec09_04   packed(09:04);
  dcl-s iData_Dec09_05   packed(09:05);
  dcl-s iData_Dec09_06   packed(09:06);
  dcl-s iData_Dec09_07   packed(09:07);
  dcl-s iData_Dec09_08   packed(09:08);
  dcl-s iData_Dec10_00   packed(10:00);
  dcl-s iData_Dec10_02   packed(10:02);
  dcl-s iData_Dec10_05   packed(10:05);
```

```
dcl-s iData_Dec10_09   packed(10:09);
dcl-s iData_Dec11_00   packed(11:00);
dcl-s iData_Dec11_02   packed(11:02);
dcl-s iData_Dec11_04   packed(11:04);
dcl-s iData_Dec11_05   packed(11:05);
dcl-s iData_Dec11_09   packed(11:09);
dcl-s iData_Dec12_00   packed(12:00);
dcl-s iData_Dec12_02   packed(12:02);
dcl-s iData_Dec12_09   packed(12:09);
dcl-s iData_Dec13_00   packed(13:00);
dcl-s iData_Dec13_02   packed(13:02);
dcl-s iData_Dec13_05   packed(13:05);
dcl-s iData_Dec13_10   packed(13:10);
dcl-s iData_Dec14_00   packed(14:00);
dcl-s iData_Dec14_02   packed(14:02);
dcl-s iData_Dec15_00   packed(15:00);
dcl-s iData_Dec15_02   packed(15:02);
dcl-s iData_Dec15_05   packed(15:05);
dcl-s iData_Dec15_06   packed(15:06);
dcl-s iData_Dec15_08   packed(15:08);
dcl-s iData_Dec15_09   packed(15:09);
dcl-s iData_Dec16_00   packed(16:00);
dcl-s iData_Dec16_02   packed(16:02);
dcl-s iData_Dec16_05   packed(16:05);
dcl-s iData_Dec17_00   packed(17:00);
dcl-s iData_Dec17_02   packed(17:02);
dcl-s iData_Dec17_05   packed(17:05);
dcl-s iData_Dec17_16   packed(17:16);
dcl-s iData_Dec18_00   packed(18:00);
dcl-s iData_Dec18_02   packed(18:02);
dcl-s iData_Dec18_05   packed(18:05);
dcl-s iData_Dec19_00   packed(19:00);
dcl-s iData_Dec20_00   packed(20:00);
dcl-s iData_Dec20_02   packed(20:02);
dcl-s iData_Dec21_00   packed(21:00);
dcl-s iData_Dec21_01   packed(21:01);
dcl-s iData_Dec21_05   packed(21:05);
dcl-s iData_Dec22_00   packed(22:00);
dcl-s iData_Dec23_00   packed(23:00);
dcl-s iData_Dec25_08   packed(25:08);
dcl-s iData_Dec26_09   packed(26:09);
dcl-s iData_Dec31_00   packed(31:00);
dcl-s iData_Dec31_02   packed(31:02);

EXEC SQL
  GET DESCRIPTOR 'myDescriptor'
    VALUE :i :iprecision = PRECISION,
             :iscale = SCALE;

// The following permutations of length and scale were derived from running a
// "distinct" query of all data on DATEDW. As new conditions are discovered,
// simply add them to this process.
select;
  when                                    iPrecision = 01 AND
                                              iScale = 00;
    EXEC SQL
      GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec01_00 = DATA;
    ds1(i).Data =                %trim(%editc(iData_Dec01_00 : 'N'));

  when                                    iPrecision = 02 AND
                                              iScale = 00;
```

```
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec02_00 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec02_00 : 'N'));

when                                    iPrecision = 02 AND
                                            iScale = 01;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec02_01 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec02_01 : 'N'));

when                                    iPrecision = 02 AND
                                            iScale = 02;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec02_02 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec02_02 : 'N'));

when                                    iPrecision = 03 AND
                                            iScale = 00;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec03_00 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec03_00 : 'N'));

when                                    iPrecision = 03 AND
                                            iScale = 01;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec03_01 = DATA;
  ds1(i).Data =               %trim(%editc(iData_Dec03_01 : 'N'));

when                                    iPrecision = 03 AND
                                            iScale = 02;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec03_02 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec03_02 : 'N'));

when                                    iPrecision = 03 AND
                                            iScale = 03;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec03_03 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec03_03 : 'N'));

when                                    iPrecision = 04 AND
                                            iScale = 00;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec04_00 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec04_00 : 'N'));

when                                    iPrecision = 04 AND
                                            iScale = 01;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec04_01 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec04_01 : 'N'));

when                                    iPrecision = 04 AND
                                            iScale = 02;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec04_02 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec04_02 : 'N'));

when                                    iPrecision = 04 AND
                                            iScale = 03;
  EXEC SQL
```

```
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec04_03 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec04_03 : 'N'));

when                                    iPrecision = 04 AND
                                            iScale = 04;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec04_04 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec04_04 : 'N'));

when                                    iPrecision = 05 AND
                                            iScale = 00;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec05_00 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec05_00 : 'N'));

when                                    iPrecision = 05 AND
                                            iScale = 01;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec05_01 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec05_01 : 'N'));

when                                    iPrecision = 05 AND
                                            iScale = 02;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec05_02 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec05_02 : 'N'));

when                                    iPrecision = 05 AND
                                            iScale = 03;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec05_03 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec05_03 : 'N'));

when                                    iPrecision = 05 AND
                                            iScale = 04;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec05_04 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec05_04 : 'N'));

when                                    iPrecision = 05 AND
                                            iScale = 05;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec05_05 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec05_05 : 'N'));

when                                    iPrecision = 06 AND
                                            iScale = 00;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec06_00 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec06_00 : 'N'));

when                                    iPrecision = 06 AND
                                            iScale = 02;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec06_02 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec06_02 : 'N'));

when                                    iPrecision = 06 AND
                                            iScale = 03;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec06_03 = DATA;
```

```
  ds1(i).Data =                %trim(%editc(iData_Dec06_03 : 'N'));

when                                    iPrecision = 06 AND
                                            iScale = 04;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec06_04 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec06_04 : 'N'));

when                                    iPrecision = 06 AND
                                            iScale = 05;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec06_05 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec06_05 : 'N'));

when                                    iPrecision = 07 AND
                                            iScale = 00;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec07_00 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec07_00 : 'N'));

when                                    iPrecision = 07 AND
                                            iScale = 02;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec07_02 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec07_02 : 'N'));

when                                    iPrecision = 07 AND
                                            iScale = 04;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec07_04 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec07_04 : 'N'));

when                                    iPrecision = 07 AND
                                            iScale = 05;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec07_05 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec07_05 : 'N'));

when                                    iPrecision = 07 AND
                                            iScale = 06;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec07_06 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec07_06 : 'N'));

when                                    iPrecision = 07 AND
                                            iScale = 07;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec07_07 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec07_07 : 'N'));

when                                    iPrecision = 08 AND
                                            iScale = 00;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec08_00 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec08_00 : 'N'));

when                                    iPrecision = 08 AND
                                            iScale = 02;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec08_02 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec08_02 : 'N'));
```

```
when                                    iPrecision = 08 AND
                                            iScale = 05;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec08_05 = DATA;
  ds1(i).Data =                 %trim(%editc(iData_Dec08_05 : 'N'));

when                                    iPrecision = 08 AND
                                            iScale = 06;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec08_06 = DATA;
  ds1(i).Data =                 %trim(%editc(iData_Dec08_06 : 'N'));

when                                    iPrecision = 08 AND
                                            iScale = 08;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec08_08 = DATA;
  ds1(i).Data =                 %trim(%editc(iData_Dec08_08 : 'N'));

when                                    iPrecision = 09 AND
                                            iScale = 00;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec09_00 = DATA;
  ds1(i).Data =                 %trim(%editc(iData_Dec09_00 : 'N'));

when                                    iPrecision = 09 AND
                                            iScale = 01;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec09_01 = DATA;
  ds1(i).Data =                 %trim(%editc(iData_Dec09_01 : 'N'));

when                                    iPrecision = 09 AND
                                            iScale = 02;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec09_02 = DATA;
  ds1(i).Data =                 %trim(%editc(iData_Dec09_02 : 'N'));

when                                    iPrecision = 09 AND
                                            iScale = 03;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec09_03 = DATA;
  ds1(i).Data =                 %trim(%editc(iData_Dec09_03 : 'N'));

when                                    iPrecision = 09 AND
                                            iScale = 04;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec09_04 = DATA;
  ds1(i).Data =                 %trim(%editc(iData_Dec09_04 : 'N'));

when                                    iPrecision = 09 AND
                                            iScale = 05;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec09_05 = DATA;
  ds1(i).Data =                 %trim(%editc(iData_Dec09_05 : 'N'));

when                                    iPrecision = 09 AND
                                            iScale = 06;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec09_06 = DATA;
  ds1(i).Data =                 %trim(%editc(iData_Dec09_06 : 'N'));
```

```
when                                    iPrecision = 09 AND
                                            iScale = 07;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec09_07 = DATA;
  ds1(i).Data =                 %trim(%editc(iData_Dec09_07 : 'N'));

when                                    iPrecision = 09 AND
                                            iScale = 08;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec09_08 = DATA;
  ds1(i).Data =                 %trim(%editc(iData_Dec09_08 : 'N'));

when                                    iPrecision = 10 AND
                                            iScale = 00;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec10_00 = DATA;
  ds1(i).Data =                 %trim(%editc(iData_Dec10_00 : 'N'));

when                                    iPrecision = 10 AND
                                            iScale = 02;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec10_02 = DATA;
  ds1(i).Data =                 %trim(%editc(iData_Dec10_02 : 'N'));

when                                    iPrecision = 10 AND
                                            iScale = 05;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec10_05 = DATA;
  ds1(i).Data =                 %trim(%editc(iData_Dec10_05 : 'N'));

when                                    iPrecision = 10 AND
                                            iScale = 09;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec10_09 = DATA;
  ds1(i).Data =                 %trim(%editc(iData_Dec10_09 : 'N'));

when                                    iPrecision = 11 AND
                                            iScale = 00;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec11_00 = DATA;
  ds1(i).Data =                 %trim(%editc(iData_Dec11_00 : 'N'));

when                                    iPrecision = 11 AND
                                            iScale = 02;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec11_02 = DATA;
  ds1(i).Data =                 %trim(%editc(iData_Dec11_02 : 'N'));

when                                    iPrecision = 11 AND
                                            iScale = 04;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec11_04 = DATA;
  ds1(i).Data =                 %trim(%editc(iData_Dec11_04 : 'N'));

when                                    iPrecision = 11 AND
                                            iScale = 05;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec11_05 = DATA;
  ds1(i).Data =                 %trim(%editc(iData_Dec11_05 : 'N'));

when                                    iPrecision = 11 AND
```

```
                                        iScale = 09;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec11_09 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec11_09 : 'N'));

when                                    iPrecision = 12 AND
                                        iScale = 00;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec12_00 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec12_00 : 'N'));

when                                    iPrecision = 12 AND
                                        iScale = 02;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec12_02 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec12_02 : 'N'));

when                                    iPrecision = 12 AND
                                        iScale = 09;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec12_09 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec12_09 : 'N'));

when                                    iPrecision = 13 AND
                                        iScale = 00;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec13_00 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec13_00 : 'N'));

when                                    iPrecision = 13 AND
                                        iScale = 02;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec13_02 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec13_02 : 'N'));

when                                    iPrecision = 13 AND
                                        iScale = 05;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec13_05 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec13_05 : 'N'));

when                                    iPrecision = 13 AND
                                        iScale = 10;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec13_10 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec13_10 : 'N'));

when                                    iPrecision = 14 AND
                                        iScale = 00;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec14_00 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec14_00 : 'N'));

when                                    iPrecision = 14 AND
                                        iScale = 02;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec14_02 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec14_02 : 'N'));

when                                    iPrecision = 15 AND
                                        iScale = 00;
```

```
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec15_00 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec15_00 : 'N'));

when                                    iPrecision = 15 AND
                                            iScale = 02;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec15_02 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec15_02 : 'N'));

when                                    iPrecision = 15 AND
                                            iScale = 05;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec15_05 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec15_05 : 'N'));

when                                    iPrecision = 15 AND
                                            iScale = 06;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec15_06 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec15_06 : 'N'));

when                                    iPrecision = 15 AND
                                            iScale = 08;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec15_08 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec15_08 : 'N'));

when                                    iPrecision = 15 AND
                                            iScale = 09;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec15_09 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec15_09 : 'N'));

when                                    iPrecision = 16 AND
                                            iScale = 00;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec16_00 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec16_00 : 'N'));

when                                    iPrecision = 16 AND
                                            iScale = 02;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec16_02 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec16_02 : 'N'));

when                                    iPrecision = 16 AND
                                            iScale = 05;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec16_05 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec16_05 : 'N'));

when                                    iPrecision = 17 AND
                                            iScale = 00;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec17_00 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec17_00 : 'N'));

when                                    iPrecision = 17 AND
                                            iScale = 02;
  EXEC SQL
```

```
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec17_02 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec17_02 : 'N'));

when                                    iPrecision = 17 AND
                                            iScale = 05;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec17_05 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec17_05 : 'N'));

when                                    iPrecision = 17 AND
                                            iScale = 16;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec17_16 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec17_16 : 'N'));

when                                    iPrecision = 18 AND
                                            iScale = 00;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec18_00 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec18_00 : 'N'));

when                                    iPrecision = 18 AND
                                            iScale = 02;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec18_02 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec18_02 : 'N'));

when                                    iPrecision = 18 AND
                                            iScale = 05;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec18_05 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec18_05 : 'N'));

when                                    iPrecision = 19 AND
                                            iScale = 00;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec19_00 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec19_00 : 'N'));

when                                    iPrecision = 20 AND
                                            iScale = 00;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec20_00 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec20_00 : 'N'));

when                                    iPrecision = 20 AND
                                            iScale = 02;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec20_02 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec20_02 : 'N'));

when                                    iPrecision = 21 AND
                                            iScale = 00;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec21_00 = DATA;
  ds1(i).Data =                %trim(%editc(iData_Dec21_00 : 'N'));

when                                    iPrecision = 21 AND
                                            iScale = 01;
  EXEC SQL
    GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec21_01 = DATA;
```

```
      ds1(i).Data =                %trim(%editc(iData_Dec21_01 : 'N'));

    when                                    iPrecision = 21 AND
                                                iScale = 05;
      EXEC SQL
        GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec21_05 = DATA;
      ds1(i).Data =                %trim(%editc(iData_Dec21_05 : 'N'));

    when                                    iPrecision = 22 AND
                                                iScale = 00;
      EXEC SQL
        GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec22_00 = DATA;
      ds1(i).Data =                %trim(%editc(iData_Dec22_00 : 'N'));

    when                                    iPrecision = 23 AND
                                                iScale = 00;
      EXEC SQL
        GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec23_00 = DATA;
      ds1(i).Data =                %trim(%editc(iData_Dec23_00 : 'N'));

    when                                    iPrecision = 25 AND
                                                iScale = 08;
      EXEC SQL
        GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec25_08 = DATA;
      ds1(i).Data =                %trim(%editc(iData_Dec25_08 : 'N'));

    when                                    iPrecision = 26 AND
                                                iScale = 09;
      EXEC SQL
        GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec26_09 = DATA;
      ds1(i).Data =                %trim(%editc(iData_Dec26_09 : 'N'));

    when                                    iPrecision = 31 AND
                                                iScale = 00;
      EXEC SQL
        GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec31_00 = DATA;
      ds1(i).Data =                %trim(%editc(iData_Dec31_00 : 'N'));

    when                                    iPrecision = 31 AND
                                                iScale = 02;
      EXEC SQL
        GET DESCRIPTOR 'myDescriptor' VALUE :i :idata_dec31_02 = DATA;
      ds1(i).Data =                %trim(%editc(iData_Dec31_02 : 'N'));

    other;
      ds1(i).Data = 'Error on row ' + %trim(%editc(iRow : 'Z')) + ', Column ' +
        %trim(iColumn_Name) + ' ';

  endsl;


END-PROC ;
```