Subject: Re: large array
Posted by Wolf Schweitzer on Wed, 13 Oct 2004 16:19:33 GMT
View Forum Message <> Reply to Message

Dick Jackson wrote:

> L64
> By default, the result of WHERE is 32-bit integer when possible, and 64-bit
> integer if the number of elements being processed requires it. Set L64 to
> force 64-bit integers to be returned in all cases.

Thank you for this tip! Yes, I could get a sensible response from IDL
this time.

> Note
> Only 64-bit versions of IDL are capable of creating variables requiring a
> 64-bit result. Check the value of !VERSION.MEMORY_BITS to see if your IDL is
> 64-bit or not.

Yes, it's 64-bit.

--
Ben Tupper wrote:

> IDL> help, !Version,/str
> ** Structure !VERSION, 8 tags, length=76, data length=76:
> ARCH STRING 'ppc'
> OS STRING 'darwin'
> OS_FAMILY STRING 'unix'
> OS_NAME STRING 'Mac OS X'
> RELEASE STRING '6.1'
> BUILD_DATE STRING 'Jul 14 2004'
> MEMORY_BITS INT 32
> FILE_OFFSET_BITS
> INT 32

OK, this reflected that I use an AIX 64-bit version of IDL 6.1.

> (2) If you are fortunate enough to have 64-bit access then you can use
> the /L64 keyword to WHERE, HISTOGRAM, etc. so that 64-bit integers are
> returned.

Thanks! I have found that the /L64-keyword is the key for these functions.

> >> However, the visualisation of slice subscripts to this array later
> >> does not display any interesting information; instead, the images of
> >> these large data files look different each time and they do not

> >> reflect the content of the data.
> >>
>
> Could you explain that again?

Here is what I meant:

Having read an integer image array of the dimensions ~[2048,2048,900] using the assoc function from a ~7 GB file, showing a slice such as TV (imagearray[*,*,12]), or [*,*,800] results in some type of noise, garbage, or however you want to put it. It looks off. Some of the slice images look like mixed byte order, some are simply noisy, and many are empty.

And it is not a windowing, byte order, variable type, tvscl, or file reading problem - I have smaller data sets of the same type that I can process perfectly to show that this is not the problem. It's the file size with certainty.

Here is what I did:

Encouraged and inspired that I am IN FACT using a system that DOES support large arrays "for the most part", I have established that my version of IDL 6.1 / 64-bit CAN handle integer arrays of, say, 2000 x 2000 x 900 items.

So, if it is not the variable itself - since that 'can be handled', it must be the variable set-up which happens when I read the volume data.

Now it seems that each time I try to read my 7.0 GB raw volume data file using READU or ASSOC, there must be problems transferring data into an array of >2 GB size, because any subscript later does not yield sensible results but garbage.

This is what I did:

imagearray = assoc (.., intarr[2000,2000,900], ..)
imagearray = temporary (imagearray [0])
...

Visualizing slices out of the imagearray variable does not yield sensible results. All of the slice image subscripts look simply off (as explained above).

I checked the integrity of the file ITSELF by reading just a couple of slices out of it into a small variable - such as just grabbing 400 MB -, and the file is a perfect scan. Just large.

Then I tried readu. This did not work at all but caused a crash each time:

```
...
readu, imagearray
...
```

So, I figured that large files can not be read in a single read process,
so I tried this:

```
chunk=intarr (2000,2000,coupleofslices)

imagearray = intarr (2000,2000,900) ;--- A HUGE ARRAY!!!!!

...
point_lun, lun, ... long64(2000) * 2000 * slicecount
            ;-- position anywhere inside file using a L64 variable

readu, chunk   ;--- only read small portion into memory

imagearray [*,*,slicecount:slicecount+coupleofslices-1] = chunk

  or

imagearray [*,*, 0:coupleofslice-1] = chunk
...
```

This works very well. I can use imagearray as a huge array and still
visualize whatever I want with it - filling this variable is the
problem. All slice images subscripts of the type imagearray [*,*,slice#]
work very well and without 'garbage' results as I experienced previously.

It appears to me, that IDL may not handle the READU or ASSOC function
well with these large variables.

Is there a special way to handle these options using very large files?
According to the handbook, there are no further options to be considered.

Do I need to run AIX off harddisk partitions that are formatted using
the Large File Option - or is it enough to read the volume data from
Large File enabled partitions (which is what I am doing now)? Does IDL
try to create 'copies' of files in some temp-folder - then it would be
important for these temp-files to also be large file enabled.

Apparently, applications written for / compiled under AIX 4.2 and

earlier could only handle up to 2GB chunks - despite a filesystem being able to store larger files. In IDL, !version.file_offset_bits returns '64'.

Since I can successfully read a smaller portion of the file, and then store it into the large array that I have set up, I probably just have to to rewrite this file-reading routine and then acquire the file-data in chunks. I still wonder, what keeps 'assoc' and 'readu' to read these large files.

Thanks for the valuable tips!

Wolf.

---