Command Reference: TableToTimeSeries()

Create time series from a table

Version 10.21.00, 2013-06-27

Note: This command may be migrated to two separate commands (one for single column data values and one for multiple column data values) if editing the command becomes confusing.

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. The table typically is read using one of the following commands:

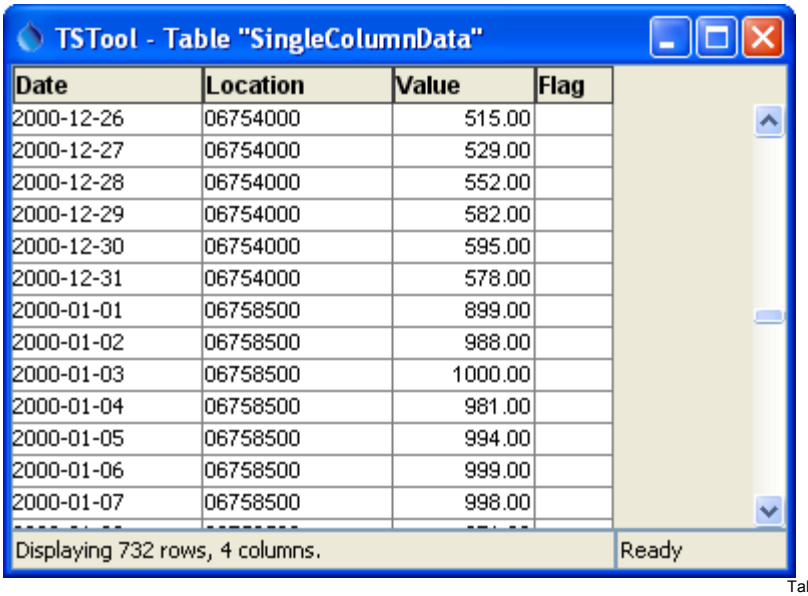
- `ReadTableFromDataStore()` – for example, define an ODBC DSN connection to a database and query time series using an SQL statement.
- `ReadTableFromDelimitedFile()` – for example, read time series from a comma-separated-value (CSV) file.
- `ReadTableFromExcel()` – for example, read time series from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

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 one of two 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.
- 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.

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 will be addressed in future TSTool updates.

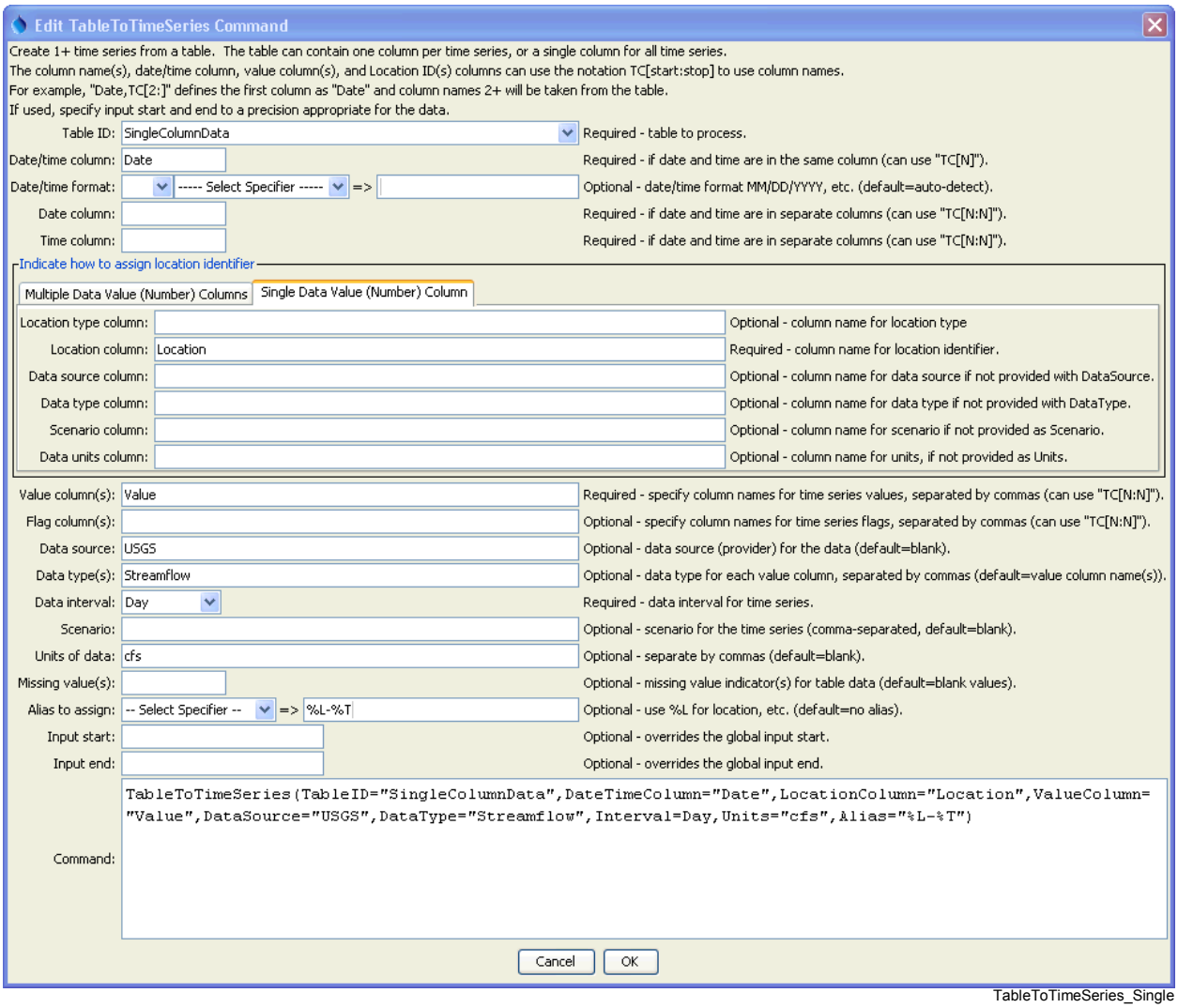
An example of a table with single data value column with flags is shown in the following figure (note that a column is used for the location identifier and that the location is different for the topmost and bottommost records).



TableToTimeSeries_Single_Data

Simple Table with Data Values in a Single Column

In the above example, the list of unique locations 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_Single

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

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. In this case, use the `DataSourceColumn, UnitsColumn` and similar parameters for single column input, and the command will initialize each time series with metadata from the row where a new location ID is detected.

TSTool - Table "SingleColumnData"

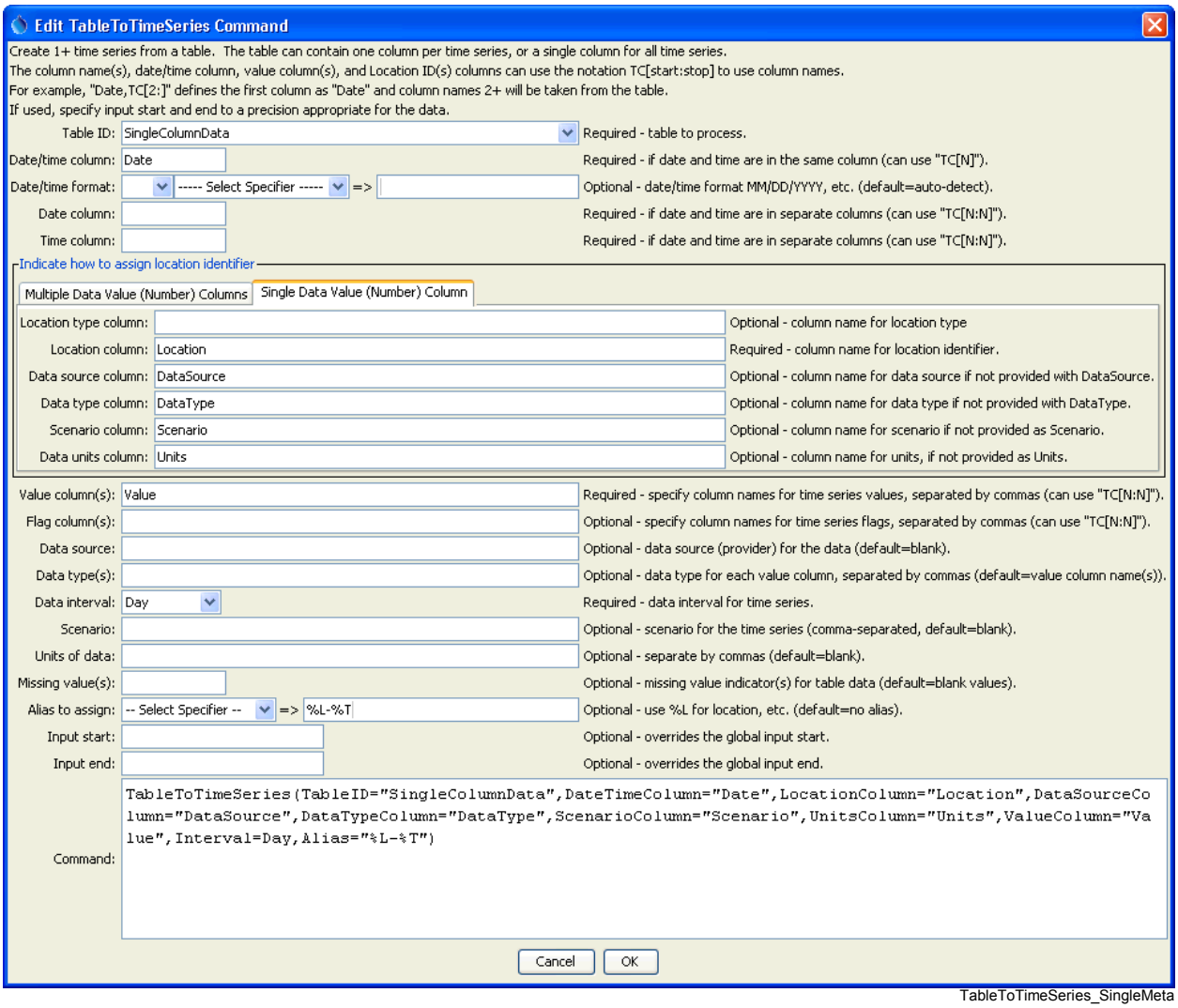
Date	Location	Value	LocationType	DataSou...	DataType	Scenario	Units
2000-12-25	06754000	520.00	StreamGage	USGS	Streamflow	Irrig	cfs
2000-12-26	06754000	515.00	StreamGage	USGS	Streamflow	Irrig	cfs
2000-12-27	06754000	529.00	StreamGage	USGS	Streamflow	Irrig	cfs
2000-12-28	06754000	552.00	StreamGage	USGS	Streamflow	Irrig	cfs
2000-12-29	06754000	582.00	StreamGage	USGS	Streamflow	Irrig	cfs
2000-12-30	06754000	595.00	StreamGage	USGS	Streamflow	Irrig	cfs
2000-12-31	06754000	578.00	StreamGage	USGS	Streamflow	Irrig	cfs
2000-01-01	06758500	899.00	StreamGage	USGS	Streamflow	Irrig	cfs
2000-01-02	06758500	988.00	StreamGage	USGS	Streamflow	Irrig	cfs
2000-01-03	06758500	1000.00	StreamGage	USGS	Streamflow	Irrig	cfs
2000-01-04	06758500	981.00	StreamGage	USGS	Streamflow	Irrig	cfs
2000-01-05	06758500	994.00	StreamGage	USGS	Streamflow	Irrig	cfs
2000-01-06	06758500	999.00	StreamGage	USGS	Streamflow	Irrig	cfs

Displaying 732 rows, 8 columns. Ready

TableToTimeSeries_SingleMeta_Data

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

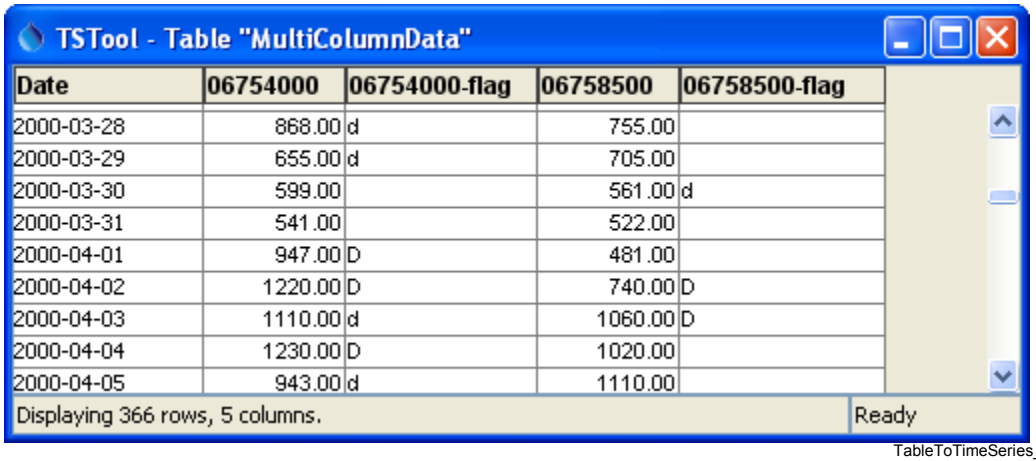
The following dialog is used to edit the command and illustrates the command syntax when processing single-column data from the above example. Note that time series metadata are specified with command parameters.



TableToTimeSeries_SingleMeta

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

An example of multi-column data with flags is shown in the following figure, where each time series has its own data and flag columns:

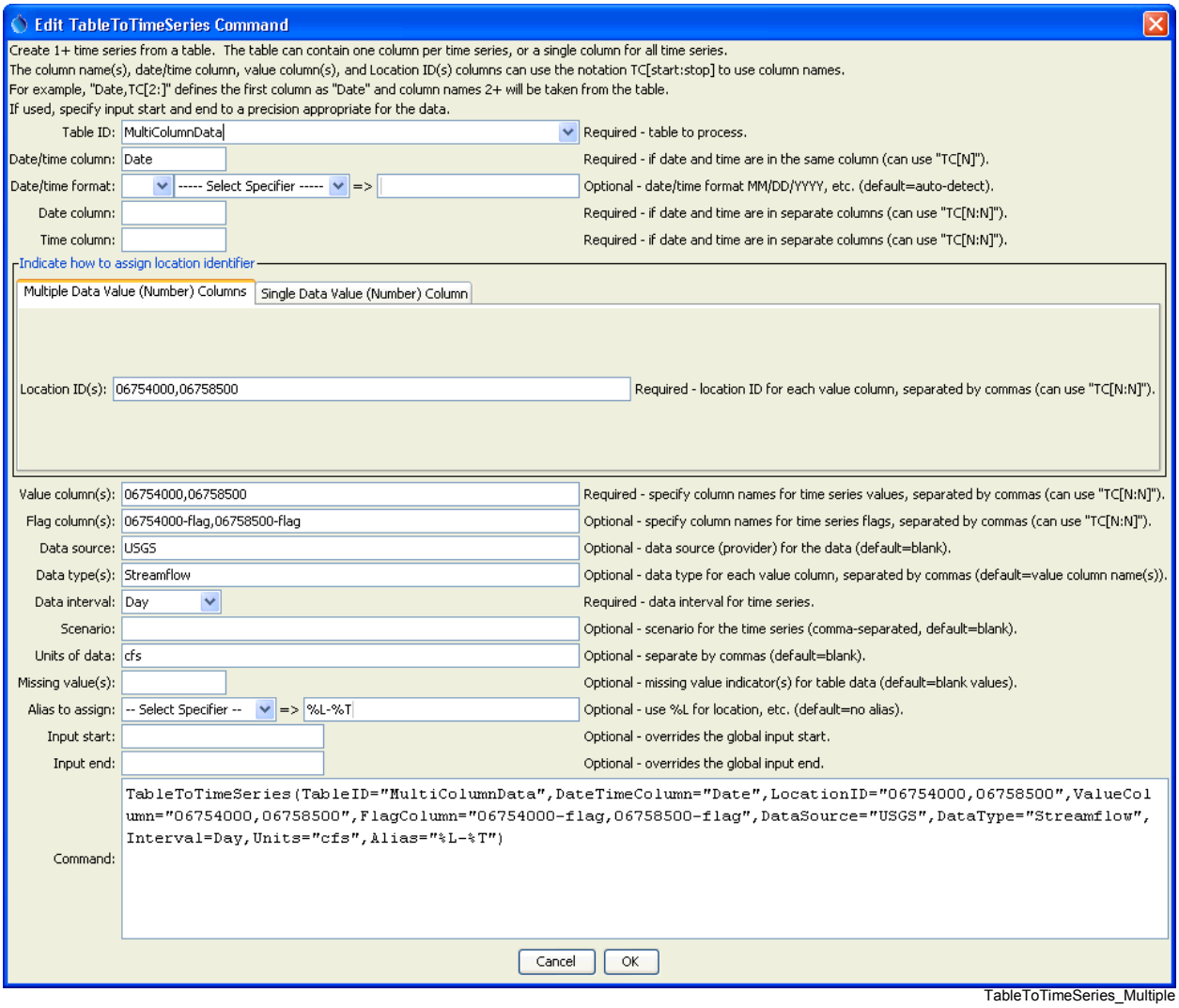


Date	06754000	06754000-flag	06758500	06758500-flag
2000-03-28	868.00	d	755.00	
2000-03-29	655.00	d	705.00	
2000-03-30	599.00		561.00	d
2000-03-31	541.00		522.00	
2000-04-01	947.00	D	481.00	
2000-04-02	1220.00	D	740.00	D
2000-04-03	1110.00	d	1060.00	D
2000-04-04	1230.00	D	1020.00	
2000-04-05	943.00	d	1110.00	

TableToTimeSeries_Multiple_Data

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 a Single Column

The command syntax is as follows:

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

Command Parameters

Parameter	Description	Default
`TableID`	The identifier for the table to read.	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	Required if `DateColumn` is not specified.

Parameter	Description	Default
	`DateTimeFormat` parameter if 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.
`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.	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.
`ValueColumn`	The name(s) of column(s) containing data values.	None – must be

Parameter	Description	Default
	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.	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.
`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 provider 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.
`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.
`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.	No alias will be assigned.
`InputStart`	The date/time to start reading data.	All data or global input start.
`InputEnd`	The date/time to end reading data.	All data or global input end.