
Subject: Re: Faster way ?

Posted by [JD Smith](#) on Tue, 28 Sep 2004 15:49:36 GMT

[View Forum Message](#) <> [Reply to Message](#)

On Mon, 27 Sep 2004 20:46:14 +0000, Dick Jackson wrote:

```
>
> "Craig Markwardt" <craigmnet@REMOVEcow.physics.wisc.edu> wrote in message
> news:onr7onip35.fsf@cow.physics.wisc.edu...
>> rats@mail.geog.uvic.ca (Rafael Loos) writes:
>>> Hi, I am trying to find the number of values that are within a range
>>> ...
>>> I have an Array that has 3 columns and 5 millions lines. Thats what I
>>> am doing ...
>>>
>>> number = WHERE((Array[1,*] GE Min) AND (Array[1,*] LE Max), geralX)
>>>
>>> I am storing the number inside the variable geralX ... It is taking
>>> 0.23 seconds ... but I want to know if there is a faster way to find
>>> that ...
>>
>> If you are doing this many times in a loop and ARRAY is unchanging, it
>> may be worth extracting ARRAY[1,*] into its own variable. That way, you
>> will save the time of extracting each iteration.
>>
>> If you just want the total number of elements that match your filter,
>> you can use total, as in:
>>
>> filter = (Array[1,*] GE Min) AND (Array[1,*] LE Max) geralX =
>> total(filter)
>
> Even with the two uses of Array[1,*], I got 30-40% time reduction with
> this:
>
> array1 = Array[1,*]
> number = WHERE((Array1 GE Min) AND (Array1 LE Max), geralX)
>
> ... and then splicing in your method gave a total of about 45% time
> reduction:
>
> array1 = Array[1,*]
> geralX = Total((Array1 GE Min) AND (Array1 LE Max))
```

It may not be directly relevant to this problem, but if you only care about whether **any** values match the filter (i.e. geralX gt 0) then you can use:

```
geralX = ~array_equal((Array1 GE MinVal) AND (Array1 LE MaxVal),0b)
```

which offers some slight gains (though not as much as you'd think: most the time is spent on the comparison operations). By the way, it's not fair to precompute min/max for HISTOGRAM outside of the time accounting. When you move it back in, I get:

Original Method (msec)	651.46804
Histogram Method (msec)	87.692976
Where Method (msec)	211.58504
Total Method (msec)	95.319033
Array_Equal method (msec)	86.041927

which depends somewhat on how quickly ARRAY_EQUAL finds a non-complying value (and can therefore abort). Another testament to the heavy internal optimization of HISTOGRAM.

JD
