source: http://www.MSSQLTips.com/tip.asp?id=3210 -- printed: 5/6/2014 11:39:25 PM

Validate File Properties with PowerShell Prior to Import into SQL Server

Written By: Tim Smith -- 5/6/2014

Problem

We recently faced an issue with an internal data provider where we imported a flat file at 1AM, but in the past few months, the data provider began providing an incomplete or inaccurate file because it lacked the proper length (or didn't exceed a certain size). We knew that if the file had over two million lines, it was legitimate, yet incomplete files would appear with only a few thousand lines, causing the import process to start which led the automated morning system to "think" it had imported the data for the day. Because of the overnight process, no developer was around to recognize the problem and by morning, production was ready to go with an incomplete data set.

Solution

We could offer a couple of solutions outside of coding a solution: (1) have someone check everything early before production starts or (2) let go of the data provider (for large organizations, this often won't work - startups have a huge advantage here). In our case, we implemented a check on the file before importing. This helped in our case because the file size limited the **latest time** that the file could be imported (4 AM was too late, it had to be started by 3 AM), but consider that this may not be an issue for you, so you could re-loop a check, or call an SSIS package (with a script check) again.

For this solution, we can perform file checks and ensure that we're importing the correct file. In the case of the data provider, if the file exceeded a certain line length or a file size, it was a legitimate file. Some situations may call for other checks and these checks should happen before an import and either (1) stop the import process and notify developers (as customers need to know why daily data are not present), or (2) call the SSIS package or a SQL Server Agent Job again later. For this example, we will use the file inflationdata.csv; note that any file can be checked by changing the file path.

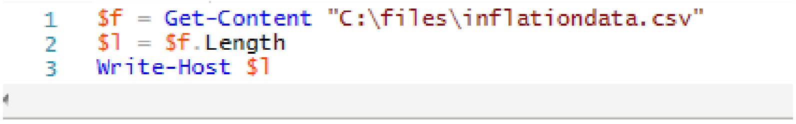
In the case of file length, in a job step before a PowerShell import, we can measure the file line length; or as in the below code, we can wrap an import around an IF statement. The PowerShell code check is below:

```
$f = Get-Content "C:\files\inflationdata.csv"
$l = $f.Length

if ($l -gt 120000)
{
 ## Import function
}
```

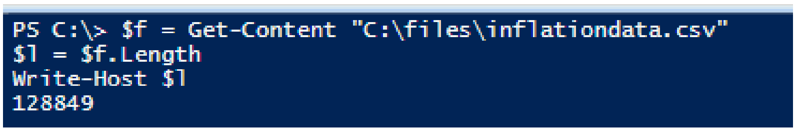
A C# check, which can be altered if using an SSIS script task, is below:

```
long cnt = File.ReadLines(@"C:\files\inflationdata.csv").Count();
if (cnt > 120000)
{
    // Import

}
```

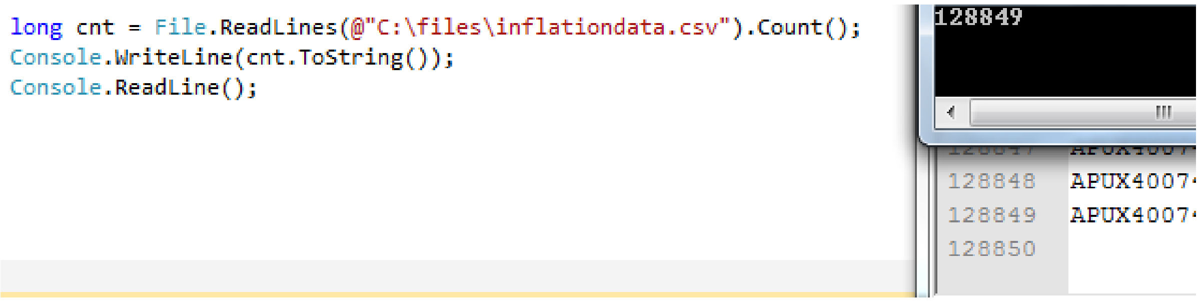
Screenshots of testing (PS Write-Host or C# Console.WriteLine help in testing before wrapping imports):



```
$f = Get-Content "C:\files\inflationdata.csv"
$l = $f.Length
Write-Host $l
```



```
PS C:\> $f = Get-Content "C:\files\inflationdata.csv"
$l = $f.Length
Write-Host $l
128849
```



```
long cnt = File.ReadLines(@"C:\files\inflationdata.csv").Count();
Console.WriteLine(cnt.ToString());
Console.ReadLine();
```

```
128849
```

```
128848   APUX4007
128849   APUX4007
128850
```

As a note here, another approach would be to measure the size of the file. For instance, if the file exceeds a certain amount of MB/GB, then import. In our case, we knew that if the file was over a certain amount of lines, it was valid and could be imported. For a separate process, though, we also used the below approach because one of our files had to exceed 2.7GB before we knew it was valid and obtaining the file size was the faster approach. The below code shows this:

```
$x = Get-ChildItem "C:\files\inflationdata.csv" | Measure-Object -Sum length
## Measures MB
if (($x.Sum / 1MB) -gt 3)
## Measures GIG
##if (($x.Sum / 1GB) -gt 3)
{
    ## Import function
}
```

```
FileInfo fl = new FileInfo(@"C:\files\inflationdata.csv");
// MB (fl.Length is measured in bytes, so for MB use 1024 squared):
double mb = (fl.Length / 1048576.00);
//// GIG for large files (use 1024 cubed):
//double gig = (fl.Length / 1073741824.00);
if (mb > 3)
{
 // Import
}
```

This solution eliminated the problem and with automation we were able to notify customers early if data from the data provider wouldn't appear for that day. In large scale operations where we may not have the choice to switch data providers, we definitely want to alert our customers if their is an error.

Next Steps

- If you need to perform checks on file imports, wrap imports with necessary checks.
- Test file line length and file size.