Subject: Re: Specifying DOUBLE precision and using WHERE
Posted by Richard G. French on Sat, 26 Jun 1999 07:00:00 GMT
View Forum Message <> Reply to Message

Johnny Lin wrote:
>
> hi folks,
>
> i'm using WHERE to do a test for a missing value in an array of data.
> the funny thing is the result of WHERE seems to change depending on
> whether I use the function DOUBLE to set the type of the missing
> value field (or of the data i'm testing), or whether I set it using
> "d" as the exponent. is there a difference in specifying the type
> of a variable using "d" vs. DOUBLE? below is a sample of 4 different
> cases that illustrate what i'm describing.

As far as I can tell, everything is working the way it should in this example.
There is roundoff error in the single precision representation of your value of missing, so when you compare a single and double precision version of
'missing', they will not be equal to each other. If you set 'missing' to something that can be represented WITHOUT roundoff error in single and double precision, I think you would get different results:

```
;CASE #1 (missing and data are both FLOAT):
missing=-9.e+0
data=randomn(-34,20)
data[where(data lt 0.1)]=missing
print,data
print,where(data eq missing)

;CASE #2 (missing is DOUBLE using "d" exponent and data is FLOAT):
missing=-9.d+0
data=randomn(-34,20)
data[where(data lt 0.1)]=missing
print,data
print,where(data eq missing)

;CASE #3 (missing is made DOUBLE using DOUBLE fctn. and data is FLOAT):
missing=-9.e+0
missing=double(missing)
data=randomn(-34,20)
data[where(data lt 0.1)]=missing
```

```
print,data
print,where(data eq missing)

;CASE #4 (missing is DOUBLE using "d" exponent and data is DOUBLE using
DOUBLE fctn.):
missing=-9.d+0
data=double(randomn(-34,20))
data[where(data lt 0.1)]=missing
print,data
print,where(data eq missing)
end
```

gives:

```
     -9.00000     -9.00000     -9.00000     -9.00000     -9.00000
0.882273     0.784302     0.505769     0.271391     -9.00000
     -9.00000     -9.00000     -9.00000     -9.00000     -9.00000
0.280262      1.55990     -9.00000     0.456618     -9.00000
           0           1           2           3           4
9          10          11          12          13          14
          17          19
     -9.00000     -9.00000     -9.00000     -9.00000     -9.00000
0.882273     0.784302     0.505769     0.271391     -9.00000
     -9.00000     -9.00000     -9.00000     -9.00000     -9.00000
0.280262      1.55990     -9.00000     0.456618     -9.00000
           0           1           2           3           4
9          10          11          12          13          14
          17          19
     -9.00000     -9.00000     -9.00000     -9.00000     -9.00000
0.882273     0.784302     0.505769     0.271391     -9.00000
     -9.00000     -9.00000     -9.00000     -9.00000     -9.00000
0.280262      1.55990     -9.00000     0.456618     -9.00000
           0           1           2           3           4
9          10          11          12          13          14
          17          19
      -9.0000000      -9.0000000      -9.0000000      -9.0000000
-9.0000000      0.88227320      0.78430194      0.50576895
      0.27139145      -9.0000000      -9.0000000      -9.0000000
-9.0000000      -9.0000000      -9.0000000      0.28026229
       1.5599000      -9.0000000      0.45661816      -9.0000000
           0           1           2           3           4
9          10          11          12          13          14
          17          19
```