TenDo User's Guide
Contents

Introduction .................................................................................................. 4

Installing TenDo ........................................................................................... 5
  Installing TenDo on Windows........................................................................ 5
  Installing TenDo on Linux or Mac................................................................. 5

Basic usage .................................................................................................. 7
  Required switches........................................................................................ 7
  Table name.................................................................................................. 8
  1010data queries........................................................................................ 8
    Sending Queries from Files....................................................................... 9
    Returning an Entire Table....................................................................... 9
  Specifying a gateway (-g)......................................................................... 10

Output Control ........................................................................................... 11
  Saving TenDo Results in a File................................................................. 11
    Specifying How Much Data is Returned.............................................. 11
  Specifying the Output Format............................................................... 12
    Formatting Option Rules and Examples............................................. 13
      Using Column and Row Delimiters.................................................. 13
      Including Column Names and Headers........................................... 13

Managing connectivity .............................................................................. 16
  Kill commands......................................................................................... 16
  Connecting from a corporate proxy......................................................... 16
  Forced retry logic.................................................................................... 16
  SAM pools............................................................................................... 17

Session management ................................................................................. 18
  Viewing folder contents.......................................................................... 18
  Saving query results to the server.......................................................... 18
    Overwriting an existing table............................................................... 19
    Specifying table name and display information................................. 19
      Viewing Table Meta-Data................................................................. 19
  Appending Tables.................................................................................... 19
    Appending Query Results to an Existing Table................................... 19
    Appending Existing 1010data Tables................................................... 20
  Controlling New Table Permissions...................................................... 20
  Saving Query Results to an FTP Account.............................................. 20
  Deleting Tables....................................................................................... 20

Advanced Topics: Shell Scripts................................................................. 21
  Setting Environment Variables.............................................................. 21
    Setting Environment Variables in Windows...................................... 22
Setting Environment Variables in MacOS and in Linux .......................................................... 22
Session management .................................................................................................................. 22
Possessing and terminating an existing session .................................................................. 22
Possessing an existing session .............................................................................................. 22
Terminating an existing session ............................................................................................ 23
Staying logged in and logging out options .......................................................................... 23
Memory Management .......................................................................................................... 23
Using a Session ID .................................................................................................................. 23
Variable Substitution .............................................................................................................. 24
Example Scripts ...................................................................................................................... 25
Windows: A TenDo Batch Script ............................................................................................ 25
Linux and MacOS: A Bash Script ............................................................................................ 25

Return codes .......................................................................................................................... 27

Support Requests .................................................................................................................. 29
Required Files ......................................................................................................................... 29
Creating the XML Log File .................................................................................................. 29
Creating the TenDo Debugging File ..................................................................................... 29
Most 1010data users are familiar with our web-based interface. Many don’t realize that there are several ways to interact with 1010data, each with its own features that make specific tasks faster and more efficient. For running queries, especially queries that are run on a regular basis, TenDo provides a fast, efficient, programmatic way to interact with 1010data.

TenDo is program run from a Command Line Interface (CLI). It is built to submit queries to 1010data written in our Macro Language.

Users who typically interact with 1010data via their web browsers are used to doing everything in steps. First, they go to the website. Then they log in. Then, they open a table. Then, they start to select rows, or run a query, and so on. TenDo provides a way for our users to perform all these steps in a single transaction. Instead of first logging into the system, then opening a table, with TenDo you have the power to do all this in one easy step. Here’s a list of some advantages offered by TenDo:

- You can login to 1010data, specify a table and run a query all in one step, without the need to wait for results or personally monitor the system.
- You can create scripts that automate the process even further. If you run the same query on the same table in regular intervals, TenDo is your new best friend.
- Once you’ve create scripts you can schedule them to run using your operating system's scheduler, allowing for even more automation of your interactions with 1010data.
- Some people just prefer to work with a CLI.
Installing TenDo

To install TenDo on your system visit the following link: http://www.1010data.com/downloads/tendo/tendo.zip. Clicking this link will automatically download the zip file containing the TenDo unified installer. To unzip the file from your terminal on MacOS or Linux you can use the gunzip command from the directory where the TenDo zip file is saved, as follows:

$gunzip tendo.zip

This command will unpack the contents of the tendo.zip file. From there, follow the instructions for installation that are specific to your operating system. If you are on a Windows system, you can simply right-click the icon in the directory where you saved the TenDo zip file, then select Extract All....

Installing TenDo on Windows

To install TenDo on Windows, first identify if you have a 32-bit or 64-bit version of the operating system. Then open the corresponding folder under the Tendo folder (e.g., if you have a 32-bit copy, open the Win32 folder and if you have a 64-bit copy, open the Win64 folder). Then, simply double-click the executable file inside.

If you're not sure how to tell what version of Windows you're using complete the following steps:

1. Click the Start button
2. Right-click the Computer icon
3. Click the Properties option in the menu that appears

A new Window will appear that contains a lot of information about your computer. You can find the version of Windows you have installed in the System section, as in the screenshot below:

![System Information]

Installing TenDo on Linux or Mac

To install TenDo on a Linux system, first identify whether you have a 32-bit or 64-bit version of Linux. Then, open the corresponding folder (e.g., if you have a 32-bit copy, open the lin32 folder and if you have a 64-bit copy, open the lin64 folder). Then, copy the tendo executable inside the folder to the file path:

/usr/local/bin

To install on a Mac, follow the same procedure as the Linux instructions above, except copy executable file from the Tendo/osx directory instead of one of the linux directories.

Note: In order to copy the file to

/usr/local/bin

You will most likely need administrative access on your computer.
You can also add the directory where TenDo is stored to your $PATH variable. This method doesn't require you to have administrative access. To run TenDo as a command from any directory, simply add the directory where TenDo is saved by following these steps.

1. Go to your home directory with the following command
   • $cd ~

2. Open your bash profile with the following command:
   • $vim .bash_profile

   Note: vim is just one text editor that is available in a standard bash installation. If you're not familiar with Vim you can use your favorite text editor instead.

3. Add the following command to the next empty line of the file:
   • export PATH="$PATH:$HOME/[CURRENT TENDO DIRECTORY]"

4. Save the changes to your file

Once you've added the current TenDo directory to your $PATH you can invoke TenDo from any directory on your computer.

   Note: You may have to close your terminal and re-open it before the changes to your $PATH take effect.

Once you've copied the file to its correct location, you will be able to run tendo as a shell command.

The next section will cover all the basic information you need to query 1010data with TenDo.
Basic usage

Once you’ve successfully installed TenDo on your system, you can start submitting queries to 1010data. There are just a few simple components to a basic TenDo command. Here’s what the anatomy of a basic query in TenDo looks like:

```bash
$tendo -u [USERNAME] -p [PASSWORD] [TABLE] "[QUERY]"
```

Keep in mind that this is a basic TenDo command. There are plenty of options and alternative methods available, and we’ll get to them a little later in this manual. But these are the pieces of information that are required by TenDo. Let’s go through each piece one at a time.

**Note:** When you see a word in all caps enclosed in brackets ([ ]), this represents a variable. So when you see [USERNAME] in a command, you would substitute your 1010data username. This notation will be used throughout this document.

### Required switches

#### Table 1: A Basic TenDo Command

<table>
<thead>
<tr>
<th>Command Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>tendo</code></td>
<td>How the TenDo program is started. All TenDo commands begin with this.</td>
</tr>
<tr>
<td><code>-u [USERNAME]</code></td>
<td>The <code>-u</code> tells TenDo you are providing a username to send to 1010data. After the <code>-u</code>, simply type a space and then your 1010data username.</td>
</tr>
<tr>
<td><code>-p [PASSWORD]</code></td>
<td>The <code>-p</code> tells TenDo you are providing a password to send to 1010data. After the <code>-p</code>, simply type a space and then your 1010data password.</td>
</tr>
<tr>
<td><code>[TABLE]</code></td>
<td>Provide the 1010data base table your query will run on.</td>
</tr>
<tr>
<td><code>&quot;[QUERY]&quot;</code></td>
<td>Provide your 1010data query here. The entire query must be contained in the quotation marks.</td>
</tr>
</tbody>
</table>

While this basic command structure is all that is required of most interactions using TenDo, there are some important exceptions of which you should be aware:

- If you don’t provide a password in your command, TenDo will automatically prompt you for it. Examples for the rest of the manual won’t include passwords.
- If you place your query directly in your TenDo command, quotation marks in the query need to be escaped with a backslash (\) (See **Example 1**).
- If your company uses a custom URL to access 1010data, or you want to use a version of 1010data that’s different from the version your company has been provisioned to use, you will need to provide a gateway in addition to the information detailed above.

#### Example 1: Escaping Quotation Marks

Escaping quotations marks in the Windows Command Prompt or in a Linux/MacOS shell is done using the following syntax:

```bash
$tendo -u [USERNAME] [TABLE] "<sel value="(store=1)"/>
```

Keep these basic rules in mind and sending simple queries to 1010data can be done quickly and with minimal interaction.
Table name

If you're not sure of the exact format you need to provide for the base table variable, it's easy to find. Log into 1010data, select Tools > Object Manager, and select your table. The internal table name appears in the bottom of the Object Manager pane.

If we were to query this table with our example query above it would appear in the TenDo command as follows:

```
$ tendo -u [USERNAME] pub.training_exercise_data.sales_orders_2017 "[QUERY]"
```

1010data queries

TenDo requires you send a query written in valid 1010data Macro Language code. If you're used to building queries in the 1010data browser-based interface, you can copy valid Macro Language code from that interface into your TenDo command. To copy the code out of 1010data's web-interface, first build and run your query. Then, click the Query tab. The Query tab displays your query in Macro Language code.
For TenDo, simply copy the query code (starting with `<sel`) and paste it into your TenDo command. Just remember, you have to put any code you paste in quotation marks (" ") as follows:

```
$ tendo -u [USERNAME] pub.training_exercise_data.sales_orders_2017 "<sel value="(orderdate=20170201)"/>
```

Notice in the above example we had to escape the quotation marks (" "). This can be an effective technique if you just need to use a small query. Most users do a lot more than simple selection statements in their queries. If you want to use larger, more complex queries, typing them in the command itself isn't very practical. Fortunately, there's a more efficient way.

**Note:** Depending on the shell you're using, you may encounter other characters that must be escaped. A 1010data Insights Platform best practice is to always send your query in a file.

### Sending Queries from Files

Sending short, simple queries in your TenDo commands is a good way to get comfortable with TenDo's basic functionality. But you're most likely interested in sending much more powerful queries than simple selection statements, or other basic analysis. TenDo can accept queries stored in text files so you don't have to type them out each time, or even copy and paste them. Instead, you can provide TenDo with a file location as an argument, preceded by the `@` symbol, as follows:

```
$ tendo -u [USERNAME] [TABLE] @[QUERYFILE]
```

You can also copy Macro Language queries from the Query tab of TRS and paste them into the file you provide TenDo. However, make sure you always save the file as an ASCII file.

### Returning an Entire Table

If you want to return an entire table in TenDo, you can place a single dot (.) character in place of the query, as follows:

```
```

This command returns the entire base table, in this case `pub.training_exercise_data.sales_orders_2017`, in a .csv format on your terminal or Command Prompt screen.

With these basic concepts you have everything you need to send a query to 1010data with TenDo. The next section will look at what 1010data returns after it has run your query, and what you can do with TenDo to control that output.
Specifying a gateway (-g)

When you log in to 1010data, whether its via TenDo or the web GUI, by default you will automatically be logged in to the version of 1010data that your organization has been provisioned to use. This is the default behavior and no action on your part is required. However, there are some situations in which you may want to use a different version of 1010data than your default version. For instance, our beta environment is often used by 1010data users who want to see what features and improvements are on the horizon. Furthermore, some of our clients use custom URLs for 1010data access. This means that instead of going to the standard www2.1010data.com URL for access, they have some custom corporate URL such as: www.analyticsplayground.com.

In either instance (e.g., you want to use a specific version of 1010data you're not provisioned for, or your company uses a custom corporate gateway) you will need to specify your gateway to TenDo. The following two examples show the syntax for each circumstance you may encounter when you need to specify a gateway. In both cases you will use the -g option. To specify a specific version of 1010data, you only need to specify the version, as follows:

```
c:>tendo -u [USERNAME] -g beta-7.03 [TABLE] @[QUERYFILE]
```

The example above tells TenDo not to use the standard gateway logic but to instead create a session in 1010data version 7.03. The current beta versions of 1010data begin at version 7.01.

To specify a custom corporate gateway, you need to provide the fully qualified URL, as follows:

```
```

If you need to use a corporate gateway and also want to use a specific version of 1010data you can combine the previous examples, as follows:

```
```
Output Control

Now that you know how to request query results from the 1010data Insights Platform via TenDo, it would be helpful to know how to manage and save your results. Without specifying a file to store your results, TenDo will automatically show results to your CLI window in a Comman Seaparated Values (.csv) format. However, you can also save this CSV information in a .csv file, as well as specify the exact amount of data saved in the file.

Saving TenDo Results in a File

To save your results to a .csv file, TenDo provides the -o1010data option. It is called by placing -o in your TenDo command followed by a space and the location of the file where you want to save your results. The -o option can be placed anywhere in your TenDo command except between the Table and the Query. However, this manual will place all output options at the end of example commands for syntactical (as well as logical) consistency. Here's an example:

```
$ tendo -u [USERNAME] [TABLE] @[QUERYFILE] -o [FILEPATH]
```

If you only provide TenDo with a file name the output file will be saved in your current working directory. However, the best solution is to simply provide a file path. This is done with the -o option, as follows:

```
$ tendo -u [USERNAME] [TABLE] @[QUERYFILE] -o ~/Downloads/tendoResults.csv
$ tendo -u [USERNAME] [TABLE] @[QUERYFILE] -o C:\Users\TLin\Downloads\tendoResults.csv
```

TenDo returns all data as UTF-8 encoded characters.

**Note:** UTF-8 is a standard character set that is used as a standard for computer interactions that require characters. TenDo only returns data encoded in UTF-8.

When you specify a file name, TenDo will simply create the file for you if it doesn't exist. If it does exist, TenDo will overwrite the existing file with the new output. If you don't have permission to create a file in the specified location, TenDo will return an error.

Specifying How Much Data is Returned

TenDo provides functionality to control how many rows of data are returned from your query results, as well as where the rows returned begin. The -n option specifies the number of rows that are returned from your query results. The -n option does not apply to data sets that are rendered from a QuickApp.

The -i option controls the row at which the returned results begin. At 1010data we refer to this as the "offset." These options can be used individually, or in unison to further specify the result output:

**Specify the Number of Rows Return with -n:**

```
$ tendo -u [USERNAME] [TABLE] @[QUERYFILE] -o [FILEPATH] -n 10
```

This command will return the first 10 rows of your query results and save them to the specified file. It should be noted that this is the maximum number of rows to return. If there are fewer rows available, all rows will be returned.

**Specify the First Row of Results to Return:**

```
$ tendo -u [USERNAME] [TABLE] @[QUERYFILE] -o [FILEPATH] -i 11
```

This command will return the rows of results from your query beginning with and including the 11th row.

**Use -n and -i in Unison to Control Returned Results:**

```
$ tendo -u [USERNAME] [TABLE] @[QUERYFILE] -o [FILEPATH] -n 10 -i 11
```

This command will return 10 rows of results, starting with row 11.
Specifying the Output Format

When you receive query results from 1010data via TenDo you have numerous options for how the data is formatted. These options are applicable whether the results are printed to the console or written to a file. The following table provides formatting options for TenDo commands as well as brief descriptions for what each option does. Some options are "position dependent," meaning they behave differently depending on where in the TenDo command they are placed.

Table 2: Data Formatting Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-f</td>
<td>Formats the data in columns returned to TenDo in the same way they are formatted in the 1010data GUI. For example, dates are typically formatted as 8-digit integers when returned to TenDo. This option will format the date as it is formatted in the GUI, whether it has slashes, dashes, etc.</td>
</tr>
</tbody>
</table>
| -`     | Replaces new lines with a backtick (`) character in the column headings. This way column headings don't stack on top of each other in the console or output file the way they would if formatted to do so in the 1010data GUI.  
**Note:** In most shells, including bash and Windows Command Prompt, the backtick character must be escape with a backslash (`\`) character. |
| -c     | Specify a column delimiter, if you want a different delimiter than the default command (,) character. This option takes a single character as an argument. |
| -r     | Specify a row delimiter if you want a different row delimiter than the default delimiter, which is a new line in the results. This option takes a single character as an argument. |
| -h     | Include column names in results. |
| -H     | Include column headings in results. |
| -d     | Returns the 1010data Insights Platform data types of the columns. Data types are indicated as:  
• i for integers (integer)  
• f for floating point numbers (decimal)  
• a for strings of characters (text)  
• j for 64-bit integers (bigint)  
For more information about platform data types, see *Data Types*. |
| -m     | Returns the format string of the columns in the results according to 1010data format types |
| -F     | Produces fixed width columns for the results |
Formatting Option Rules and Examples

When using the formatting options specified in the table above there are a number considerations to make. This section provides details to help you get the specific output format you're looking for.

Using Column and Row Delimiters

One output option you may need is to place a specific character between table columns and/or rows in the data output. By default, columns are separated by commas (,) and rows are separated by new lines. This is known as "delimiting" the data so it's easier to tell where one row or column ends and another begins.

You may encounter a situation in which you need to delimit rows or columns with a different character than the default options. For example, it is common to separate columns with a pipe character (|). We can do this by using the -c option in the following manner:

**Note:** While it is common to delimit columns with a pipe (|), this character is also used in most command line interfaces for special purposes. To avoid errors, you have to escape the pipe character with a backslash. This is shown in the example below.

```
$tendo -c \| -u [USERNAME] pub.training_exercise_data.sales_orders_2017
```

The command above will produce the following results if output to the console:

```
| 51 | 597 | 2010-01-31 | 56 | 1.500000 | 1.549000 |
| 52 | 476 | 2010-01-31 | 96 | 9.500000 | 9.259000 |
| 53 | 471 | 2010-01-31 | 97 | 1.100000 | 0.560000 |
| 54 | 750 | 2010-01-31 | 98 | 1.150000 | 1.769000 |
| 55 | 751 | 2010-01-31 | 99 | 1.100000 | 1.390000 |
| 56 | 760 | 2010-01-31 | 100 | 1.200000 | 1.890000 |
| 57 | 761 | 2010-01-31 | 101 | 1.000000 | 1.000000 |
| 58 | 762 | 2010-01-31 | 102 | 1.000000 | 1.000000 |
| 59 | 763 | 2010-01-31 | 103 | 1.000000 | 1.000000 |
| 60 | 660 | 2010-01-31 | 104 | 1.500000 | 1.000000 |
| 61 | 661 | 2010-01-31 | 105 | 1.000000 | 1.000000 |
| 62 | 662 | 2010-01-31 | 106 | 1.000000 | 1.000000 |
| 63 | 663 | 2010-01-31 | 107 | 1.000000 | 1.000000 |
| 64 | 850 | 2010-01-31 | 108 | 1.000000 | 1.000000 |
| 65 | 851 | 2010-01-31 | 109 | 1.000000 | 1.000000 |

It is also important to note that the other options in this section are position sensitive relative to the -c and -r options. If you need to use these options you should place them ahead of other formatting options in your command to avoid confusion.

Including Column Names and Headers

There are two ways that columns are labeled in 1010data. Column Names are the actual values 1010data uses to recognize columns in a table. Column Names must be all lowercase letters and cannot include spaces or extra characters. The column Names are what you use when writing 1010data expressions. Column Headings, on the other hand, can use capital letters, spaces and other symbols. Headings are meant to be human readable and are disregarded by 1010data when operating on data. Column Names
are returned by including the `-h` option. Column **Headings** are returned by including the `-H` option. These options are position dependent upon one another, as shown in the following two examples (notice we're also including the `-c` option to separate the columns).

```
```

**Note:** Note, when using the pipe (`|`) character in a Windows command line the `|` must be escaped with a caret (`^`) character, as follows:

```
```

This command will produce the following results when printed in the console:

```
<table>
<thead>
<tr>
<th>transaction_id</th>
<th>account</th>
<th>store</th>
<th>date</th>
<th>sku</th>
<th>units</th>
<th>sales</th>
<th>cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>51</td>
<td>1</td>
<td>1218515</td>
<td>2561</td>
<td>5.00000000</td>
<td>1.00000000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>52</td>
<td>473</td>
<td>1218515</td>
<td>3971</td>
<td>1.00000000</td>
<td>1.00000000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>53</td>
<td>730</td>
<td>1218516</td>
<td>4962</td>
<td>6.00000000</td>
<td>1.00000000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>54</td>
<td>730</td>
<td>1218516</td>
<td>4962</td>
<td>6.00000000</td>
<td>1.00000000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>55</td>
<td>740</td>
<td>1218515</td>
<td>561</td>
<td>1.00000000</td>
<td>1.00000000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>56</td>
<td>740</td>
<td>1218515</td>
<td>561</td>
<td>1.00000000</td>
<td>1.00000000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>57</td>
<td>740</td>
<td>1218515</td>
<td>561</td>
<td>1.00000000</td>
<td>1.00000000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>58</td>
<td>740</td>
<td>1218515</td>
<td>561</td>
<td>1.00000000</td>
<td>1.00000000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>59</td>
<td>740</td>
<td>1218515</td>
<td>561</td>
<td>1.00000000</td>
<td>1.00000000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>60</td>
<td>740</td>
<td>1218515</td>
<td>561</td>
<td>1.00000000</td>
<td>1.00000000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>61</td>
<td>740</td>
<td>1218515</td>
<td>561</td>
<td>1.00000000</td>
<td>1.00000000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>62</td>
<td>740</td>
<td>1218515</td>
<td>561</td>
<td>1.00000000</td>
<td>1.00000000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>63</td>
<td>740</td>
<td>1218515</td>
<td>561</td>
<td>1.00000000</td>
<td>1.00000000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>64</td>
<td>740</td>
<td>1218515</td>
<td>561</td>
<td>1.00000000</td>
<td>1.00000000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>65</td>
<td>740</td>
<td>1218515</td>
<td>561</td>
<td>1.00000000</td>
<td>1.00000000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>66</td>
<td>740</td>
<td>1218515</td>
<td>561</td>
<td>1.00000000</td>
<td>1.00000000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>67</td>
<td>740</td>
<td>1218515</td>
<td>561</td>
<td>1.00000000</td>
<td>1.00000000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>68</td>
<td>740</td>
<td>1218515</td>
<td>561</td>
<td>1.00000000</td>
<td>1.00000000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>69</td>
<td>740</td>
<td>1218515</td>
<td>561</td>
<td>1.00000000</td>
<td>1.00000000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>70</td>
<td>740</td>
<td>1218515</td>
<td>561</td>
<td>1.00000000</td>
<td>1.00000000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>71</td>
<td>740</td>
<td>1218515</td>
<td>561</td>
<td>1.00000000</td>
<td>1.00000000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>72</td>
<td>740</td>
<td>1218515</td>
<td>561</td>
<td>1.00000000</td>
<td>1.00000000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>73</td>
<td>740</td>
<td>1218515</td>
<td>561</td>
<td>1.00000000</td>
<td>1.00000000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>74</td>
<td>740</td>
<td>1218515</td>
<td>561</td>
<td>1.00000000</td>
<td>1.00000000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>75</td>
<td>740</td>
<td>1218515</td>
<td>561</td>
<td>1.00000000</td>
<td>1.00000000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>76</td>
<td>740</td>
<td>1218515</td>
<td>561</td>
<td>1.00000000</td>
<td>1.00000000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>77</td>
<td>740</td>
<td>1218515</td>
<td>561</td>
<td>1.00000000</td>
<td>1.00000000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>78</td>
<td>740</td>
<td>1218515</td>
<td>561</td>
<td>1.00000000</td>
<td>1.00000000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>79</td>
<td>740</td>
<td>1218515</td>
<td>561</td>
<td>1.00000000</td>
<td>1.00000000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>80</td>
<td>740</td>
<td>1218515</td>
<td>561</td>
<td>1.00000000</td>
<td>1.00000000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>81</td>
<td>740</td>
<td>1218515</td>
<td>561</td>
<td>1.00000000</td>
<td>1.00000000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>82</td>
<td>740</td>
<td>1218515</td>
<td>561</td>
<td>1.00000000</td>
<td>1.00000000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>83</td>
<td>740</td>
<td>1218515</td>
<td>561</td>
<td>1.00000000</td>
<td>1.00000000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>84</td>
<td>740</td>
<td>1218515</td>
<td>561</td>
<td>1.00000000</td>
<td>1.00000000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>85</td>
<td>740</td>
<td>1218515</td>
<td>561</td>
<td>1.00000000</td>
<td>1.00000000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>86</td>
<td>740</td>
<td>1218515</td>
<td>561</td>
<td>1.00000000</td>
<td>1.00000000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>87</td>
<td>740</td>
<td>1218515</td>
<td>561</td>
<td>1.00000000</td>
<td>1.00000000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>88</td>
<td>740</td>
<td>1218515</td>
<td>561</td>
<td>1.00000000</td>
<td>1.00000000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>89</td>
<td>740</td>
<td>1218515</td>
<td>561</td>
<td>1.00000000</td>
<td>1.00000000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>90</td>
<td>740</td>
<td>1218515</td>
<td>561</td>
<td>1.00000000</td>
<td>1.00000000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>91</td>
<td>740</td>
<td>1218515</td>
<td>561</td>
<td>1.00000000</td>
<td>1.00000000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>92</td>
<td>740</td>
<td>1218515</td>
<td>561</td>
<td>1.00000000</td>
<td>1.00000000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>93</td>
<td>740</td>
<td>1218515</td>
<td>561</td>
<td>1.00000000</td>
<td>1.00000000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>94</td>
<td>740</td>
<td>1218515</td>
<td>561</td>
<td>1.00000000</td>
<td>1.00000000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>95</td>
<td>740</td>
<td>1218515</td>
<td>561</td>
<td>1.00000000</td>
<td>1.00000000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>96</td>
<td>740</td>
<td>1218515</td>
<td>561</td>
<td>1.00000000</td>
<td>1.00000000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>97</td>
<td>740</td>
<td>1218515</td>
<td>561</td>
<td>1.00000000</td>
<td>1.00000000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>98</td>
<td>740</td>
<td>1218515</td>
<td>561</td>
<td>1.00000000</td>
<td>1.00000000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>99</td>
<td>740</td>
<td>1218515</td>
<td>561</td>
<td>1.00000000</td>
<td>1.00000000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>100</td>
<td>740</td>
<td>1218515</td>
<td>561</td>
<td>1.00000000</td>
<td>1.00000000</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

The next example shows what happens if you reverse the order:

```
```

This example will place the Headings ahead of the names, as shown below:
You can also include a different delimiter for both column **Names** and column **Headings** than that provided for the actual columns. Here is an example that produces one delimiter for names, one for headings and a third for the columns themselves:

```
$ tendo -c \- -h -c \@ -H -c \| -u [USERNAME]
```

**Note:** Notice in the above example all three delimiting characters (such as @ and |) need to be escaped with the backslash (\) character.

The above example will produce the following output to the console:
Managing connectivity

In this section we will cover the various configuration options for TenDo. TenDo provides functionality for terminating sessions, keeping sessions alive and connecting to 1010data from behind corporate proxy servers.

Kill commands

You may encounter a situation in which the account you use to connect to 1010data has already established a connection outside of your TenDo session. This could happen if you recently logged into 1010data via another interface and didn't log out, or if you use a shared account within your workgroup. If you try to send a query in TenDo while your account is already logged in, TenDo will return an error:

can't log in: rc=5 (Already logged in)

In this event, you will need to end the existing connection before TenDo can use the account to connect to 1010data. You do this with a Kill Command. This functionality is provided by TenDo with the -k option. This option doesn't take an argument and is called, as follows:

$ tendo -k -u [USERNAME] [TABLE]@[QUERYFILE]

This command will automatically disconnect your existing session and run your query via TenDo.

Connecting from a corporate proxy

If you connect to the internet from behind a corporate proxy server, you can tell TenDo to connect through the proxy. TenDo provides this functionality with the -P option, followed by a space and the proxy information, formatted as follows:

$ tendo -u [USERNAME] -P http://[PROXYUSERNAME]:[PROXYUSERPASSWORD]@www.corporateproxyserver.com:8080 [TABLE]@[QUERYFILE]

This command routes the TenDo connection through the specified proxy server, then connects to 1010data and runs your query as normal. If you do not know the correct proxy server information to provide, ask your network administrator. If you encounter the Transport error 7: Couldn't connect to server message, you need to add proxy credentials.

Note: Not all proxy servers require a user name and password. Check with your network administrator to find out exactly what information you need to connect via your company’s proxy server.

Forced retry logic

Note: TenDo automatically sends HTTP keep-alive packets to 1010data, which usually prevents connection failures. The techniques in this section should only be used when a connection issue has been encountered.

Some queries take longer to finish running than others. Occasionally, a TenDo session will timeout while a query is running if the query is very large. This is especially true if you are behind a corporate proxy server. To help manage your connection and prevent unwanted timeouts from occurring, TenDo provides two useful options: -- and --retries. You can use these options in unison to prevent timeouts from happening before your query finishes running.

First, let's look at the -- option. This option takes any number greater than 300 as an argument. The argument is the number of seconds you want the system to wait before it retries the transaction. The
number you specify for ~ should be lower than the time it takes your proxy server to cause a timeout. The ~ option is called in a TenDo command as follows:

$ tendo -u [USERNAME] [TABLE] @[QUERYFILE] ~ 400

This command tells TenDo to reestablish the transaction after 400 seconds. If a timeout has occurred, when TenDo retries the query, it will pick up where it left off at the time of the timeout. Note that we have not provided a number of times to retry. If you do not provide a number of retries using the --retries option, TenDo will default to 10 total attempts; the initial attempt and 9 retries.

If you need TenDo to retry more than 10 times, you can specify your retry preference with --retries. You can only use --retries if you have provided a retry duration with the ~ option, as follows:

$ tendo -u [USERNAME] [TABLE] @[QUERYFILE] ~ 400 --retries=15

This command tells TenDo to retry the query every 400 seconds for 15 consecutive attempts. If you multiply your specified retry interval by the number of retries you've specified, the product will be the total amount of time (in this case 6000 seconds or 1 hour and 40 minutes) before the transaction will timeout entirely. This is important, because your total time before a complete timeout should roughly be as long as the maximum theoretical time it will take your query to run. You may need to experiment a little with your proxy server and your queries to really nail down the formula.

Note: Queries make numerous "transactions" with 1010data when they are run. The retries option applies to each individual transaction in a query. This means that if your query makes one transaction with the system that times out twice before completing, the next transaction will start with a new set of retries, as specified in your command. Remember, 10 retries is the default. So, if you don't specify a number of retries then every transaction in your queries will have 10 retries. If you specify 15, each transaction will have 15...etc.

**SAM pools**

If you access 1010data as part of a SAM pool, you need to use different login credentials than if you had your own, unique 1010data ID. SAM stands for: Shared Access Management. SAM pools are generally used to share a single set of login credentials with a larger pool of actual users. Specifying your SAM pool is done with the --pool option in TenDo. It takes the Pool Name as it's only parameter. However, you will still need to pass TenDo a username and password, as follows:

$ tendo --pool=[SAMPOOLNAME] -u [POOLOWNERNAME] -p [POOLOWNERPASSWORD] [TABLE] @[QUERYFILE]

This tells TenDo to login to 1010data with the first free account in the SAM pool. If no accounts are presently available, TenDo will return an error stating that the system is busy.
Session management

TenDo also provides functionality for managing data on the server. Just as in the web-based interface, TenDo allows you to create new tables, save query results to existing tables, delete tables and set table permissions. This section will cover TenDo operations that control server-side data.

Viewing folder contents

When you need to change table and folder permissions it’s helpful to know exactly what the contents of the folders are. TenDo provides options for viewing the contents of folders. To see the contents of a specified folder you use the .. command after specifying the folder, as follows:

```
$ tendo -u [USERNAME] [FOLDER] ..
```

The above command lists the contents of a single folder, including sub-folders. However, the .. command doesn't list the contents of the subfolders. To see the contents of a folder and its sub-folders, use the ... command, as follows:

```
$ tendo -u [USERNAME] [FOLDER] ...
```

Saving query results to the server

There are several ways to save the results of a 1010data query to the server with TenDo. Note that if you save or append your query results to the server, TenDo will not print or save the results to your computer. The most common way to save results to the server is to create and save a new table. In TenDo, you do this as follows:

```
$ tendo -u [USERNAME] [FOLDER].[NEWTABLENAME]=[TABLE] @[QUERYFILE]
```

This command will create a new table in a specified folder, that is made up of the data returned by your query on the base table specified. The table below defines each component of the bolded elements in the command example:

**Table 3: Creating a New Table with TenDo**

<table>
<thead>
<tr>
<th>Command Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[FOLDER]</td>
<td>This folder specifies the location of your new table. You must have permission to write to the folder you specify for the new table to be created.</td>
</tr>
<tr>
<td>[NEWTABLENAME]</td>
<td>Specifies the name of the new table that will be created in the folder specified in the previous command component.</td>
</tr>
<tr>
<td>[TABLE]</td>
<td>Specifies the base table your query will run on.</td>
</tr>
<tr>
<td>@ [QUERYFILE]</td>
<td>The query you want to run on the base table. The results of this query will be saved to the new table in 1010data.</td>
</tr>
</tbody>
</table>

It is important to note the exact syntax of the example command above. The entire section of the command, from the folder through the base table must not have any spaces.
Overwriting an existing table

In addition to creating a brand new table, you can overwrite an existing table on 1010data with TenDo. The syntax is very similar to the previous example command, except to overwrite an existing table you need to use the -y option. Here’s an example:

**Note:** To overwrite a table you must own the table you want to overwrite.

```
$ tendo -u [USERNAME] [FOLDER].[NAMEOFEXISTINGTABLE]=TABLE @QUERYFILE -y
```

This command will overwrite an existing 1010data table with the results of the query provided. If you do not use the -y option, and the name of the table you’re trying to write already exists, TenDo will return an error.

Specifying table name and display information

When you create a new table or overwrite an existing table with TenDo, you can also specify the display title, subtitle and table help information. This is done by providing the new table’s display information in parentheses immediately after the new table is specified. Here’s an example:

```
$ tendo -u [USERNAME] [FOLDER].[NEWTABLENAME](Query Results Table)=TABLE @QUERYFILE
```

This command specifies the Title of the newly created table. To also include a Subtitle and Table Help Information, you can include the information in the parenthesis, separated by semicolons (;). Here’s an example:

```
$ tendo -u [USERNAME] [FOLDER].[NEWTABLENAME](Query Results Table Title;Subtitle;This is where you place helpful information that describes the table's purpose)=TABLE @QUERYFILE
```

This command will create the specified information in the table on 1010data, and that information will be visible in the 1010data web-based interface. If you want to include some information and leave other fields blank the syntax is simple. All you have to do is provide the semicolons but leave the information out. For instance, if we did not want to include a Subtitle in our last example, the command would look like this:

```
$ tendo -u [USERNAME] [FOLDER].[NEWTABLENAME](Query Results Table Title;;This is where you place helpful information that describes the table's purpose)=TABLE @QUERYFILE
```

**Note:** In these examples you have to escape the parentheses and semicolons.

Viewing Table Meta-Data

To see information about a table, use the ? command immediately after the table. The ? character must be escaped in both Linux/MacOS shells and Windows Command Prompt, as follows:

```
$ tendo -u [USERNAME] [TABLE]?
```

Appending Tables

TenDo provides functionality for both appending query results to an existing table, as well as appending multiple tables already on the server. Let’s look at the syntax for each instance.

**Note:** In order to append tables to one another, all tables must have the same exact column format.

Appending Query Results to an Existing Table

To add the results of a query to an existing 1010data table, simply use the += operator instead of the regular = operator from the previous example. For instance:

```
$ tendo -u [USERNAME] [FOLDER].[EXISTINGTABLENAME]+=[TABLE] @QUERYFILE
```
**Note:** Do not insert spaces around the plus-equals (+=) sign.

This command takes the results from a query on a base table and appends those results to a table that already exists in 1010data. The existing table can be any table you have permission to write to, or the base table itself.

### Appending Existing 1010data Tables

As in the 1010data GUI, you can append existing tables to one another. This is done very simply with the + operator, as follows:

```
$ tendo -u [USERNAME] [NEWTABLENAME]=[TABLE1]+[TABLE2]+[TABLE3]+...[NthTABLE]
```

**Note:** Do not insert spaces around the equals sign (=) or plus signs (+).

This command creates a new table that contains the data from table 1 through the Nth table, in the order that they were passed to TenDo.

### Controlling New Table Permissions

You can also control the access permissions to the new tables you create with TenDo interactions. By default, all new tables you save are private and can only be seen by you.

You can change the permissions of your new table with the -Y option.

```
tendo -u -[USERNAME] [NEWTABLENAME]=[TABLE] @[QUERYFILE] -Y
```

Finally, to grant specific users access to a table you use the -Y option and provide usernames as the arguments, as follows:

```
$ tendo -u [USERNAME] [NEWTABLENAME]=[TABLE] @[QUERYFILE] -Y [USER1],[USER2],[USER3]
```

This command grants permission to view the table only to the users specified in the command. The usernames should be separated by commas (,) with no spaces between each comma and the next username. Note that as the owner of the table, you do not need to specify yourself.

### Saving Query Results to an FTP Account

You may have a need to save your query results to your 1010data FTP in order to access them later from a different location. In order to do this, you must have an active FTP account with 1010data. Assuming you have FTP access, you can save your query results to your FTP account in the form of a .txt or .csv file.

Here's the command structure:

```
$ tendo -u [USERNAME] ftp:[NEWFILENAME]=[TABLE] @[QUERYFILE]
```

This command will save the query results to the specified file in your 1010data FTP account. Remember, it should be either a .txt or a .csv file type.

### Deleting Tables

**Note:** You must own a table to delete it.

You can delete tables in 1010data via TenDo as follows:

```
$ tendo -u [USERNAME] [TABLE]*=0 -y
```

This command deletes the specified table. Note that in order to delete the table, you need to include the -y option. -y is required for any operation that deletes or overwrites something in 1010data.
Advanced Topics: Shell Scripts

Since the most powerful uses of TenDo all have to do with automation, in a way all the previous sections of this manual have led us to this moment. Here, we will learn how to customize your TenDo settings so that it knows certain information about your account. This will allow you to include less information in each command. Finally, we will look at how to use TenDo to build and run shell scripts. In Windows, a shell script is called a Batch File. Batch files always end in a .bat file extension.

In MacOS and Linux, shell scripts go by a few different names, but both Shell Script or Bash Script are appropriate. Shell script files in MacOS and Linux end in the .sh file extension.

Shell scripts are extremely useful. They allow you to write many commands in a single file that will be executed in the order in which they appear. Often, these commands work together, each command using the results of the one that preceded it to accomplish a task that would be difficult or impossible with a single command. However, you can create a shell script that runs a series of completely unrelated commands as well. This section will detail the concepts and best practices needed for writing shell scripts that contain multiple TenDo commands.

Setting Environment Variables

While reading this manual you might have noticed that most, if not all, TenDo commands contain the same information. For instance, every example TenDo command in this guide contains `-u [USERNAME]` as part of the command. This is fine if you're only sending a single command. But what if you want to use TenDo to send 10 queries to 1010data? Or 100? In such a case, the easiest thing to do is set your username once so you don't have to include it in every single command. To help you do this, TenDo contains numerous Environment Variables that you can set to any value. This means that once you've set these values, you no longer have to include them in subsequent TenDo commands. The following table describes the available TenDo environment variables:

Table 4: A Basic TenDo Command

<table>
<thead>
<tr>
<th>Environment Variable</th>
<th>Equivalent TenDo Option/Switch</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TENTENGW</td>
<td>-g</td>
<td>Sets your 1010data gateway. You need to specify this value if you use a custom corporate gateway to access 1010data, or if you are interested in using specific version of 1010data. The default value is: <a href="https://www2.1010data.com/cgi-bin/gw.k">https://www2.1010data.com/cgi-bin/gw.k</a></td>
</tr>
<tr>
<td>TENTENUID</td>
<td>-u</td>
<td>Your 1010data username. There is no default value for this variable.</td>
</tr>
<tr>
<td>TENTENPW</td>
<td>-p</td>
<td>Your 1010data password. No default for this variable.</td>
</tr>
<tr>
<td>TENTENPROXY</td>
<td>-P</td>
<td>Sets the proxy server for your TenDo session. No default for this variable.</td>
</tr>
<tr>
<td>TENTENSID</td>
<td>-s</td>
<td>Your 1010data Session ID. A unique identifier assigned to you when you begin your 1010data</td>
</tr>
<tr>
<td>Environment Variable</td>
<td>Equivalent TenDo Option/Switch</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>-------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td>session. This can be used as an alternate method to authenticate with 1010data, so long as you have the same session open. We will go into greater detail about Session IDs in the next session.</td>
</tr>
</tbody>
</table>

**Setting Environment Variables in Windows**

In the Windows **Command Prompt**, you set a variable with the **set** keyword. The following command sets the default gateway for your 1010data session in Command Prompt:

```plaintext
C:\> set TENTENGW=www.examplegateway.com
```

Once you send this command to Windows, this environment variable will be available to TenDo as long as you keep your command prompt open. If you include this line as part of a batch script, then the environment variable will be set every time you run the script. You can also permanently add any environment variable to your Command Prompt by going to the **Environment Variables** dialog. This is found on the **Advanced** tab of the **System Properties** dialog.

**Setting Environment Variables in MacOS and in Linux**

In a Linux or Mac shell, you set the environment variables using the **export** keyword, as follows:

```plaintext
$ export TENTENGW=www.examplegateway.com
```

You can also add environment variables to your shell environment permanently by adding an environment variable, like the example above, to your `.profile` or `.bashrc` file.

No matter which operating system you're working with, you should get in the habit of setting your environment variables at the very beginning of your shell script.

**Session management**

**Possessing and terminating an existing session**

When using TenDo, and especially when writing shell scripts that contain TenDo commands, it is important to effectively manage your session status. In some cases you will want to start a new session, while in others you will want to reuse a session. There will also be times when you work with shell scripts that you want to resume an existing session but at some point clear your memory within your sequence of commands. TenDo provides options for all these situations.

This section focuses on two options that will help you manage your TenDo sessions. But first, let's clarify what a **session** is exactly. When you log on to 1010data you are assigned a finite amount of memory as a workspace within which you can open tables, run queries and perform analysis on data. The exact amount of memory you are allocated is dependent on a number of factors, including your company's terms with 1010data. However, regardless of how much memory you have access to one thing is certain: if you do enough work during a session without ending it or clearing the memory you will run out. If you run out of memory you will encounter errors.

**Possessing an existing session**

There are several reasons you may want to resume work on an existing session. First, resuming a session means you don't need to login to 1010data again. Logging in takes both time and server resources, so ideally you will only login to 1010data once during a working session. Secondly, resuming a session allows you to take advantage of work you've already done. In other words, if you've already run a query during the
session your results are cached and the server doesn't need to do that work again. If you run a query and then make small changes to it or add operations, only the new calculations will need to be performed.

**Note:** If you use the -K option and no session exists for your account then a new session will be created.

TenDo allows you to resume a session by "possessing" it. This works whether the session was started in TenDo, the web interface or any other 1010data interface. To possess a session you will include the -K option in your TenDo command, as follows:

```bash
$ tendo -K -u [USERNAME] [TABLE] @[QUERYFILE]
```

The command above will find the open session associated with your user ID and use it to run the queries you send via TenDo. An example of a shell script that uses this technique is provided later in this section.

### Terminating an existing session

As it is sometimes useful to resume an existing session, it is sometimes necessary to end an existing session before beginning work. For example, you may have worked for some time in the 1010data web environment then stopped to work on something else. Now you need to do some work with TenDo, but you're not sure exactly where you left off. In such an instance the best thing to do is to simply terminate your last 1010data session and start a new one. This is done with the -k command. The following example shows the correct syntax:

```bash
$ tendo -k -u [USERNAME] [TABLE] @[QUERYFILE]
```

### Staying logged in and logging out options

The default behavior for TenDo is to logout of 1010data after a command or query has been executed by the system. This is true except when using the -K, -S and -s options to reuse an existing 1010data session. When writing shell scripts the objective is often to execute multiple commands in the same session without logging out. For this, TenDo provides the -L option. This option will keep the account used in the TenDo command where it’s called logged in to the system until otherwise specified. Here’s an example:

```bash
$ tendo -k -L -u [USERNAME] [TABLE] @[QUERYFILE]
```

In this example we terminate any existing session that might be open, start a new session, run a query, and stay logged in to 1010data after finishing.

If you want to override the default behavior of the -K, -S or -s options and log out of 1010data after TenDo is finished executing your query you can use the -l option, as follows:

```bash
$ tendo -K -l -u [USERNAME] [TABLE] @[QUERYFILE]
```

### Memory Management

When you possess a session with TenDo, you also have the option to clear the memory of your session without terminating it. This is done with the -J option. The command would fit into a set of commands as follows:

```bash
$ tendo -J -K -u [USERNAME] [TABLE] @[QUERYFILE]
```

How quickly your workspace memory fills up is a function of both how many queries you run and how complex each query is. You may need to monitor your memory usage and experiment with your script to optimize your memory management.

### Using a Session ID

**Note:** The following information is no longer the best practice with the latest version of TenDo. This is legacy information in case you have an older version of TenDo, or are maintaining legacy scripts that use this technique. However, you should update your TenDo version as soon as possible and use the current best practices and methods as detailed in the previous section.
In 1010data, you don't need to log in to a new session every time you need to use the system. Logging in is an expensive operation. You can avoid login overhead and improve the performance of your scripts by authenticating with a Session ID. Every time you logon to 1010data, whether it's with TenDo, the web-based interface or any other method, you are automatically assigned a Session ID. This ID tells 1010data who you are and what is happening in your 1010data interaction. To retrieve your Session ID, use the -S option in TenDo, as follows:

```
$ tendo -S -u [USERNAME]
```

This command will print your session ID to stdout. Taking this one step further, you can set the TENTENSID environment variable so that it stores your session ID. You do this using a combination of the export or set keyword (depending on your operating system). Here's how you'd do it in a Linux or MacOS shell:

```
$ export TENTENSID=$(tendo -u [USERNAME] -S)
```

This command will assign the results of the command to your TENTENSID environment variable, which can then be used to authenticate for an entire session. Once you've performed this command, your session ID is stored in your shell session. You will not need to provide this information in any TenDo commands until you either end your session or close your shell window. If you want to manually send the session ID, you can send it to 1010data with the -s option. The command looks like this:

```
$ tendo -s $TENTENSID -u [USERNAME] [TABLE] @[QUERYFILE]
```

Once the session ID has been sent to 1010data, you will remain authenticated without the need to provide user credentials with every command. In order to logout of a session, use the -s option in combination with the -l option, as follows:

```
$ tendo -s $TENTENSID -l -u [USERNAME]
```

### Variable Substitution

There's one more important piece of functionality to cover before we get to our example shell scripts, which will tie all these concepts together. TenDo provides the means to define your own variables and their values, then use that definition in your Macro Language query. Let's look at a small example query, and then how would can go about substituting a value in the query with a variable. Here's our code with the value defined in the query itself:

```xml
<sel value="(store=1)"/>
```

If we wanted to define the store value in the TenDo command itself, our Macro Language code would be:

```xml
<sel value="(store=['[STOREVARIABLE']'])"
```

In this example the selection compares the value of store, a column in our table, with our user-defined variable, [STOREVARIABLE]. Now, we can define the value of [STOREVARIABLE] in our TenDo command. This gives us the flexibility to modify our query without the need to edit the actual code. The following example shows how to define the variable in a TenDo command:

```
$ tendo -u [USERNAME] [TABLE] @[QUERYFILE] -[STOREVARIABLE]=2 -o results.csv
```

In this example, TenDo will parse the query you provide and substitute the defined value, 2, for any instance it finds of our [STOREVARIABLE] variable. You can also perform multiple variable substitutions in the same command, as follows:

```
$ tendo -u [USERNAME] [TABLE] @[QUERYFILE] -[STOREVARIABLE]=2 -[ITEMVARIABLE]=3 -o results.csv
```

### Note:

**Macro substitution:**

- `-[V]=X` Substitute X for `[[V]]` in macros (double square brackets required)
- `-[[]]@FN` Read `[[V]]=X` definitions (one per line) from file FN
Example Scripts

At this point we've covered all the concepts necessary for writing TenDo-driven shell scripts. In this section, we'll look at how it all comes together, by looking at an example from both the Windows and Linux/MacOS scripting languages. If you're not comfortable with writing shell scripts, this would be a good time to brush up on the CLI of your operating system of choice.

Windows: A TenDo Batch Script

First, we'll go over an example of a Windows Batch file. Notes on the meaning of the code will be provided in the script's comments. In a Windows batch script, comments are indicated by a leading pair of double colons, like this:

:: comment text goes here

This example will focus on concepts and syntax, without providing many actual values for the variables presented:

:: Define gateway, username, password. If you don't use a custom gateway
:: you can omit the gateway
@echo off
set TENTENGW=%https://www2.1010data.com/cgi-bin/[VERSION]/gw.k
set TENTENUID=%[USERNAME]
set TENTENPW=%[PASSWORD]
echo UID: %TENTENUID% Gateway: %TENTENGW%

:: query to select on a specific store in a chain
echo ^<sel value="(store='[[STOREVARIABLE]]')"/> > [EXAMPLEQUERYFILE].xml
tendo -L -k

tendo -K [TABLE] @[EXAMPLEQUERYFILE].xml -[[STOREVARIABLE]]=3 -o
    store3QueryResults.csv

tendo -K [TABLE] @[EXAMPLEQUERYFILE].xml -[[STOREVARIABLE]]=2 -o
    store2QueryResults.csv

tendo -K [TABLE] @[EXAMPLEQUERYFILE].xml -[[STOREVARIABLE]]=1 -o
    store1QueryResults.csv -l

del [EXAMPLEQUERYFILE]*.xml

Linux and MacOS: A Bash Script

While the syntax of a bash script differs from that of a batch script, the basic concepts for sending a series of TenDo commands are exactly the same regardless of the shell you use. In this version, explanations will also be made in the comments of the code. In a bash script, comments are indicated by the pound (#) sign, except for the very first # sign in the line:

#!/bin/bash

# Define gateway, username, password. If you don't use a custom gateway
# you can omit the gateway
export TENTENGW=https://www2.1010data.com/cgi-bin/[VERSION]/gw.k

The above line is a special directive to the operating system and should be ignored for the purposes of this TenDo manual.

#!/bin/bash
# Define gateway, username, password. If you don't use a custom gateway
# you can omit the gateway
export TENTENGW=https://www2.1010data.com/cgi-bin/[VERSION]/gw.k
export TENTENUID=[USERNAME]
export TENTENPW=[PASSWORD]

echo "UID:" $TENTENUID " Gateway:" $TENTENGW

# query to select on a specific store in a chain
echo "<sel value="(store='[[STOREVARIABLE]]')\"/"/>" > [EXAMPLEQUERYFILE].xml

tendo -L -k

tendo -K [TABLE] @[EXAMPLEQUERYFILE].xml -[[STOREVARIABLE]]=3 -o store3QueryResults.csv

tendo -K [TABLE] @[EXAMPLEQUERYFILE].xml -[[STOREVARIABLE]]=2 -o store2QueryResults.csv

tendo -K [TABLE] @[EXAMPLEQUERYFILE].xml -[[STOREVARIABLE]]=1 -o store1QueryResults.csv -l

rm [EXAMPLEQUERYFILE]*.xml
Every transaction sent to the 1010data Insights Platform returns an XML response containing at least a return code and its corresponding message. The error codes and messages are described here.

## Return codes

A nonzero value in the `<rc>` tag indicates there was an error in processing your transaction. The error can be related to the syntax or semantics of the XML content submitted to the server. The codes are designed to give programs the ability to gracefully deal with errors without having to interpret the contents of the `<msg>` tag. Possible return codes are as follows:

### Table 5: Return codes and messages

<table>
<thead>
<tr>
<th>Error code <code>&lt;rc&gt;</code></th>
<th>Message <code>&lt;msg&gt;</code></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Success</td>
</tr>
<tr>
<td></td>
<td>Your transaction completed successfully</td>
</tr>
<tr>
<td>1</td>
<td>Unclassified error</td>
</tr>
<tr>
<td>2</td>
<td>XML error</td>
</tr>
<tr>
<td>3</td>
<td>Missing user identification</td>
</tr>
<tr>
<td></td>
<td>You did not provide a UID in the POST request.</td>
</tr>
<tr>
<td>4</td>
<td>Invalid user identification</td>
</tr>
<tr>
<td></td>
<td>The user ID you provided was invalid.</td>
</tr>
<tr>
<td>5</td>
<td>Already logged in</td>
</tr>
<tr>
<td></td>
<td>The user ID specified already has an established 1010data session.</td>
</tr>
<tr>
<td>6</td>
<td>Missing transaction type</td>
</tr>
<tr>
<td>7</td>
<td>Invalid transaction type</td>
</tr>
<tr>
<td>8</td>
<td>Missing element</td>
</tr>
<tr>
<td>9</td>
<td>Invalid element value</td>
</tr>
<tr>
<td>10</td>
<td>Invalid element contents</td>
</tr>
<tr>
<td>11</td>
<td>Missing attribute</td>
</tr>
<tr>
<td>12</td>
<td>Invalid attribute name</td>
</tr>
<tr>
<td>13</td>
<td>Invalid attribute value</td>
</tr>
<tr>
<td>14</td>
<td>Invalid directory name</td>
</tr>
<tr>
<td>15</td>
<td>No such directory</td>
</tr>
<tr>
<td>16</td>
<td>Invalid table name</td>
</tr>
<tr>
<td>17</td>
<td>No such table</td>
</tr>
<tr>
<td>18</td>
<td>No such directory or table</td>
</tr>
<tr>
<td>19</td>
<td>Directory already exits</td>
</tr>
<tr>
<td>20</td>
<td>Table already exits</td>
</tr>
<tr>
<td>Error code &lt;rc&gt;</td>
<td>Message &lt;msg&gt;</td>
</tr>
<tr>
<td>----------------</td>
<td>---------------</td>
</tr>
<tr>
<td>21</td>
<td>No query specified</td>
</tr>
<tr>
<td>22</td>
<td>Not currently implemented for Quick Queries</td>
</tr>
<tr>
<td>23</td>
<td>Too many values</td>
</tr>
<tr>
<td>24</td>
<td>Empty table</td>
</tr>
<tr>
<td>25</td>
<td>Problem saving table</td>
</tr>
<tr>
<td>26</td>
<td>Cannot save into 'uploads' directory</td>
</tr>
<tr>
<td>27</td>
<td>Problem deleting table</td>
</tr>
<tr>
<td>28</td>
<td>Problem saving file</td>
</tr>
<tr>
<td>29</td>
<td>Duplicate column names</td>
</tr>
<tr>
<td>30</td>
<td>Duplicate table names</td>
</tr>
<tr>
<td>31</td>
<td>No FTP permission</td>
</tr>
<tr>
<td>32</td>
<td>Not enough available space left in your account.</td>
</tr>
<tr>
<td>33</td>
<td>Problem moving table or directory</td>
</tr>
<tr>
<td>34</td>
<td>Problem modifying table or directory attributes</td>
</tr>
<tr>
<td>35</td>
<td>Not Logged in</td>
</tr>
<tr>
<td>36</td>
<td>&lt;cols&gt; contains columns that do not appear in result</td>
</tr>
<tr>
<td>37</td>
<td>You do not own the specified group</td>
</tr>
<tr>
<td>38</td>
<td>System is busy</td>
</tr>
<tr>
<td>39</td>
<td>You are not at an authorized IP address</td>
</tr>
<tr>
<td>41</td>
<td>1010 data is unavailable</td>
</tr>
</tbody>
</table>

A nonzero value in the <rc> tag indicates there was an error in processing your transaction. The error can be related to the syntax or semantics of the XML content submitted to the server. The codes are designed to give programs the ability to gracefully deal with errors without having to interpret the contents of the <msg> tag.
Support Requests

If you are submitting queries via TenDo or running TenDo commands in a shell script and are experiencing errors or unexpected behavior, 1010data offers full support. In order to submit a support request, there are some specific pieces of information we need in order to help. This section contains instructions for properly obtaining and submitting the files and information necessary to fulfill a support request.

Required Files

Depending on the exact issue you're experiencing there are a number of files that must be provided.

All support requests require the following information:

- An XML log file
- A TenDo debugging log file
- The query you submitted when you experienced the problem

Note: If you are reading the query from a file then you should include that file in your support email.

Additionally, if you were executing TenDo commands via a shell script, you will need to provide your shell script code.

Creating the XML Log File

If you are experiencing problems with a single TenDo command, or you have written a shell script and know exactly which TenDo command in the script is causing the issue, then you should create entirely new log files using the commands in the following examples.

The first file you need to provide in a support email is the log of the XML interactions between TenDo and 1010data. This is done with the --xmllog option. This option requires the location of a text file where the log will be stored, as follows:

```
$ tendo -u [USERNAME] --xmllog=[XMLLOGFILEPATH].1010xml
```

In the example above, the log of XML interactions between TenDo and the database will be saved to a text file at the path you specify. You can name this file anything you like. However, you should give the file a name that indicates it contains the XML log information, as opposed to the TenDo debugging output.

Creating the TenDo Debugging File

In addition to the XML log file you must also provide the TenDo debugging logs in a separate file. This is done with a combination of TenDo’s -v option and your shell’s ability to output the results of a command to a file.

```
$ tendo -u [USERNAME] [TABLE] @[QUERYFILE] -vvv 2> [TENDOLOGFILEPATH].1010log
```

In the example the -v option is used three times in order to increase the level of detail that TenDo will provide in the log information. You should always use -vvv when creating a log to send in a support request. By default, the -vvv option will output TenDo debugging information to stderr, so the next part of the command, 2> [TENDOLOGFILEPATH].1010log outputs this information directly to a text file instead.

You should combine both commands into a single statement so that both files are created at the same time. When you do so, the command will look like this:

```
$ tendo -u [USERNAME] [TABLE] @[QUERYFILE] --xmllog=[XMLLOGFILEPATH].1010xml --
     -vvv 2> [TENDOLOGFILEPATH].1010log
```

As stated at the beginning of this section, the command above should be used if you are certain of the exact TenDo command that is causing the issue. For instance, if you write a shell script that contains 50
commands and determine that the 30th command is causing an issue, you should use the command in the previous example. However, if you can’t isolate the exact command that is causing the issue you will need to append the log information for every command into the same file. This command is very similar to our previous example but contains the append commands `+=` and `2>>` respectively, as follows:

```bash
$ tendo -u [USERNAME] [TABLE] @[QUERYFILE] --xmllog+=[XMLLOGFILEPATH].1010xml -vvv 2>> [TENDOLOGFILEPATH].1010log
```

Once you have created files with the necessary log information you can submit a support request to 1010data. To do this, send an email to support@1010data.com. Again, make sure you provide the following information with your support request:

- The XML log file
- The TenDo debugging log file
- The query you are running with TenDo. If you are reading the query from a file you should provide the file.
- The shell script code you are using (when applicable)