

Obtaining and Using Euro Currency Rates in SAS® Programs
Rick Langston
SAS Institute, Inc., Cary, NC

BACKGROUND

The euro became the official currency of the European Economic Community on January 1, 1999. On that date, the currencies of eleven countries of Europe became the euro in lieu of their historical currencies. Those eleven currencies are the Belgian franc, the German mark, the Spanish peseta, the French franc, the Irish pound, the Luxembourg franc, the Dutch florin, the Austrian schilling, the Portuguese escudo, and the Finnish markka. Other countries will eventually join the list as their respective governments and voting public decide to do so. (Greece has since done this, on January 1, 2001). On September 28, 2000 Denmark voted down conversion to the euro, as another case. Other European countries are in various phases of decisions on the subject of conversion.

The SAS System provides a set of functions and formats to facilitate the conversion between the euro and historical currencies, and also conversion between the euro and the currencies of non-adherent countries.

These formats and informats are called EURFRxxx and EURTOxxx (where xxx is the standard 3-character abbreviation for the country). The function is EUROCRR, which allows for conversion between any two currencies.

The informat/format/function set (herewith referred to as the euro IFF set) use a built-in table to obtain conversion rates. This works fine for the eleven initial adherent countries, because their conversion rates were made irrevocable with the adoption of the euro. However, for all other currencies, their rates fluctuate constantly in an open currency market. This aspect was recognized when the euro IFF set was first implemented, and the rates can be specified by an external table and/or macro variables, either of which will override the builtin rates.

This paper describes a method for obtaining the most current rates and incorporating them into a dynamic table so that the euro IFF set will give properly conversion results.

OFFICIAL CURRENCY RATES

The official governing bank for the euro is the European Central Bank (ECB). Their web site, www.ecb.int, is the official web site to obtain rates pertaining to the euro. As of the time of this writing, the specific web page for obtaining currency rates is

<http://www.ecb.int/home/eurofxref.htm>

This web page contains rates displaying in the following fashion (using a snapshot from May 29, 2001):

Currency	Spot	Currency	Spot
USD US dollar	0.8552	LVN Latvian lat	0.5416
JPY Japanese yen	102.88	MTL Maltese lira	0.3955
DKK*Danish krone	7.4575	PLN*Polish zloty	3.4529
GBP*Pound sterling	0.60320	ROL*Romanian leu	24540
SEK*Swedish krona	9.0350	SIT*Slovenian tolar	217.5675
CHF*Swiss franc	1.5260	SKK Slovakian koruna	42.938
ISK Icelandic krona	88.24	TRL*Turkish lira	956700
NOK*Norwegian krone	7.8865	AUD Australian dollar	1.6608
BGN Bulgarian lev	1.9461	CAD Canadian dollar	1.3167
CYP Cyprus pound	0.57690	HKD Hong Kong dollar	6.6704
CZK*Czech koruna	34.219	NZD New Zealand dollar	2.0285
EEK Estonian kroon	15.6466	SGD Singaporean dollar	1.5449
HUF*Hungarian forint	254.00	KRW South Korean won	1102.78
LTN Lithuanian litas	3.4217	ZAR South African rand	6.7732

Because the web page consists of an HTML table and it is displayed based on your browser, the information seen above won't display the same way on your screen. I have made two other modifications to what is seen above: all decimals line up, and I've put a * beside the currencies actually recognized by the euro IFF set. Also recognized by the euro IFF set but not displayed above are the Russian ruble (RUR) and the Yugoslavian dinar (YUD).

Note the use of the 3-character abbreviation. This is the same abbreviation used by the euro IFF set. Not all currencies listed here are recognized by the euro IFF set.

Note also that these rates indicate the value of 1 euro in the specified currency. For example, 1 euro was worth .8552 US dollars on May 29, 2001. If you wanted to know how many euros were in one US dollar, you'd need the reciprocal of this rate (1/.8552, or 1.1693). This means it took 1.1693 euros to make one US dollar on May 29, 2001.

EXTERNAL TABLES WITH THE EURO IFF SET

The euro IFF set can use an external table for its rates. The table is stored in a file referenced by the EURFRTBL fileref. The entries in the table are as follows:

```
EURFRxxx=rate1
EURFRyyy=rate2
EURFRzzz=rate3
```

where xxx, yyy, and zzz are 3-character abbreviations for currencies. The values indicated by rate1, rate2, and rate3 are the number of units of the currency comprising one euro, just like the rates that appear in the ECB web site table. For example, the EURFRTBL entry for pounds sterling, using the rates seen in our May 29 table above, would be

```
EURFRGBP=0.60320
```

As a complete example:

```
filename eurfrtbl temp;
data _null_; file eurfrtbl;
    input; put _infile_ cards4;
EURFRGBP=0.60320
;;;

data _null_;
    n_euros = eurocurr(1,'gbp','eur');
    n_pounds = eurocurr(1,'eur','gbp');
    put n_euros= n_pounds=;
run;
```

The result is

```
n_euros=1.6578249337
n_pounds=0.6032
```

The EUROCURR function has the arguments

```
to_units =
eurocurr(from_units,from_curr,to_curr);
```

So in our example, the first use of EUROCURR is to convert 1 pound sterling into euros. The second use is to convert 1 euro to pounds sterling. As we expect, n_pounds is equal to the rate given in the EURFRGBP value.

MERGING THE ECB TABLES WITHIN THE SAS PROGRAM

We now know we can obtain current rates from the ECB web site, and we can dynamically specify a table for the euro IFF set to use. So here's how we can merge these abilities into a single SAS program.

In version 8 of the SAS System, the HTTP access method was made generally available. We can use this access method to read web pages directly. The syntax for this access method is as follows:

```
filename fileref HTTP 'web-page-address'
authentication-info;
```

where fileref is the fileref you want to use 'web-page-address' is the web site address, and authentication-info is whatever is necessary to access external web pages. It may be likely that you'll need a userid, password, and proxy address, depending on your security setup:

```
filename fileref HTTP 'web-page-address'
proxy='address' userid=userid
pass='password' ;
```

An approach I used in testing this access method on a Unix system was to read my userid and password information from the .netrc file, looking for a particular machine (in this example, called abc) to obtain the userid and password:

```
data _null_; infile '~/.netrc' length=1;
    input @; input @1 line $varying200. 1;
    machine=scan(line,2);
    if machine='abc';
    userid=scan(line,4);
    pass=scan(line,6);
    call symput('myuserid',trim(userid));
    call symput('mypass',trim(pass));
run;
```

```
filename xxx http
'http://www.ecb.int/home/eurofxref.htm'
user=&myuserid. pass=&mypass."
proxy='<our proxy machine name>';
```

```
data _null_;
  call symput('myuserid',' ');
  call symput('mypass',' ');
run;
```

This SAS code allows me to use my real userid and password but without having to write it in a SAS program. Note that as soon as the FILENAME statement is processed, I reset those macro variables to blanks to avoid exposure of the values.

Note that if you are running a version of the SAS System prior to Version 8, and you don't have access to the HTTP access method, an alternative may be to use the lynx command. The lynx command is available on a variety of

platforms. See the official lynx web site at <http://lynx.browser.org>. The FILENAME statement for lynx would incorporate the use of the PIPE method:

```
filename xxx pipe 'lynx -source
http://www.ecb.int/home/eurofxref.htm';
```

The SAS code to read from this fileref is exactly the same as that from the fileref using the HTTP access method.

Here's the SAS code to read the HTML tables as they currently exist at the ECB web site. The SAS code also creates the EURFRTBL table that the euro IFF set will need, as well as %LET statements for the macro symbol version of these rates.

```
/*-----*/
/* Read in the currency abbreviations for the currencies that are */
/* recognized by the euro IFF set. The first 11 currencies are those */
/* with the 01JAN1999 irrevocable rates */
/*-----*/

data curr;
  length abbr $3;
  input abbr $ @@;
  first11=_n_<=11;
  cards;
ATS BEF FIM FRF DEM IEP ITL LUF NLG PTE ESP CHF DKK GBP GRD SEK CZK HUF NOK RUR TRL PLZ ROL
YUD SIT
;
run;
proc sort data=curr; by abbr;

/*-----*/
/* Read the rates from the ECB web site. */
/*-----*/

data rates(keep=abbr desc rate);
  infile xxx length=1;
  retain part 0;
  length desc $40 abbr $3;
  retain abbr desc;

/*-----*/
/* Read the line of HTML. The HTML lines at the time of this */
/* implementation had the 0x0d 0x0a carriage-return / line-feed in */
/* MS-DOS style. On non-Windows systems, the 0x0d will remain as a */
/* data character, so we need to remove the character. Also we */
/* create an upcased version of the line so we can look at HTML tags */
/* with case-insensitivity. */
/*-----*/
```

```

input @; input @1 line $varying200. 1;
line=left(compress(line,'Od'x));
uline=upcase(line);

/*-----*/
/* We will effectively ignore all HTML text until the TABLE tag is */
/* seen. our 'table' variable indicates we've seen this tag.      */
/*-----*/

if uline='<TABLE' then do;
    table=1;
    retain table;
    return;
end;
if table;

/*-----*/
/* At this point we'll be reading from the HTML table of rates. We */
/* will traverse through every token on the line. Tokens are       */
/* separated by < and >, so HTML tags and the data are separate. We */
/* are only interested in the data between the <TD> </TD> tags.    */
/* We are interested in the 3 data items that appear in the TD     */
/* parts. Note an example of the section:                          */
/* <tr bgcolor="#fff7cc">                                          */
/* <td align="center">USD</td>                                     */
/* <td>US dollar</td>                                             */
/* <td align="right">0.8480</td>                                   */
/* <td bgcolor="#ffffff"></td>                                     */
/* <td align="center">LVL</td>                                     */
/* <td>Latvian lat</td>                                           */
/* <td align="right">0.5377</td>                                   */
/* </tr>                                                          */
/* We see both the USD and LVL currencies defined in this row.    */
/*-----*/

td=0;
do i=1 to 100; /* do while(1); is more appropriate but dangerous */
    piece=left(scan(uline,i,'<>'));
    if piece='/TABLE'
        then stop;
    if piece=' '
        then leave;
    if piece='TD ' then do;
        td=1;
        end;
    else if piece='/TD' then td=0;
    else if td then do;
        part=mod(part,3);
        if part=0 then do;
            abbr=piece;
            end;
        if part=1 then do;
            desc=left(scan(line,i,'<>')); /* get original casing */
            end;
        else if part=2 then do;

```

```

        rate=input(piece,best12.);
        output;
        end;
        part+1;
        end;
    end;
run;
proc sort data=rates; by abbr;

/*-----*/
/* This macro performs conversions from all desired currencies to euro, */
/* using the builtin rates or the EURFRTBL rates if they are available. */
/*-----*/

%macro doconv(iter);
data conv&iter.; set curr(where=(first11=0));
    value&iter.=eurocurr(1,abbr,'EUR');
run;
%mend;

/*-----iteration 1: using builtin rates-----*/

%doconv(1);

/*-----*/
/* Create the EURFRTBL table using the rates we obtained from the ECB */
/* web site. We only emit the rates for the currencies that we are */
/* interested in. Be sure to add the greek drachma, now with an */
/* irrevocable rate. Other adoptive currencies would be added in this */
/* way, since they rates will not be in the ECB table. */
/*-----*/

filename eurfrtbl temp;

data curr;
    file eurfrtbl;
    length abbr $3;
    merge rates(in=have) curr(in=want where=(first11=0)); by abbr;
    if _n_=1 then put 'EURFRGRD=340.750'; /* Greek drachma */
    if want then do;
        newrate=have;
        output curr;
        end;
    if want and have;
    put @1 'EURFR' abbr $char3. '=' rate;
run;

data _null_; infile eurfrtbl; input; put _infile_; run;

/*-----iteration 2: using eurfrtbl rates-----*/
%doconv(2);

/*-----*/
/* Now merge the different converted values (builtin vs. current rate) */
/* to see how they compare. */
/*-----*/

```

```
data _null_; merge conv1 conv2; by abbr;
  put abbr= value1= value2=;
run;
```

Here is what the EURFRTBL looks like, using
May 30, 2001 rates:

```
EURFRGRD=340.750
EURFRCHF=1.5206
EURFRCZK=34.195
EURFRDKK=7.4556
EURFRGBP=0.5973
EURFRHUF=253.75
EURFRNOK=7.93
EURFRROL=24392
EURFRSEK=9.125
EURFRSIT=217.6353
EURFRTL=995000
```

Here is the output from the final merged DATA
step:

abbr=CHF	value1=0.6233248146	value2=0.657635144
abbr=CZK	value1=0.0286892183	value2=0.0292440415
abbr=DKK	value1=0.1335097442	value2=0.1341273673
abbr=GBP	value1=1.4283020916	value2=1.6742005692
abbr=GRD	value1=0.0029335406	value2=0.0029347029
abbr=HUF	value1=0.0038413522	value2=0.0039408867
abbr=NOK	value1=0.1087228329	value2=0.1261034048
abbr=PLZ	value1=0.2380952381	value2=0.2380952381
abbr=ROL	value1=0.0729394602	value2=0.000040997
abbr=RUR	value1=0.050586807	value2=0.050586807
abbr=SEK	value1=0.106770191	value2=0.1095890411
abbr=SIT	value1=0.0052356021	value2=0.0045948428
abbr=TRL	value1=0.0029681341	value2=1.0050251E-6
abbr=YUD	value1=0.0765438903	value2=0.0765438903

The values for PLZ, RUR, and YUD did not
change since they don't appear in the ECB
table. Some (TRL and ROL in particular) have
changed drastically due to market fluctuations
since the builtin table was created.

USING PICTURE AND MULTIPLIERS FOR THE RATES

Although the US dollar (USD) symbol is not part of the euro IFF set, it is set by our code above. If we want to convert between the USD and euro, the US dollar, we can incorporate it into a MULT= option of a PICTURE format. However, you have to be aware of what the MULT= does with respect to a decimal point. Consider the following SAS code:

```
/* Create a macro variable for the US
Dollar rate */

data _null_; set rates;
  if abbr='USD';
  call symput('EURFRUSD',
    trim(
      left(put(rate,best12.))));
run;

/* Create a picture format using this rate
as the multiplier */

proc format;
  picture usd other='000,000,009.99'
    (prefix='$' mult=&eurfrusd.);
run;

/* Try this format with $1 to see the
result as the number of euros.
It will be wrong. */

data _null_; x=1; put x=usd.; run;
```

Indeed, the result is

x=\$0.00

We don't get what we expect! This is because without a MULT= option, picture processing will multiply by 10**n, where n is the number of places to the right of the decimal point. But if a MULT= option is given, as it is here, no accommodation for the decimal point takes place, and it's assumed that the number has already been multiplied by the number of decimal places, or that the MULT= value factors this in.

```
/* Now try the following, which does work */

data _null_; x=1; x=x*100; put x=usd.; run;
```

This results in

x=\$0.84

Meaning that 1 euro is worth .84 US dollars.

```
/* Instead create with the multiplier
included */

data _null_; set rates;
  if abbr='USD';
  call symput('EURFRUSD',
    trim(
      left(
        put(rate*100,best12.))));
run;
proc format;
  picture usd other='000,000,009.99'
    (prefix='$' mult=&eurfrusd.);

/* This time the answer is as expected */

data _null_; x=1; put x=usd.; run;
```

And it is also generated as

x=\$0.84

BEING AWARE OF OVERUSE OF WEB SITES

This SAS program will read the ECB site for rates. Please be aware that this site is accessed by users all over the world, and you should not access it more often than necessary. The values are changed each business day, around 1:30pm GMT. If you need to have access to these values, it is probably best to make a local copy of the values and then access that local copy throughout the day. Otherwise the CEB web site may be overwhelmed by accesses.

SAS is a registered trademark or trademark of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.