Command Reference: WriteTableToDelimitedFile()

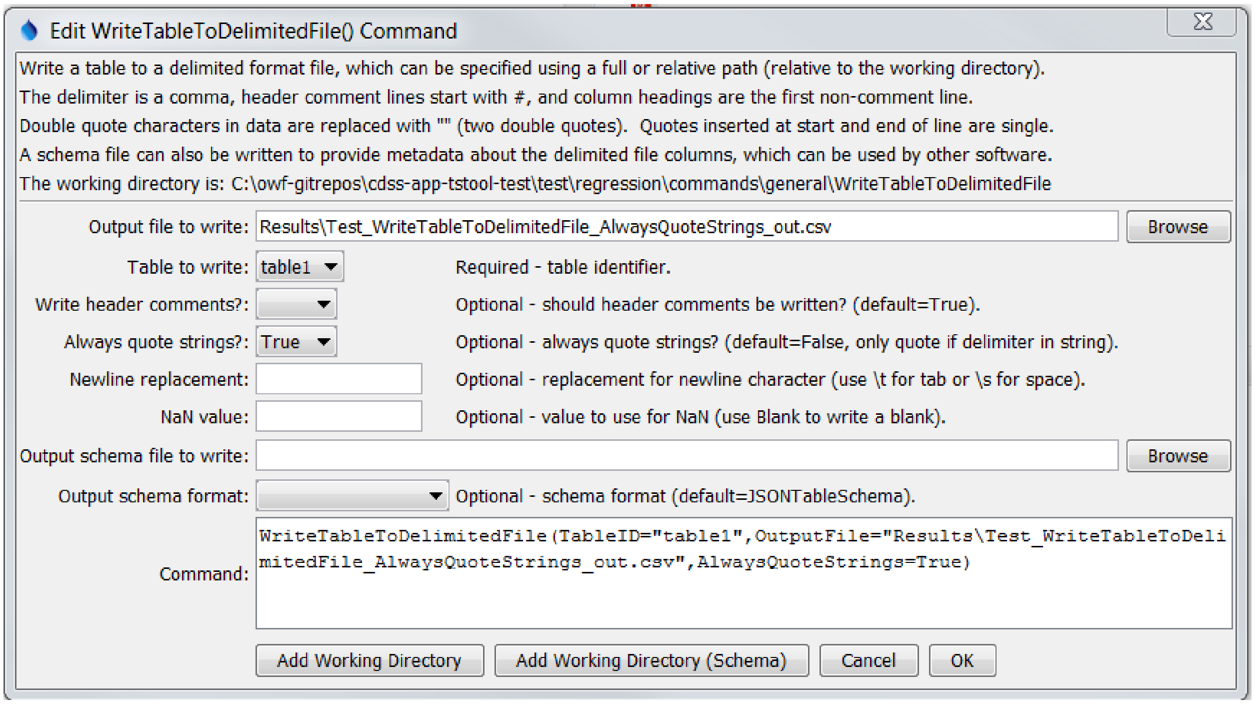
Write a table to a delimited file

Version 11.13.00, 2017-01-19

The `WriteTableToDelimitedFile()` command writes a table to a delimited file. Currently only the comma is supported as the delimiter. This command is the analog to the `ReadTableFromDelimitedFile()` command. It can be used to provide tabular data to other programs, such as spreadsheet programs and geographic information systems.

The default is to write a standard file header using comment lines that start with the # character. If available, column names will be written in double quotes as the first non-comment row. Formatting for cell values is limited and the default precision of floating point numbers may include too many digits – this will be addressed in future updates.

The following dialog is used to edit the command and illustrates the syntax for the command.



Write a table to a delimited format file, which can be specified using a full or relative path (relative to the working directory).
The delimiter is a comma, header comment lines start with #, and column headings are the first non-comment line.
Double quote characters in data are replaced with "" (two double quotes). Quotes inserted at start and end of line are single.
A schema file can also be written to provide metadata about the delimited file columns, which can be used by other software.
The working directory is: C:\owf-gitrepos\cdss-app-tstool-test\test\regression\commands\general\WriteTableToDelimitedFile

Output file to write: Results\Test_WriteTableToDelimitedFile_AlwaysQuoteStrings_out.csv

Table to write: table1 — Required - table identifier.

Write header comments?: — Optional - should header comments be written? (default=True).

Always quote strings?: True — Optional - always quote strings? (default=False, only quote if delimiter in string).

Newline replacement: — Optional - replacement for newline character (use \t for tab or \s for space).

NaN value: — Optional - value to use for NaN (use Blank to write a blank).

Output schema file to write:

Output schema format: — Optional - schema format (default=JSONTableSchema).

Command:

```
WriteTableToDelimitedFile(TableID="table1",OutputFile="Results\Test_WriteTableToDelimitedFile_AlwaysQuoteStrings_out.csv",AlwaysQuoteStrings=True)
```

WriteTableToDelimitedFile

WriteTableToDelimitedFile() Command Editor

The command syntax is as follows:

```
WriteTableToDelimitedFile(Parameter=Value,…)
```

Command Parameters

Parameter	Description	Default
`TableID`	Identifier for the table to write. Can be specified using processor `${Property}`.	None – must be specified.
`OutputFile`	The name of the file to write, as an absolute path or relative to the command file location. Can be specified using processor `${Property}`.	None – must be specified.
`WriteHeaderComments`	Indicates whether to write the header comments, `True` or `False`. Some programs, such as Esri's ArcGIS do not handle delimited files with comments.	`True`
`AlwaysQuoteStrings`	Indicates whether values in string columns should always be surrounded by double quotes: • `False` – only quote strings that contain the delimiter • `True` – always quote strings An example of using `AlwaysQuoteStrings=True` is to quote identifiers that have a leading zero (e.g., `01234567`). Not quoting may cause the values to be interpreted as integers when read from the delimited file.	`False`
`NewlineReplacement`	The string to replace newlines in string values, necessary to prevent unexpected line breaks in output rows. In order to handle newlines from various systems, the following patterns are replaced in sequence: • `\r\n` • `\n` • `\r` The following special parameter values are recognized: • `\t` – replace newline with tab • `\s` – replace newline with space	Do not replace newlines (file format may be unusable because of unexpected line breaks).
`NaNValue`	The value to write for `NaN` data values. Specify `Blank` to write a blank (empty string).	`NaN`
`OutputSchemaFile`	Name of schema file to write, useful to help other software understand contents of the delimited file. See the `OutputSchemaFormat` parameter.	Do not create schema file.
`OutputSchemaFormat`	Schema format, one of the following, output is limited but will be expanded in the future: • `JSONTableSchema` – see http://specs.frictionlessdata.io/json-table-schema • `GoogleBigQuery` – see https://cloud.google.com/bigquery/docs/reference/rest/v2/tables	`JSONTableSchema`