

1010 data

Tendo User Guide
Version 1

0 1 1 1 0 1 1 1 1 1 1 0 1
0 1 1 0 0 1 0 0 1 1 0 1 1
0 1 1 1 1 0 0 0 1 1 1 0 0
1 1 0 1 0 1 0 0 1 0 1 1 0
0 1 1 1 1 0 0 0 1 0 0 0 1
1 0 0 1 0 0 1 1 1 0 1 0 1
0 0 1 0 1 0 0 1 1 1 1 1 0
1 1 0 1 1 0 1 1 1 1 0 0 1
0 0 0 0 0 0 0 1 0 0 1 1 1
0 1 1 1 1 0 0 1 0 1 0 1 0
1 0 1 1 0 1 1 0 1 1 0 0 1
0 0 0 0 0 0 1 0 0 1 0 1 1
0 0 1 0 0 1 0 0 1 1 0 0 0
1 1 0 0 1 1 0 1 0 1 1 1 1
0 1 1 0 1 0 1 1 0 0 0 0 0
0 0 0 1 0 0 1 0 0 1 1 0 1
0 0 0 1 0 0 1 0 0 1 1 1 1
0 0 0 0 0 0 1 0 1 0 1 0 1
1 1 1 0 0 0 1 1 0 1 1 0 0
0 0 1 0 1 1 0 1 0 1 0 0 1
0 1 0 0 1 0 **1 0 1 0** 0 0 1
1 1 0 0 1 0 0 0 1 0 0 1 1
0 1 0 1 1 0 1 1 0 1 0 0 0
0 0 1 0 1 1 0 1 0 1 1 0 0
0 1 1 1 0 1 0 0 0 0 1 0 0
0 1 1 1 0 1 0 1 0 0 0 1 1
1 0 1 0 0 0 0 1 1 1 0 0 1
0 0 1 0 0 0 1 1 1 1 1 1 1
0 0 1 0 0 1 1 0 1 1 1 0 0
0 1 0 1 0 0 0 1 1 0 1 1 0
0 0 1 1 0 1 0 0 1 0 0 1 0
0 1 0 0 1 0 0 1 1 0 1 1 1
1 0 0 1 0 0 0 1 1 0 0 1 1
1 1 0 0 1 1 1 1 1 1 1 1 1
0 0 0 0 0 1 0 1 1 1 0 1 1
1 0 0 1 0 1 0 1 1 1 1 1 0
1 1 1 1 0 0 1 1 1 0 0 1 1
1 1 1 0 0 0 0 0 0 1 1 0 1
0 0 1 0 0 1 1 0 1 0 1 1 1
0 0 1 1 1 1 0 0 0 0 0 0 0
1 0 0 1 1 0 0 0 0 0 0 1 0

Introducing Tendo

Tendo is a command-line tool designed to provide direct access to the 1010data system. It can quickly execute queries, store results, and perform any data manipulation that 1010data provides.

Table of Contents

Installing	3
Getting Started.....	3
Getting Help	4
Command Format.....	4
Connecting	4
Gateway	4
Password.....	5
Proxy.....	5
Killing prior sessions	5
Running Queries.....	5
Command Line.....	6
Running queries from a file	6
Returning all the data from a table.....	6
1010SQL.....	7
Variable Substitution.....	7
Controlling data output	8
Saving output to a file.....	8
Running multiple queries	9
Formatting Output	9
Custom formatting.....	10
<i>Quoting</i>	10
<i>Delimiting</i>	11
Server-side operations	12
Managing Sessions/Storing Defaults.....	13
Managing Sessions.....	13
Storing Defaults	13
Miscellaneous	14

Installing

Under the Windows operating system, installing Tendo is as simple as running `tendo_w32.exe`. Once installed, Tendo is in the system path and is available within the Windows command shell. For 64-bit Windows operating systems (Windows XP x64 Edition, Windows Server 2003 x64 Edition, etc.) only, you may optionally install a 64-bit version of Tendo by running `tendo_w64.exe`. There are no important differences between the two versions and the 32-bit version will work equally well on a 64-bit operating system.

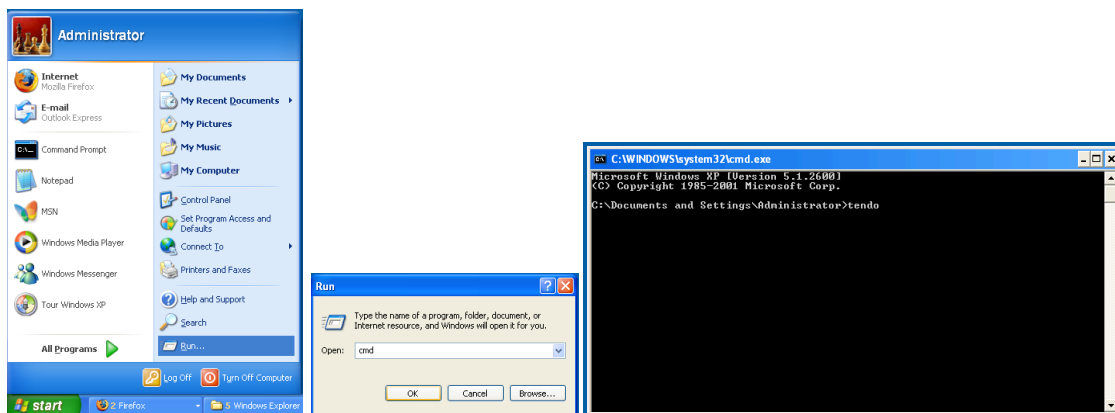
Tendo is also available for the Mac OS X and Linux operating systems (32- and 64-bit). To install Tendo on these operating systems, copy the executable file from the appropriate directory (`mac32`, `mac64`, `linux32`, `linux64`) into the directory of your choice, e.g. `/usr/local/bin`.

Getting Started

Tendo is invoked and passed arguments like any other command-line program.

For this guide, we will use the Windows Shell. Exact keystrokes are set in a [monospace font](#). Optional items are placed in [\[brackets\]](#). User supplied input (such as a username or table name) are in *italics*.

After Tendo is installed, go to Start -> Run. In the box, type `cmd` and hit enter. This should open a terminal in which you can type commands - the Windows Shell. To make sure Tendo was is functioning properly, type `tendo` and hit enter. A set of options should fill the screen. If not, double-check that Tendo was installed correctly.



Getting Help

Quick help is obtained by running `tendo -` and more detailed help can be found with `tendo --`

Command Format

Tendo relies on options passed through the command line.

The basic syntax is `tendo [OPTIONS] [QUERY]` where both options and query format are described below.

Connecting

Tendo ordinarily authenticates (logs in) to 1010data each time it is run. You can also maintain a logged-in session over multiple invocations of Tendo; this is discussed in Managing Sessions.

Minimally a user, password and gateway are required to connect. Note that these can be stored in environment variables, eliminating the need to specify them on the command line. Environment variables are discussed in Storing Defaults.

Gateway

The `-g` option specifies which 1010data gateway to connect to.

The gateway option accepts full 1010data gateway URLs as well as shortcuts (described below). If you use 1010data's GUI via `www.1010data.com`, you can try the following command to view the contents of a short demo table (public weather station data). If you use `www2.1010data.com`, use that host in the URL instead. Remember to replace the username with your own. You will be prompted for a password.

```
tendo -g http://www.1010data.com/cgi-bin/gw.k -u username
pub.demo.weather.stations .
```

Note: the single period at the end of the above example is part of the command line!

Tendo accepts gateway shortcuts, which are less cumbersome than full gateway URLs. Currently these are `www` and `www2`, corresponding to `www.1010data.com` and `www2.1010data.com`. Occasionally 1010data may provide access to a beta system; shortcuts for these are `wwwbeta` and `www2beta`. The following is identical to the command above but uses a shortcut:

```
tendo -g www -u username pub.demo.weather.stations .
```

Password

Normally Tendo prompts for a password each time it authenticates. The `-p` option allows you to put the password on the command line

```
tendo -g www -u username -p secretpass  
pub.demo.weather.stations .
```

Proxy

If you are unable to connect to 1010data using Tendo, your site may require http traffic to pass through a proxy. Tendo allows you to specify a proxy with the `-P` option.

The proxy specification format is `http://USER[:PASSWORD]@HOST[:PORT]` where parts in brackets are optional. For example, to specify a proxy on port 8000 for which a username but no password is required:

```
tendo -g www -u username -P  
http://proxyuser@exampleproxy.com:8000  
pub.demo.weather.stations .
```

Your network administrator should be able to provide you with the correct proxy string.

Killing prior sessions

`-k` kills any existing logged-in session under the given username, allowing Tendo access. This is useful if you are encountering 'already logged in' errors. Note that only one session, whether GUI or API (including Tendo), may be logged in with a given user ID at one time. If, for example, you are logged in via the GUI and use Tendo with the `-k` option, your GUI session will be logged out

Running Queries

Tendo accepts queries on either the command line or as a file (including standard input). Queries come after all options. The table is specified before the macro, as in `tendo TABLE QUERY`

Tables are specified by their full 1010data path, e.g. `pub.demo.weather.stations` as in the earlier example. Queries are written in the 1010data macro language, For more information on the macro language, see the online help in the 1010data GUI.

Command Line

Because we are using the Windows Shell, double quotes in the query must be backslash-escaped. Additionally, the entire query must be enclosed in quotes to prevent spaces from being interpreted:

```
tendo -g www -u username pub.demo.weather.stations "<sel value=\"(state='GA')\"/>"
```

Running queries from a file

Specifying long queries on the command line can be unwieldy. A more convenient method is to place the query in a text file. The file containing a query can be specified using `@filename` in place of QUERY in your Tendo command.

For example, using Notepad (or your favorite text editor), place the following query in a file named `queryfile.txt`. Note that it is not necessary to use backslashes to escape the quotes in this case:

```
<sel value="(state='GA')"/>
```

The query can then be executed from the Windows shell as follows:

```
tendo -g www -u username pub.demo.weather.stations @queryfile.txt
```

A single dash (-) can also be given in place of a filename. Tendo will then take its query from the “standard input” (that is, the console, unless the shell input has been redirected). This allows you to type or paste your query directly into the shell window:

```
tendo -g www -u username pub.demo.weather.stations @-  
(note that you must enter control-Z on Windows or control-D on Mac/Linux to end the input)
```

The `@-` option is also useful for piping query text from other programs; see your shell documentation for more information.

Returning all the data from a table

As we saw in the first example above, you may retrieve a complete table by using a single period (.) in place of a macro-language query. The period serves as the “empty query,” i.e. one with no operations.

1010data Quick Queries are queries saved in the 1010data GUI so that they appear to be tables. When the “table” is opened, the Quick Query is run and the results are used as the basis for further operations. A Quick Query pathname may be specified in place of a physical table in a Tendo command. With the period (.) the entire

output of the Quick Query is returned. If a macro-language query is passed to Tendo, it will be run on the output of the Quick Query.

1010SQL

In addition to the native 1010data macro language, Tendo supports 1010SQL, a dialect of SQL which is translated into macro language. 1010SQL queries are specified as a single argument beginning with the keyword SELECT and ending with a semicolon (;) and the table on which the query is run is included in the query. For example:

```
tendo -g www -u username "select * from
pub.demo.weather.stations;"
```

For more information on 1010SQL, see the 1010ODBC documentation.

Variable Substitution

It is possible to do variable substitution with Tendo. This means that a placeholder can be embedded in a macro query or 1010SQL query which will be replaced by an argument on the command line. Variable substitution is useful to reuse queries with different parameters. Inside the query, a variable to be substituted is delineated by double brackets: `[[target]]`. Note that unlike the rest of this document, the double brackets are literal and both must be there. Multiple substitutions can be specified. If a variable is found that hasn't been specified, it will be left intact with brackets. On the command line, the variable to be replaced is specified as `-[[target]]=replacement`. If the replacement contains spaces, it must be quoted on the shell, as in: `-[[name]]="NEW YORK"`. To run two queries determining the weather stations in two states, one can run:

```
tendo -u username -g www -[[target]]=GA
pub.demo.weather.stations "<sel
value=\"(state='[[target]]')\"/>"
```

and

```
tendo -u username -g www -[[target]]=NY
pub.demo.weather.stations "<sel
value=\"(state='[[target]]')\"/>"
```

Variable substitutions can also be put in a file. One variable declaration is put per line and no dash is needed before the brackets. No quoting is necessary when putting variables in a file. For example, if a file is created called substitutions.txt with the contents:

```
[[target]]=GA (note that the file must end with a blank line)
```

then the same query can be run as above with:

```
tendo -u username -g www -[[target]]=GA
pub.demo.weather.stations "<sel
value=\"(state='[[target]]')\"/>"
```

Controlling data output

Sometimes you may wish to retrieve only some of the results of a query, or even avoid retrieving any data at all. The `-N`, `-n`, and `-i` options allow you to customize how much data you get back.

If you specify the `-N` option, Tendo will not retrieve any rows of data, but instead will output the number of rows available after running the query. For example

```
tendo -g www -u username -N pub.demo.weather.stations .
```

should indicate that the weather station table has 262 rows.

If you wish to limit the number of rows retrieved, use the `-n` option.

```
tendo -g www -u username -n 10 pub.demo.weather.stations .
```

will return 10 rows of data, starting at the first row of the table. The `-n` option specifies the *maximum* number of rows to be retrieved; if you use `-n 10` and a query return 5 rows, for example, then only those 5 rows will be retrieved.

To retrieve data starting at other than the first row of the query results, use the `-i` option. For example

```
tendo -g www -u username -i 253 pub.demo.weather.stations .
```

should return the last 10 rows (253-262) of the weather station table.

You may combine `-i` and `-n` options to retrieve any arbitrary range of data.

Saving output to a file

Much of the time, you will probably want to save the data retrieved by Tendo to a file on your system rather than displaying it on the console. Tendo provides the `-o` option to specify a destination file for the output data, for example:

```
tendo -g www -u username pub.demo.weather.stations
@queryfile.txt -o queryout.txt
```

Without the `-o` option, Tendo sends retrieved data to the “standard output” - that is, the console, unless the shell output has been redirected. Redirection allows you, for

example, to pipe the output to other programs; see your shell documentation for more information.

Running multiple queries

```
tendo table1 query1 table2 query2 (or sqlquery1 sqlquery2 ...)
```

When running multiple queries, results are concatenated. To split results for each query into files use either `-o` or `-O`:

`-o filename` option with `@` in the filename. For each query, save the result into a file with `@` replaced by the name of the file containing the query or `-` if the query was specified on the command line - the table being operated on. For example,

```
tendo -g www -k -u username -o @.txt  
pub.demo.weather.stations .
```

would store output in a file named `pub.demo.weather.stations.txt`, and

```
tendo -g www -k -u username -o @.csv  
pub.demo.weather.stations @query1.txt
```

would store output in a file named `query1.txt.csv`

If `-o` is called without an `@` filename and multiple queries are run, the results will be concatenated and put into one file. If a header option is specified, a separate header for each table will be output. See `-t` and `-T` to control how tables are separated.

`-O filename` stores the output of successive queries as `filenameNUM` where `NUM` is incremented for each query. In other words, the first query will be saved in `filename1`, the second query in `filename2`, and so on.

`-!` continues running queries even if an error is encountered. Normally, `tendo` will abort as soon as one query encounters an error; this allows for better error handling in scripts. `-!` forces `tendo` to continue. Note that under Mac/Linux, the `!` must be escaped yielding `-\!`

Formatting Output

`Tendo`'s default output is comma-separated and unformatted: numbers appear without commas and with as many digits as available after the decimal point and dates, times, and timestamps are output in raw numeric form. Additionally, no header information is provided. This behavior can be modified by `Tendo`'s large number of format options.

- **f** outputs each column according to its format specification – that is, how it is displayed in the 1010data web GUI. Normally, Tendo outputs information in a raw format. For example, time is represented internally as an integer in the format YYYYMMDD, numbers do not have commas, etc. - **f** respects the format string for each column and outputs data as the web interface would represent it. Time includes slashes, numbers may include commas, only the specified number of decimal places are given, etc.

- **I** replaces numeric infinities with a given string. For example, -**I** `inf` replaces positive infinity with `inf` and negative infinity with `-inf` (the default)

- **m** include 1010data column format header describing how each column is to be formatted with - **f**. This is in the same format as the 1010data web interface.

- **h** include column names at top (possible to use both **h** and **H** - order determines output order e.g. -**hH** -**Hh**)

- **H** include column labels at top (possible to use both **h** and **H**, see above)

- **d** include column datatype

- **@** tab-delimited with headers and formatting, in a format that is suitable for direct upload to 1010data

- **x** XML format, with format information. Suitable for direct upload to 1010data

Custom formatting

Quoting

Certain characters may be unprintable on screen or may interfere with a file format. For example, in comma-separated value format (CSV), a comma in a text field would be interpreted as two separate fields. Quoting allows these elements to be represented in a different format. The default is to use CSV-style quoting with a double-quote (“)

- **q** *C* enables CSV-style quoting, but substitutes *C* instead of a double-quote (“). Normally, if a comma is encountered in the data that field will be surrounded with double-quotes (“). This prevents the comma from being interpreted as a field separator. For example, `Smith, John` becomes `“Smith, John.”` The -**q** option allows another character to be used instead.

- **e** *C* enables C-style quoting using character *C*. C-style quoting prepends delimiters with a character, usually the backslash (\). If using a \, you must specify \\ (because the backslash itself must be escaped on the shell). For example, with -**e** `\\` `Smith, John` becomes `Smith\\, John`.

-E use XML-style escaping. XML escapes unusual characters with an ampersand, the character code, and a semicolon. For example, the < is represented as < while the double quote (") is represented as "

-Q disables any quoting. Note that if you use -Q and column data contains the delimiter character, it may not be possible for applications to read the resulting file.

-, C replace column separator in data with C (note that you may have to put a backslash before the commas in Mac/Linux to yield -\,). Instead of quoting any irregular characters in data, they can be simply replaced. For example, with -, ! Smith, John becomes Smith! John.

-, , delete column separator in data (note that you may have to put backslashes before the commas in Mac/Linux to yield -\,\,). Similar to -, , except it simply deletes the character in question. Smith, John becomes Smith John.

-` (note this is a backtick, not an apostrophe, and that you may have to put a backslash before it in Mac/Linux to yield -\`). Replaces any newlines in data with ` (this is the standard 1010data header upload format). For example, a three-line header title named would be compressed into one: line1`line2`line3.

Delimiting

For delimited formats (CSV, tab-delimited) it is possible to modify what characters delimit fields and how these characters are handled.

To demonstrate the following four options, a sample table is presented below:

Last	First	Dollars
Simpson	Jane	20
Clarke	John	40

The default options are -c , -r \\n -t \\n. This means comma-separated columns and newline-separated rows (note the escaping characters, on Mac/Linux the first part may need to be -c \,). If multiple results are being concatenated, they are separated with a blank line (newline). With standard options, this table would look like:

Simpson,Jane,20

Clarke,John,40

-c CS use CS as a column separator. By default, a comma is used to separate fields. -c specifies that another character be used. For example, tab-delimited output can be obtained by -c \\t (note that unusual characters like the \ itself must be escaped by the shell, in the case of \ it means typing \\ instead). The above table

formatted with `-c !` would look like this (note that the default row separator, the newline, is left intact):

```
Simpson!Jane!20
```

```
Clarke!John!40
```

`-C CP` put CP before each column. This can be used in conjunction with `-c`. For example, `-C :` would convert Field1,Field2 into :Field1,:Field2. The above table formatted with `-C &` would look like this:

```
&Simpson,&Jane,&20
```

```
&Clarke,&John,&40
```

`-r RS` use RS as a row separator. Similar to `-c`, this separates rows. The default is a newline. To use comma-separated columns and semicolon-separated rows, use `-r ;`. This would look as follows:

```
Simpson,Jane,20;Clarke,John,40
```

`-R RP` put RP before each row, similar to `-C`, and can be used in conjunction with `-r`. The above table formatted with `-C /` would look as follows:

```
/Simpson,Jane,20
```

```
/Clarke,John,40
```

`-t TS` put TS as a line separator between concatenated query results. When `-o` is used to output tables into a single file, they are normally separated by a blank line (newline). `-t` may be used to control this line, and `-t ""` removes the line entirely.

`-T TP` put TP after every table, similar to `-t`.

Server-side operations

There are times where it is desirable to save the result of a query directly into a table. Tendo allows this with a simple syntax: prepending the query with a new table name and an equals sign: `NEWTAB=TABLE QUERY`. For example,:

```
tendo -g www -u username  
demo.newtable=pub.demo.weather.stations .
```

will take the output of the query and store it in demo.newtable. The location for the new table must be in a folder you have permission to write to; this is the same as creating a new table in the 1010data web GUI. If you are overwriting an existing table, `-y` must be used.

If you are running successive queries and want to accrue the results in a table, `NEWTAB+=TABLE QUERY` will accomplish this. The data to be appended must be in the same column format as the existing table.

Additionally, existing tables can be merged on the server without a query. This is accomplished through `NEWTAB=TABLE1+TABLE2 [+ . . .]` which allows for multiple tables to be merged (in order) at once. All tables must have the same column format.

A table can be deleted with `TABLE*=0`, but `-y` must be present. Any operation that might overwrite or delete an existing table must include `-y`.

Finally, it may be desirable to store a table on the server and access it through 1010data's FTP interface. If you have an FTP access activated, the results of a query may be stored as follows: `ftp:FILE=TABLE QUERY` as in:

```
tendo -g www -u username  
ftp:weatherstations.txt=pub.demo.weather.stations .
```

Any formatting options will affect the FTP file's format. If you do not have FTP activated on your account, contact support@1010data.com to request more information.

Managing Sessions/Storing Defaults

Managing Sessions

Instead of reauthenticating for every query, an open session can be created using the `-S` option. This returns a session key that may be used with successive `-s` options to maintain an open session. With `-s`, a password is not required.

For example, running `tendo -g www -u username -S` will yield a session ID. Then, successive queries can be run (without a password) by passing the `-s` option with that session ID. Also note that the session ID can be stored in an environment variable. For example:

```
tendo -g www -u username -s 936437202,Qnxoekmvplskg  
pub.demo.weather.stations .
```

When finished, run Tendo with the `-s` and `-l` options to log out of the session.

Storing Defaults

There are five environment variables that can store defaults. These can be set through batch scripts or in a shell environment. If sufficient user credentials are supplied through environment variables, Tendo can be used to directly execute

queries. For example, if TENTENGW, TENTENUID, and TENTENPW are set, the following will obtain weather station data:

```
tendo pub.demo.weather.stations .
```

If both a command-line option and an environment variable are set, the command-line option will override the variable.

The five environment variables are:

- TENTENGW - the gateway, equivalent to setting `-g`
- TENTENUID - the user, equivalent to setting `-u`
- TENTENPW - the password, equivalent to `-p`
- TENTENSID - the session ID, equivalent to `-s`
- TENTENPROXY - the proxy, equivalent to `-P`

On Mac/Linux, these are set using `setenv` (tcsh) or `export` (bash). More help can be found with your shell's documentation. On Windows, environment variables are set using `set VARIABLE=value`. For example, if you normally connect to `www.1010data.com` for the 1010 web GUI, your gateway shortcut is `www` (the shortcut for `www2.1010data.com` is `www2`, this is discussed in the Gateway section above). By running `set TENTENGW=www` the gateway does not need to be specified and the command can be shortened:

```
tendo -u username pub.demo.weather.stations .
```

Unless saved as part of a logon script, environment variables can be cleared by closing and re-opening the shell.

Miscellaneous

`-J` clears the cache before the query (if any) is run. This is only with a persistent session; otherwise the query is run immediately after login

`-j` returns the total memory use (in bytes) after the query is run. This can be useful for determining how resource-intensive a query is

`-%i on` or `-%i off` sets infinity-to-NA handling, see 1010data online documentation for more information

`-%s on` or `-%s off` sets stepwise aggregation, see 1010data online documentation for more information

`-%b FACTOR` sets the blocking factor, where FACTOR is between 0 and 10 (inclusive). See 1010data online documentation for more information