Command Reference: TableToTimeSeries()

Create time series from a table

Version 11.03.05, 2015-06-14

The `TableToTimeSeries()` command creates time series from a table. This command can be used when a command to read time series from a specific file format or datastore has not been implemented or parsing the table is easier. The table typically is read using one of the following commands:

- `ReadTableFromDataStore()` – for example, define an ODBC DSN connection to a database and query time series using an SQL statement.
- `ReadTableFromDelimitedFile()` – for example, read time series from a comma-separated-value (CSV) file.
- `ReadTableFromExcel()` – for example, read time series from a comma-separated-value (CSV) file
- `ReadTableFromHTML()` – envisioned for the future.
- `ReadTableFromXML()` – under development.

TSTool internally represents tables as a collection of columns, where a column contains values of a consistent data type (e.g., integer, string, double).   A time series table requires at a minimum a date/time column (or separate date and time columns), at least one data value column, and optionally one or more columns for data flags. Data represented in three table designs are handled by this command:

- Data for multiple locations/series stored in a single column (common in a database or stream of data from a data logger) – specify the `LocationColumn` command parameter – **referred to as single column format in this command**.
- Data for multiple locations/series stored in multiple columns (common in spreadsheets and CSV files) – do not specify the `LocationColumn` command parameter but instead specify the `ValueColumn` and optionally `LocationID` parameters – **referred to as multiple column format in this command.**
- Data for one or multiple stations represented in a block of data with time on both axes (e.g., year in first column and months in other columns) – **this is referred to as block format and is under development**.

The command provides flexibility to specify time series metadata (e.g., data source, units) as command parameters, or read from the file. However, this flexibility is limited by practical considerations in supporting likely data formats. One current limitation of the command is that TSTool does not determine table column names during discovery mode (discover mode is a partial command run that allows data such as time series and table identifiers to be provided to later commands for editing). Consequently, although this command will create time series when run, it does not produce time series information in discovery mode and the time series will not be listed in later command editors. This limitation may be addressed in future TSTool updates.

Example 1: Single Column for Time Series Values with Time Series Identifier Information Specified by Command Parameters

An example of a table with single data value column with flags is shown in the following figure (note that a column is used for the location identifier and that the location is different for the topmost and bottommost records).

TableToTimeSeries_Single_DataTable

Simple Table with Data Values in a Single Column

In the above example, the list of unique time series is determined by examining the location column contents. Other time series metadata such as data source and units can be assigned using the `DataSource`, `Units`, and similar parameters.

The following dialog is used to edit the command and illustrates the command syntax when processing single-column data from the above example. Note that time series metadata are specified with command parameters.
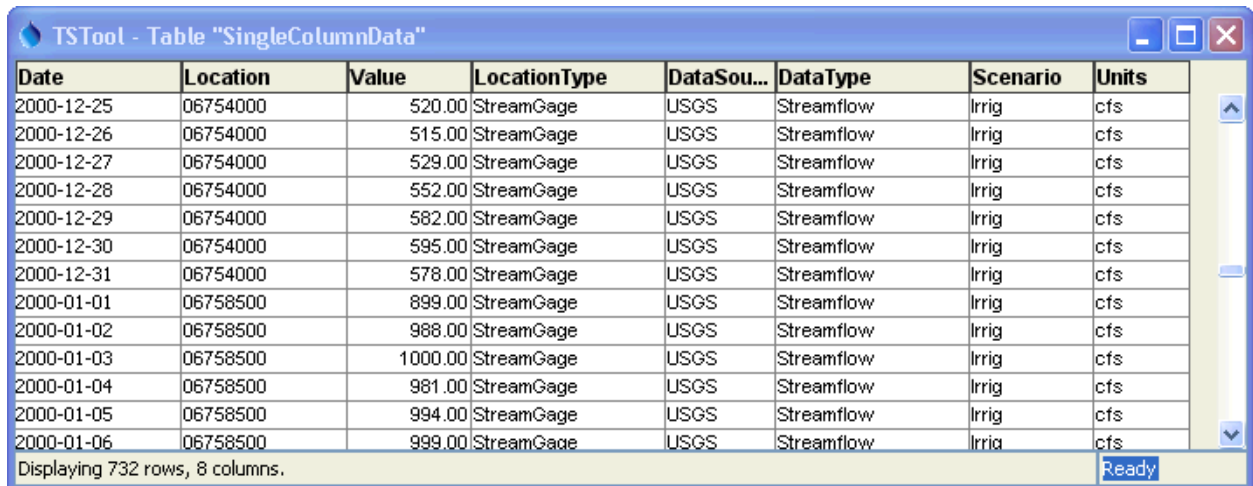

TableToTimeSeries() Command Editor for Table with Data in a Single Column

TableToTimeSeries() Command Editor Data Parameters

Example 2: Single Column for Time Series Values with Time Series Identifier Information Specified in Table

The following example is also treated as single-column because a single column of data values is present. However, metadata are taken from other columns. This data format is consistent with a database query where several tables have been joined together. Although not efficient because time series metadata is repeated for every row, the format is convenient for data translation. Use the DataSourceColumn, UnitsColumn and similar parameters to specify metadata. The unique list of time series will be determined from the combinations of location identifier and other metadata.
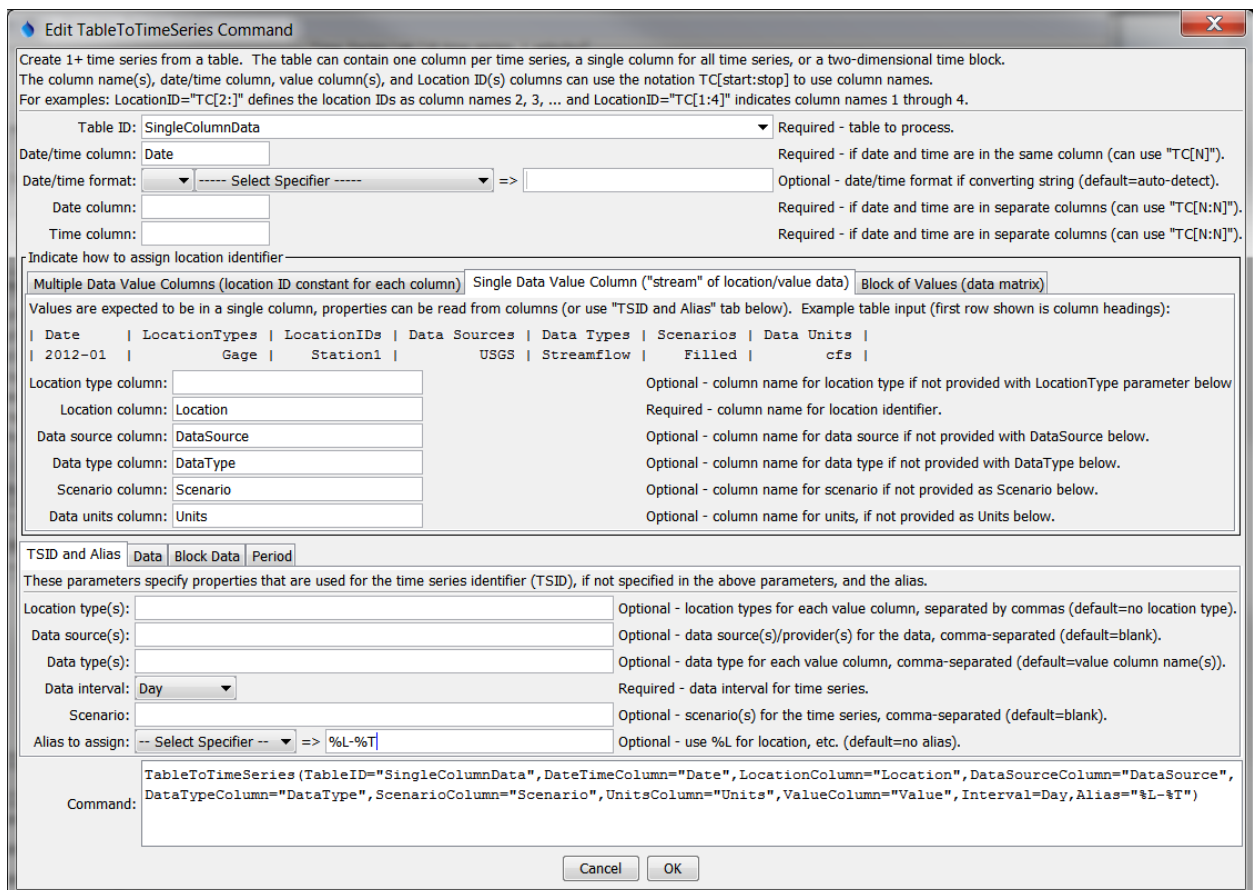
TableToTimeSeries_SingleMeta_DataTable

Table with Data Values in a Single Column and Metadata Provided in Other Columns

The following dialog is used to edit the command and illustrates syntax when processing single-column data from the above example. Time series metadata are specified with command parameters. The `ValueColumn` parameter in the **Data** tab is specified as "Value".

TableToTimeSeries_SingleMeta

TableToTimeSeries() Command Editor for Table with Single Data Column and Metadata Columns

Example 3: Multiple Columns for Time Series Values with Time Series Identifier Information Specified in Command Parameters

An example of multi-column data with flags for each time series is shown in the following figure:

TableToTimeSeries_Multiple_DataTable

Table with Multiple Data Columns

The following dialog is used to edit the command and illustrates the syntax for the command when processing multi-column data from the above table.

TableToTimeSeries_Multiple

TableToTimeSeries() Command Editor For Table with Data in Multiple Column

TableToTimeSeries_Multiple_Data

TableToTimeSeries() Command Editor For Table with Data in Multiple Column, Data Tab

Example 4: Time Series Values Specified in a Block

The following data example illustrates monthly time series values specified in a block.

TableToTimeSeries_Block_DataTable

Table with Block Data Format

The following dialog is used to edit the command and illustrates the syntax for the command when processing block data from the above table.

TableToTimeSeries() Command Editor For Table with Data in Block Format

TableToTimeSeries() Command Editor For Table with Data in Block Format – Location ID

TableToTimeSeries() Command Editor For Table with Data in Block Format – Layout Parameters

TableToTimeSeries_Block_Block

TableToTimeSeries() Command Editor For Table with Data in Block Format – Block Parameters

The command syntax is as follows:

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

Command Parameters

Parameter	Description	Default
TableID	The identifier for the table to read. Can be specified using processor `${Property}`.	None – must be specified.
DateTime Column	The column for date/time, when date and time are in one column. If the table was read in a way that the column type is "date/time", then the values are used directly. If the table was read in a way that the column type is "string", then the string is parsed using default logic or the `DateTimeFormat` parameter if specified.	Required if `DateColumn` is not specified.
DateTime Format	The format for date/time strings in the date/time column, if strings are being parsed. If blank, common formats such as `YYYY-MM-DD hh:mm` and `MM/DD/YYYY` will automatically be detected. However, it may be necessary to specify the format to ensure proper parsing. This format will be used to parse date/times from the `DateTimeColumn` or the merged string from the `DateColumn` and `TimeColumn` (if specified). The format string will depend on the formatter type. Currently, only the "C" formatter is available, which uses C programming language specifiers. The resulting format includes the formatter and specifiers (e.g., `C:%m%d%y`).	Will automatically be determined by examining date/time strings.
DateColumn	The name of column that includes the date, used when date and time are in separate columns.	Required if `DateTimeColumn` is not specified.
TimeColumn	The name of column that includes the time, used when date and time are in separate columns. If both `DateColumn` and `TimeColumn` are specified, their contents are merged with a joining colon character and are then treated as if `DateTimeColumn` had been specified.	Required if `DateColumn` is specified and the interval requires time.

Parameter	Description	Default
LocationID	**Used with multiple data column table**. The location identifier(s) to assign to time series, separated by columns if more than one column is read from the table. Column names can be specified as literal strings or as TC[start:stop] to match table column names, where start is 1+ and stop is blank to read all columns or a negative number to indicate the offset from the end column. Can be specified using processor ${Property}.	None – must be specified for multiple column data tables.
LocationType Column	**Used with single data column table**. The name of the column containing the location type.	Do not assign a location type.
LocationColumn	**Used with single data column table**. The name of the column containing the location identifier.	None – must be specified for single column data tables.
DataSource Column	**Used with single data column table**. The name of the column containing the data source.	Use the DataSource parameter, which can be blank.
DataType Column	**Used with single data column table**. The name of the column containing the data type.	Use the DataType parameter, which can be blank.
ScenarioColumn	**Used with single data column table**. The name of the column containing the scenario.	Use the Scenario parameter, which can be blank.
UnitsColumn	**Used with single data column table**. The name of the column containing the data units.	Use the Units parameter, which can be blank.
LocationType	The location type(s) to assign to time series for each of the value columns (or specify one value to apply to all columns).	No location type will be assigned.
DataSource	The data source (provider) identifier to assign to time series for each of the value columns (or specify one value to apply to all columns).	No data source will be assigned.
DataType	The data type to assign to time series for each of the value columns (or specify one value to apply to all columns).	Use the value column names for the data types.
Interval	The interval for the time series. Only one interval is recognized for all the time series in the table. Interval choices are provided when editing the command. If it is possible that the date/times are not evenly spaced, then use the Irregular interval (this is difficult to do for multiple data column tables).	None – must be specified.
Scenario	The scenario to assign to time series for each of the value columns (or specify one value to apply to all columns).	No scenario will be assigned.
Alias	The alias to assign to time series, as a literal string or using the special formatting characters listed by the command editor. The alias is a short identifier used by other commands to locate time series for processing. Can be specified using processor ${Property}.	No alias will be assigned.

Parameter	Description	Default
ValueColumn	The name(s) of column(s) containing data values. Separate column names with commas. The `TC[start:stop]` notation discussed for `LocationID` can be used. Only one column should be specified for single data column table.	None – must be specified.
FlagColumn	The name(s) of column(s) containing the data flag. Separate column names with commas. The `TC[start:stop]` notation discussed for `LocationID` can be used. If specified, the number of columns must match the `ValueColumn` parameter, although specifying blank column names is allowed to indicate that a value column does not have a corresponding flag column..	Flags are not read.
Units	The data units to assign to time series for each of the value columns (or specify one value to apply to all columns).	No units will be assigned.
Missing	Strings that indicate missing data in the table (e.g., "m"), separated by commas.	Interpret empty column values as missing data.
Handle DuplicatesHow	Indicate how to handle duplicate date/time values in the table: • `Add` – add the duplicate values (missing values are ignored) • `UseFirstNonmissing` – set the output to the first non-missing value • `UseLast` – set the output to the last value processed, even if missing • `UseLastNonmissing` – set the output to the last non-missing value processed	UseLast
BlockLayout	Indicates how data are laid out when in block format: • `Period` – a single block is used for the entire period	Block layout is not used.
BlockLayout Columns	Indicates the time slice for values in columns: • `Month` – each column includes a month	
BlockLayout Rows	Indicates the time slice for values in rows: • `Year` – each row includes a year	
BlockOutput YearType	Indicates the year type for the data block. For example, if columns are in rows and the output year type is Water, then the first value column is October. • `Calendar` – January to December • `NovToOct` – November to October • `Water` – October to September	Calendar
InputStart	The date/time to start reading data. Can be specified using processor `${Property}`.	All data or global input start.
InputEnd	The date/time to end reading data. Can be specified using processor `${Property}`.	All data or global input end.