1010data Excel Add-in User’s Guide
Contents

Introduction ................................................................................................................................. 3

General .......................................................................................................................................... 4
  Install the 1010data Excel Add-in .......................................................................................... 4
  1010data Excel Add-in Setup .................................................................................................. 4
  Determine Your Version of Excel ............................................................................................ 5
  Activate the 1010data Excel Add-in ...................................................................................... 5
  Deactivate the 1010data Excel Add-in .................................................................................. 6
  Log in from the 1010data Excel Add-In ................................................................................. 6
  Run Queries .............................................................................................................................. 7
    Run Queries in Multiple Workbooks ...................................................................................... 7
    Query Progress ...................................................................................................................... 7
  Upload Data .............................................................................................................................. 9
    Uploading Sheets .................................................................................................................. 9
  Change the default version ....................................................................................................... 10
  Options ................................................................................................................................... 12

Advanced ..................................................................................................................................... 16
  General Guidelines .................................................................................................................. 16
  Enable Advanced Features ...................................................................................................... 16
  Define Queries .......................................................................................................................... 16
    Query Sheets .......................................................................................................................... 16
    Tips .......................................................................................................................................... 17
  VBA Function Library ............................................................................................................... 19
    Login1010() .......................................................................................................................... 20
    Logout1010() .......................................................................................................................... 20
    ShowLoginWindow() .............................................................................................................. 21
    RunQSheet() .......................................................................................................................... 21
    RunQSheets() ......................................................................................................................... 21
    RunActiveQSheet() .................................................................................................................. 22
    ClearCache() .......................................................................................................................... 22
    IsSessionCurrent() .................................................................................................................. 22
    GetSession() .......................................................................................................................... 23
    ListDir() .................................................................................................................................. 23
    LoadUSheet() .......................................................................................................................... 24
    LoadActiveUSheet() ................................................................................................................. 24
    DebugLog1010() ...................................................................................................................... 25
    NewestVersion() ...................................................................................................................... 26

Troubleshooting and Support .................................................................................................... 27
  Determine the Latest Version ................................................................................................. 27
  Submit a Support Request ....................................................................................................... 28
    Enabling Debug Logging ....................................................................................................... 28
    Debug Logging ....................................................................................................................... 29
Introduction

The 1010data Excel Add-in is a utility that enables Microsoft Excel to communicate directly with the 1010data server and extract data into Excel. An Excel workbook can contain multiple queries that use the 1010data macro language. Queries are sent by Excel to the 1010data server, and results are retrieved to the specified destination within the workbook. The queries embedded within a workbook can be hidden from view, providing a seamless experience for the user.

This user guide is divided into three sections:

1. **General** - for all users
2. **Advanced** - for advanced users who will be constructing workbooks or templates with embedded queries or data to upload
3. **Troubleshooting and Support** - for users experiencing errors or unexpected behavior
This section covers the basic functionality when using templates or workbooks that already contain embedded 1010data queries (prepared by other users).

Install the 1010data Excel Add-in

The installation of the 1010data Excel Add-in is done through an installation wizard in a few easy steps. The 1010data Excel Add-in is only compatible with Excel 2010 and Excel 2013 and runs on 32-bit or 64-bit Windows systems. The version of the Excel Add-in you install should match the version of Excel you are running (i.e., the 32-bit version of the 1010data Excel Add-in should be installed if you are running the 32-bit version of Excel). To determine which version of Excel that you are running, see Determine Your Version of Excel on page 5.

Note: You must close Excel and all Windows Explorer windows before installing the 1010data Excel Add-in.

To install the 1010data Excel Add-in:

1. Go to https://www.1010data.com/support/technical-interfaces/excel-add-in and download the 1010data Excel Add-in installer by clicking either Addin 2.0 for 32-bit/64-bit Excel 2010 and 2013 or Addin 1.0 for Excel 2007 (depending on the version of Excel you are running).

2. Run the installer and follow the instructions in the installation wizard. See 1010data Excel Add-in Setup on page 4.

When installation is complete, you should see a menu titled 1010data under the Add-ins tab on the ribbon. After installation, you may need to activate the 1010data Excel Add-in. See Activate the 1010data Excel Add-in on page 5.

1010data Excel Add-in Setup

The 1010data Excel Add-in Setup Wizard guides you through the installation of the 1010data Excel Add-in. When the installer is run, the user is presented with the 1010data Excel Add-in Setup Wizard.

Click Next > to continue.
On the **Choose Install Location** page, you can specify the folder where the 1010data Excel Add-in will be installed or accept the default location.

**Note:** The 1010data.xlam and lib1010vbasdk.dll files will be installed in this folder. If you move or delete these files, the 1010data Excel Add-in will stop working.

Click **Install** to commence the installation. The following files will be installed in the destination folder:
- 1010data.xlam
- lib1010vbasdk.dll
- ExcelAdd-inUserGuide.pdf
- 1010data_Excel_Add-in_Example.xltx

When the installation is complete, click **Finish** to close the wizard.

**Note:** After the 1010data Excel Add-in has been successfully installed, you may need to manually activate it in Excel.

---

**Determine Your Version of Excel**

The version of the 1010data Excel Add-in should match the version of Excel you are running on your system.

To determine the version of Excel that you are running:

1. Check to see whether you are running the 32-bit or 64-bit version of Excel:

<table>
<thead>
<tr>
<th>Version</th>
<th>Procedure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Excel 2013</td>
<td>Click Account &gt; About</td>
</tr>
<tr>
<td>Excel 2010</td>
<td>Click File &gt; Help</td>
</tr>
</tbody>
</table>

2. Note the version displayed in the resultant dialog.

---

**Activate the 1010data Excel Add-in**

In addition to installing the 1010data Excel Add-in, you may also need to manually activate it in Excel.

The 1010data Excel Add-in must be installed before it can be activated. See *Install the 1010data Excel Add-in* on page 4.
To activate the 1010data Excel Add-in:

1. Open Excel, click **File > Options > Add-ins**.
2. From the **Manage** drop-down list box, select **Excel Add-ins** and click **Go**.
3. From the **Add-Ins** dialog box, ensure **1010data Excel Add-in** is selected from the list of available add-ins.
   
   If you have a previous version of the 1010data Excel Add-in installed:
   a) Clear the **1010data Excel Add-in** check box.
   b) **Browse** to the directory containing **1010data.xlam** (which is typically located under **AddIns \1010data Inc\1010data Excel Add-in**).
   c) Double-click the **1010data.xlam** file.
   d) Click **Yes** when asked if you want to replace the file.
   e) Ensure **1010data Excel Add-in** is selected from the list of available add-ins.
4. Click **OK**.
   
   Installation is complete. You should now see a menu titled **1010data** under the **Add-ins** tab on the ribbon.

### Deactivate the 1010data Excel Add-in

Deactivating the 1010data Excel Add-in removes access to its functionality from Excel.

To deactivate the 1010data Excel Add-in:

1. Open Excel, select **File > Options > Add-ins**.
2. Click on the **Manage** drop-down menu, select **Excel Add-ins**, and click **Go**.
3. From the **Add-Ins** dialog box, clear the **1010data Excel Add-in** check box in the list of available add-ins.
4. Click **OK**.
   
   The add-in is uninstalled.

Note: The **1010data** menu under the **Add-ins** tab on the ribbon may remain until you quit Excel; however, the **1010data** menu will not appear the next time Excel is started.

### Log in from the 1010data Excel Add-In

You must log in to the 1010data server from the 1010data Excel Add-in in order to run queries or upload data.

To log in to the 1010data server from the 1010data Excel Add-in:

1. From the **1010data** menu, click **1010data Login**.
2. Enter your 1010data **Username** and **Password**. If you have forgotten your password, click on **Forgot your password?** to be directed to a webpage where you can submit your user name and email address to receive an email containing a link to reset your password.
3. If you are already logged in to another 1010data session, you may choose one of the three options from the **If account is in use** drop-down list box:
   - **Enter existing session**: You may enter an existing session that was started through the 1010data web interface or another client-facing application. Sometimes it is useful to be logged in to 1010data via the web interface and Excel at the same time. Once you successfully import your session into Excel, you’ll be able to run queries both in Excel and in your browser with the same account without the need to establish a separate login.
     
     If you choose **Enter existing session** and one does not already exist, a new one will be created.
• **End existing session**: You may end the session currently running on 1010data and start a new one.

  If you choose **End existing session** and there is no existing session, a new one will be created regardless.

• **Do not log in**: You may choose not to log in. There may be times when you neither want to enter nor end an existing session. For instance, you may have queries that run at specific times on the 1010data server that you do not want to interrupt. If the 1010data Excel Add-in finds that a 1010data session already exists, you will be notified of this via a dialog box, and no action will be taken.

4. Press the **Secure Login** button. While the login is attempted, you should see the message **Logging in to the 1010data server...** in the bottom left corner of the Excel window.

When the login is successful, you will see a message in the bottom left corner of the Excel window indicating the user is logged into 1010data.

  **Note**: Be careful logging in or out of the web interface while you are using the Add-in. If you do this, you will be logged out of the Excel Add-in. If this happens, you will need to log in from the Add-in again before you can interact with 1010data from Excel.

---

**Run Queries**

Embedded 1010data queries can be run directly from Excel using the 1010data Excel Add-in.

To run a query in the active workbook:

1. **Specify any required inputs.** A workbook containing 1010data queries will likely require input from the user to define the report criteria.

2. **Press CTRL+Q** (or click **1010data > Run Queries > In Active Workbook**). If you have not yet logged into 1010data from Excel, you will be prompted to log in. **WARNING**: If you have an existing 1010data session in a web browser, logging in from Excel may kill your existing session. (See **Log in from the 1010data Excel Add-In** on page 6.)

Query results will appear in the workbook as queries are executed. The results are directly downloaded from 1010data to Excel as compressed binary data. The workbook can be viewed and browsed while queries are being processed. The **Query Progress** dialog box (see **Query Progress** on page 7) will indicate the state of query execution.

  **Note**: If you submit a request, whether via Excel or the browser, the system will be “busy” and appear unresponsive until prior requests have completed.

---

**Run Queries in Multiple Workbooks**

You can run queries in multiple open workbooks at the same time.

To run queries in all open workbooks:

1. **Open all workbooks containing queries that you would like to run.**

2. **Click 1010data > Run Queries > In All Open Workbooks.**

---

**Query Progress**

The **Query Progress** dialog box shows the percentage of processing completed for queries running in the current workbook.

While queries are running, a dialog box is displayed showing two progress bars:

- **The percentage of queries completed within the current workbook**
- **The percentage of total operations within the current query**
The operations that are included are:

- Tabulations
- Selections
- Merges
- Link and selects
- Aggregations on the data

The **Query Progress** dialog box also shows the amount of available workspace as the query is being processed. 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.)

If you click the **Stop All Queries** button while you are running queries, an attempt will be made to stop the query on the server. When complete, all Excel processing stops.

When all the queries have completed successfully, you will see the following dialog box:
Upload Data

Data to be uploaded to the 1010data server from the 1010data Excel Add-in is specified using an upload sheet (or u-sheet).

To upload data from the 1010data Excel Add-in to the 1010data server:

1. From the 1010data menu, click Upload Data > Add New U-Sheet.
2. Enter all required information into the input cells in the u-sheet. (See Uploading Sheets on page 9.) Hover your mouse over any input cell with a red triangle in the top right corner to see a tooltip containing information defining the required input.

Uploading Sheets

Use an upload sheet (or u-sheet) to upload data to the 1010data server using the 1010data Excel Add-in. A u-sheet can be used to upload a new table, replace an existing table, or append to a specified table on 1010data.

Table Title

(required) The title displayed for the table, which is used to help describe the contents (e.g., “Sales Detail by Customer”). May contain any combination of uppercase and lowercase letters, numbers, spaces, and special characters.

Table Name (path)

(required) The name of the table including the full path (e.g., acmeco.myfolder.mytable). The table name cannot contain any uppercase letters, spaces, or special characters.
Note: You cannot upload to the My Data folder. Also, the path you specify must be writable by you.

**Short Description** *(optional)* This short description appears next to the table title.

**Permissions** *(optional)* If blank, only the owner has permission to see the table. Otherwise, this can be a comma-separated list of user names or user groups who will have permission to see the table (e.g., list:username1, usergroup3...username6), or enter inherit so the table inherits the permissions of its parent folder.

**Replace or Append?** *(optional)* Can be blank, replace, or append

- If blank, a new table on 1010data is created.
  
  **Note:** If a table with the same name (path) exists, the upload will fail.

- If replace, the uploaded table will replace any table found with the same name (path).

- If append, the contents of the upload will be appended to any table found with the same name (path).

**Merge?** *(optional)* Enter TRUE in this field if this table will need to be merged with other tables in the future. Otherwise, this field should be left blank.

**Column Names:** *(required)* The proper name for each column (to be used in formulas). No capital letters, spaces, or symbols allowed.

**Column Types:** *(required)* Must be one of the following:

- a - (alphanumeric) For any column containing non-numeric data (except dates, use i)
- i - (integer) For numbers not containing decimals (also for dates)
- f - (float) For floating point numbers (i.e., numbers containing decimals)

**Column Formats:** *(optional)* Tells 1010data how to display the data (data display type, decimal places, width). For more information on supported formats, see Data Types and Display Formats in the 1010data User’s Guide.

**Column Headings:** *(optional)* The heading for each column to be displayed in table view. May contain spaces or special characters such as $, %, etc. If omitted, the column names will display instead.

**Table Data:** *(optional)* The actual data to be uploaded. The first blank row indicates the end of the table. Rows containing only blank values are not allowed in the middle of a table. WARNING: Data must not exceed 1,000,000 cells.

### Change the default version

You can change the default version of the 1010data Insights Platform that you log into via the 1010data Excel Add-in.

The default version setting is in the Options menu.

1. On the Excel ADD-INS tab, click **1010data > Options**.
Excel opens the **1010data Excel Addin Options** window.
For information on the 1010data Excel Add-in Options window, see Options on page 12.

2. Click the Version field. Default is automatically selected.
3. Enter the version number.
   
   Example version numbers:
   - beta-latest
   - prime-10.43
   - prime-latest
   - prod-9.64

4. Click Save Settings.
   The new default version is saved and the 1010data Excel Add-in Options closes.

   Note: If you enter a Beta version, e.g., beta-latest, the 1010data Excel Add-In displays a disclaimer.
Options

The **1010data Excel Add-in Options** window contains various settings, such as the default version and query timeout value. This window can be displayed by clicking **Options** from the 1010data Excel Add-in menu, which is accessible on the **ADD-INS** tab.

![1010data Excel Addin Options window](image)

**Figure 1: 1010data Excel Addin Options window**

**Gateway server**

Sets the gateway URL the user uses to connect to the 1010data Insights Platform.

**Connect to 1010data via a Proxy**

Indicates that the connection to the Insights Platform is made via a proxy. Selecting this option activates the proxy options.

**Proxy URL**

Specifies the URL to connect to 1010data via a proxy. This field is available when the **Connect to 1010data via a Proxy** option is selected.

A proxy URL is in this format: `http://[USER]@[HOST]:[PORT]`.

The values specified by `[USER]`, `[HOST]`, and `[PORT]` represent the proxy user name, the proxy address, and the port to connect to, respectively. `[USER]` and `[PORT]` are optional.

**Proxy password options**

Specifies options related to the proxy password. These options become active when the **Connect to 1010data via a Proxy** option is selected.

The options for entering a proxy password are:

- **Remember Proxy Password**
- **Prompt for Proxy Password at Login**
• **Proxy Does Not Require Password**

Selecting the **Remember Proxy Password** option activates a field to enter the password.

Click the **Show Password** option to display the password instead of asterisks.

**Version**

Controls which version of the Insights Platform the 1010data Excel Add-In logs in to. The **Default** option is selected automatically.

For more information, see *Change the default version* on page 10.

**Default Column Headers**

Controls what is displayed in column headers, which are at the top of each column in a table.

There are five options:

**None**

Does not display either the column name or column label in the column header.

**Column Labels**

Displays only the column label in the column header.

Labels are optional descriptive column titles. They may contain any combination of uppercase and lowercase letters, numbers, spaces, and special characters. Column labels may contain any combination of:

- Uppercase and lowercase letters \(a-z, A-Z\)
- Numbers \(0-9\)
- Spaces
- Special characters: `! @ # $ % ^ & * ( ) _ + - = : ; { } [ ] | < > , . ? ' " ~ ` `/`

For example, *Product ID, 98430 Total, and *Stores are all valid column labels.

You can have a multi-line column label, use the backtick character (``) to separate the lines, e.g., *Percentage`Total Sales (%)*.

If the column does not have a label, the column's name is displayed instead.

**Column Names**

Displays only the column name in the column header.

Names are required, single-word identifiers for columns. They are used to identify the column in value expressions and selection expression. Column names:

- Must be a single word consisting of alphanumeric characters \(a-z, 0-9\) or underscores \(_\).
- Must start with a letter.
- Must be lowercase.
- Must not end with an underscore.
- Must not contain any spaces or other special characters.

For example, *productid or product_id are valid column names, but Product Id is not.*

**Both: Labels; Names**

Displays the column label above the column name in the column header.

**Both: Names; Labels**

Displays the column name above the column label in the column header.

**Note:** You may also change this option individually on each sheet.

**Default Data Format**
Controls the format of the results returned from 1010data.

There are three options:

**Formatted**

Returned results appear in the format defined in the Macro Language.

For example, if the value is stored on the platform as 1980.239049 but the Macro Language specifies format="dec:0;type:num", the result is formatted as 1,980, and Excel displays 1,980.

For more information, see *Display formats*.

**Unformatted**

Returned results appear as Excel values without any formatting.

For example, if the value is stored on the platform as 1980.239049, Excel displays 1980.239.

**Raw**

Returned results appear in the format that the values are stored in. No formatting from Excel or the Insights Platform is applied.

For example, if the value is stored on the platform as 1980.239049, Excel displays 1980.239049.

**Note:** You may also change this option individually on each sheet.

**Auto Filter Options**

Determines how the 1010data Excel Add-in handles unmanageable filters that cannot be reapplied to the data.

When Excel implements filters, it alters the indexing of the cells it is applied to, and the Excel Add-in cannot just copy the range the filter is on. Instead, the filters are recorded and removed, the data is populated, and the filters are reapplied. Not all filters are able to be reapplied. These are called unmanageable filters.

There are two options for what the Add-in should do if it encounters unmanageable filters:

**Report Error**

The Excel Add-in stops and returns an error.

**Remove Filter**

The Excel Add-in populates the data and doesn't reapply the unmanageable filter.

**Query Timeout**

Sets the maximum number of minutes to wait for a query to complete.

**Use manual retry**

Retries the connection at a certain interval specified by the **Local Timeout**. This is primarily used for connecting via a proxy. It keeps the connection thru the proxy alive while the query may still be running.

**Local Timeout**

Sets the maximum number of minutes to wait between each manual retry.

The minimum timeout value is five minutes. The retries continue until the value specified by the **Query Timeout** is reached.

For example, if the **Local Timeout** is 5 and the **Query Timeout** is 45, the Excel Add-in will continue to manually retry every 5 minutes until either 45 minutes has elapsed or the query completes.
Prompt to Save Session Before Close

Prompts the user to save the current session before closing the window.

**Note:** This prompt only appears when closing the last open workbook, even if that workbook does not have an 1010data Excel Add-in Q-Sheet.

Download Column Labels with newlines replaced by spaces

Changes newline indicators in column labels to spaces.

A backtick (`) within a column label indicates a newline.
Advanced

The 1010data Excel Add-in can be used to embed 1010data queries within Excel workbooks (or templates). It is assumed that the reader is experienced with the 1010data macro language and with Excel, and is comfortable with Excel features, such as working with named ranges and concatenating text strings. For help with the 1010data macro language, consult the online help files within the 1010data web interface.

General Guidelines

Advanced users who will be embedding queries into Excel files should keep certain criteria in mind with respect to those files.

An Excel file containing embedded queries should ideally:

- Be saved as an .xlxt (Excel template) file and set as read-only (When the file is opened, the user will get a fresh copy of the file and will be prevented from overwriting the original file.)
- Contain one or more output or report worksheets (tabs)
- Contain one input or control worksheet, where the report user defines the values of the variables used by the queries
- Contain one or more query sheets (or q-sheets), where the queries are defined in the 1010data macro language, referencing the values specified on the input sheet
- Optionally contain a worksheet where raw data resulting from the queries will be pasted, to be referenced by formulas on the output sheets

For presentation, all worksheets except for the output sheets can be hidden.

Enable Advanced Features

Users must enable the advanced features of the 1010data Excel Add-in to see those features in the menu options.

To enable advanced features:

1. Select Options... from the 1010data menu.
2. Select Show Advanced Menu Options in the dialog box.
3. Click Save Settings.

The menu options for building templates that use the 1010data Excel Add-in will be displayed.

Define Queries

In order to submit a query to the 1010data server using the 1010data Excel Add-in, you must fill out a query sheet.

To define a query in the 1010data Excel Add-in:

1. Press CTRL-T or click 1010data > Add New Q-Sheet.
2. Enter all required information into the input cells in the q-sheet. (See Query Sheets on page 17.)
   Hover your mouse over any input cell with a red triangle in the top right corner to see a tooltip containing information defining the required input.

The new q-sheet will appear after the active worksheet. Once the query is defined, you may run the query. (See Run Queries on page 7.)

Note: You may rename the worksheet, but ensure that the first 6 characters (\_1010q) of the worksheet name remain intact. This prefix is used to identify the worksheet as a q-sheet.
Query Sheets

Users must fill out a query sheet in order to submit queries to the 1010data server using the 1010data Excel Add-in.

Each query in a workbook is defined on a *query sheet* (or *q-sheet*). There is no external limit imposed on the number of q-sheets allowed in a workbook (i.e., it is subject only to Excel’s limit on worksheets).

**Query Description:** *(optional)* This name will display in the progress bar as the queries are being processed. *(Recommended, for user documentation only.)*

**To be Applied to Table:** *(required)* The full name (including path) of the 1010data table to which the query is to be applied. Example: `pub.doc.retail.salesdetail` *(no uppercase letters, spaces, or special characters)*

**Note:** Table name is not the same as table title. The table name is used when referring to the table in queries and cannot contain any uppercase letters, spaces, or special characters. The table title is a string that is used to help describe the contents of a table (e.g., "Sales Detail by Customer")

**1010data Macro Code:** *(required)* The 1010data macro code that defines the query is entered here. Code can be copied from a text file (or from the Edit Actions panel in the 1010data web interface) and pasted into this area. Query code can include Excel formulas that create dynamic code dependent on variables and data entered on the input worksheet. The query will run until the first blank line is encountered. Ensure that there are no unintentional blank lines to guarantee that the entire macro will be processed. Blank lines can be used as breakpoints to assist in debugging erroneous macros.

**Note:** The text alignment often gets corrupted when text is pasted here. This can be corrected by left-justifying the selected text after it is pasted.
Result Destination: *(required)* A reference (an address or named range) to the top left cell (one cell only, in the same workbook) of the range where the query results are to be pasted. Example: Sheet1!$A$2

*Note:* This cell must contain a valid reference to a cell in the same workbook. Also, you should ensure there is sufficient room at the destination so that query results don't overwrite the results of other queries or data.

Max Rows to Retrieve: *(optional)* Allow result rows to be "capped" at a given number of rows. Where \( N \) is specified, the first \( N \) rows in the query results will be retrieved; if \( N \) exceeds the number of rows in the query results, all rows will be retrieved. If left blank, all the rows resulting from the query will be retrieved.

*Note:* This entry must be either blank or a number between 1 and the number of rows available on the worksheet at the destination specified.

Column Headers: *(required)* Defines whether the data will be returned with a header row and, if so, what type of header. Select one of the following from the drop-down list: None, Column Names, or Column Labels.

*Note:* Select Options from the 1010data menu to change the default value for this cell.

Data Format: *(required)* Defines what type of formatting will be applied to the query results. The entry must be either Formatted, Unformatted, or Raw. Select the desired option from the drop-down list:

- **Formatted** - Formatted data will appear in the number formats defined in the 1010data macro code and, when retrieved, will overwrite the formatting of the destination cells.
- **Unformatted** - Unformatted data will appear as Excel values without any formatting (i.e., dates will be in the internal Excel format) and, when retrieved, will not overwrite formatting of destination cells.
- **Raw** - Raw data will appear as Excel values without any formatting, but data will be in native 1010data format; formats defined in the 1010data macro code will be ignored (i.e., dates will appear in YYYYMMDD format). Retrieved data will not overwrite formatting of destination cells.

Query Enabled: *(required)* Allows or prevents a query from running. Must be either TRUE or FALSE. Defaults to TRUE. To prevent a particular query from running, set to FALSE (useful while developing or debugging a template).

Clear range before pasting: *(optional)* An address range (or defined name) specifying cells to be deleted before the query results are pasted. This is useful for clearing the results of a previous run of the query. Only contents (not formatting) will be cleared.

Example: Sheet1!$A$2:$D$40

*Note:* If populated, this cell must contain a valid reference to a range of cells in the same workbook.

**WARNING:** Double-check input before running; the contents of the range referenced here will be deleted without recourse.

**Tips**

You may want to incorporate these suggestions when using Excel formulas within queries.
1010data Excel Add-in User's Guide | Advanced | 19

1010data macro code can include Excel formulas that create dynamic code dependent on variables and data entered on the input worksheet. Keep these points in mind as you define these formulas:

- Define Excel names for input cells to be referenced in formulas that evaluate to dynamic macro code (code dependent on Excel formulas and other cells).
- Use Excel's `TEXT()` function to properly display the value of dates.
- If a line of dynamic macro code will be used in more than one q-sheet, define this as a formula in a centralized location, such as a dedicated worksheet or the input worksheet. Define a name for each cell that contains such a formula.
- Use the `&` operator to concatenate string (text) values.
- Use 2 double quotes in a row (""') to tell Excel to ignore the special meaning of a single double quote ('').

A number of these recommendations are demonstrated in the following example.

**Example**

1. For a date range selection, define input cells named `From` and `To`.
2. In a cell named `DateSelector`, enter a formula such as:
   
   ```excel
   ="<sel value=""between(date; &TEXT(From,"yyyymmdd")&"; &TEXT(To,"yyyymmdd")&")""'/>"
   ```

3. At the desired line within your macro code, enter:
   
   ```excel
   =DateSelector
   ```

Given dates `From"1/1/2007"` and `To"2/1/2007"`, this formula will evaluate to:

```xml
<sel value="between(date; 20070101; 20070201)"/>
```

**Note:** It is advisable to first run your macro within the web interface to ensure desired results are obtained, then copy the macro code into your Excel workbook, replacing the lines of code that are to be dynamic with the proper formulas or names.

**VBA Function Library**

The user can interact with 1010data using Visual Basic for Applications macros through a set of functions supplied by the 1010data Excel Add-in.

The 1010data Excel Add-in provides a number of VBA functions that can be called to execute queries programmatically from VBA macros. These allow the user to run a complex set of queries with dynamic input or to create jobs that can be scheduled to run at designated times. Using these functions, you can log in or out of 1010data, run queries, clear the cache, or upload data.

To call these functions programmatically, your VBA project must have a reference to the 1010data Excel Add-in. To add the reference, click `Tools > References` from the VBA menu bar, select `A1010data` from the list of available references, and click OK.
Login1010()

Instantiates a 1010data API session.

Syntax

Public Function Login1010(UserID As String, LoginPW As String, NoKill As Boolean, Possess As Boolean) As Integer

Input

UserID 1010data user ID
LoginPW 1010data password
No Kill (optional) If True, will not terminate existing session for this user ID (by default, existing session will be terminated upon new login).
Possess (optional) If True, will enter an existing 1010data session. If no session exists, a new one is created.

Return value

0 success
1 failure
-1 fatal error

Logout1010()

Ends a 1010data API session.

Syntax

Public Sub Logout1010(Optional Quiet As Boolean)
**Input**

**Quiet** *(optional)* If True, will not prompt user or display interactive error messages.

---

**ShowLoginWindow()**

Displays the login dialog box prompting the user to login.

**Syntax**

```vba
Public Sub ShowLoginWindow()
```

---

**RunQSheet()**

Validates and runs the specified query sheet.

**Syntax**

```vba
Public Function RunQSheet(QSheet as Worksheet, Optional Quiet As Boolean) As Integer
```

**Input**

**QSheet** Must be a Worksheet object referencing the q-sheet to be run.

**Quiet** *(optional)* If True, will not prompt user or display interactive error messages.

**Return value**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>success</td>
</tr>
<tr>
<td>1</td>
<td>failure</td>
</tr>
<tr>
<td>-1</td>
<td>fatal error (further query attempts are likely to fail)</td>
</tr>
</tbody>
</table>

**Example**

Example usage:

```vba
result = a1010data. RunQSheet(Worksheets("_1010q my qsheet"))
```

---

**RunQSheets()**

Validates and runs all q-sheets in the active workbook.

**Syntax**

```vba
Public Function RunQSheets(Optional Quiet As Boolean, Optional ByRef Msg As String) As Integer
```

**Input**

**Quiet** *(optional)* If True, will not prompt user or display interactive error messages.

**Output**

**Msg** *(optional variable passed by reference for output only)* Is assigned status message.
### RunActiveQSheet()
Validates and runs the active query sheet only.

**Syntax**

```vbnet
Public Function RunActiveQSheet(Optional Quiet As Boolean) As Integer
```

**Input**

- **Quiet** *(optional)* If True, will not prompt user or display interactive error messages.

**Return value**

- 0 success
- 1 failure
- -1 fatal error (further query attempts are likely to fail)

---

### ClearCache()
Clears the 1010data cache.

**Syntax**

```vbnet
Public Function ClearCache(Optional Quiet As Boolean) As Integer
```

**Input**

- **Quiet** *(optional)* If True, will not prompt user or display interactive error messages.

**Return value**

- 0 success
- 1 failure

---

### IsSessionCurrent()
Indicates if the 1010data Excel Add-in has a current and valid 1010data session.

**Syntax**

```vbnet
Public Function IsSessionCurrent() As Boolean
```

**Return value**

- True Session is current and valid
- False Session is not current or valid
**GetSession()**

Returns session credentials for the active API session.

**Syntax**

```vba
Public Function GetSession() As Variant

Return value
```

- **array()**
  - If there is a current and valid session, the return value is an array containing the session handle, user ID, session ID, and session password, in elements 0, 1, 2, and 3 of the array, respectively.
  - NULL If no current session exists.

**Example**

Example usage:

```vba
Dim x As Variant
x = a1010data.GetSession
If IsArray(x) then
    ttsid = x(0)
    uid = x(1)
    sid = x(2)
    epswd = x(3)
End If
```

**ListDir()**

List the contents of a directory: gets "children" of given directory, formats results as a table, and pastes into the worksheet (with header row by default) at the specified location.

**Syntax**

```vba
Public Function ListDir(dir As String, destination As String, _
fields As String, _
Optional getType As String = "both", _
Optional quiet As Boolean = False, _
Optional noHeaders As Boolean = False, _
Optional descendants as Boolean = False) As Integer
```

**Input**

- **dir** Full path of directory for which contents will be listed.
- **destination** Range (named or address) of top left cell where results should be pasted.
- **fields** Comma-delimited list of table or directory attributes to retrieve (can be table or directory attributes; irrelevant attributes are ignored; a column will be created in the output table for each field specified). Available fields (attributes) for tables and directories can be found in the 1010data API documentation for the gettab and getdir transactions respectively. An additional field `entrytype` indicating `dir` or `tab` is also available.
- **getType** *(optional)* Can be `dir`, `tab`, or `both`. Defaults to `both`.
- **quiet** *(optional)* Set to `true` to suppress output to user in case of error. Defaults to `false`. 
noHeaders

(optional) Set to true to return output without a header row listing the fields retrieved. Defaults to false (headers appear by default).

descendants

(optional) Set to true to include subdirectories and their contents. Defaults to false (only the specified directory).

Return value

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>success</td>
</tr>
<tr>
<td>1</td>
<td>failure</td>
</tr>
<tr>
<td>-1</td>
<td>fatal error (further attempts are likely to fail, connectivity problem or the like)</td>
</tr>
</tbody>
</table>

Example

Example usage:

```vba
result = a1010data.ListDir("pub.demo.baseball", "sheet1!$A$1", "id,name,title,rows")
```

The above will generate the table below and paste it to sheet1!$A$1:

<table>
<thead>
<tr>
<th>id</th>
<th>name</th>
<th>title</th>
<th>rows</th>
</tr>
</thead>
<tbody>
<tr>
<td>37429</td>
<td>pub.demo.baseball.master</td>
<td>Players</td>
<td>15350</td>
</tr>
<tr>
<td>37433</td>
<td>pub.demo.baseball.batting</td>
<td>Batting</td>
<td>78881</td>
</tr>
<tr>
<td>37434</td>
<td>pub.demo.baseball.pitching</td>
<td>Pitching</td>
<td>32896</td>
</tr>
<tr>
<td>37435</td>
<td>pub.demo.baseball.fielding</td>
<td>Fielding</td>
<td>59884</td>
</tr>
<tr>
<td>37436</td>
<td>pub.demo.baseball.allstars</td>
<td>All-Star Teams</td>
<td>3707</td>
</tr>
<tr>
<td>37437</td>
<td>pub.demo.baseball.hof</td>
<td>Hall of Fame</td>
<td>239</td>
</tr>
<tr>
<td>37438</td>
<td>pub.demo.baseball.managers</td>
<td>Managers</td>
<td>2895</td>
</tr>
<tr>
<td>37439</td>
<td>pub.demo.baseball.teams</td>
<td>Teams</td>
<td>2327</td>
</tr>
<tr>
<td>37440</td>
<td>pub.demo.baseball.awards</td>
<td>Awards</td>
<td>1346</td>
</tr>
<tr>
<td>37441</td>
<td>pub.demo.baseball.postbatting</td>
<td>Post-Season Batting</td>
<td>7439</td>
</tr>
<tr>
<td>37442</td>
<td>pub.demo.baseball.postpitching</td>
<td>Post-Season Pitching</td>
<td>2776</td>
</tr>
</tbody>
</table>

LoadUSheet()

Loads the specified upload sheet.

Syntax

```vba
Public Function LoadUSheet(sht As Worksheet, Optional Quiet As Boolean = False) as Integer
```

Input

- sht
  - Must be a Worksheet object referencing the u-sheet to be uploaded.
- Quiet
  - (optional) If True, will not prompt user or display interactive error messages.
Return value

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>success</td>
</tr>
<tr>
<td>1</td>
<td>failure</td>
</tr>
</tbody>
</table>

**Example**

Example usage:
```
result = a1010data.LoadUSheet(Worksheets("_1010u my usheet"))
```

**LoadActiveUSheet()**

Loads the active upload sheet.

**Syntax**

```
Public Function LoadActiveUSheet() as Integer
```

**Return value**

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>success</td>
</tr>
<tr>
<td>1</td>
<td>failure</td>
</tr>
</tbody>
</table>

**Example**

Example usage:
```
result = a1010data.LoadActiveUSheet()
```

**DebugLog1010()**

Enables debug logging for the 1010data Excel Add-in with the option to add 1010data XML logging.

**Syntax**

```
Public Function DebugLog1010(FileNameInput As String, _
Overwrite As Boolean, _
EnableXML As Boolean, _
Optional Quiet As Boolean = False) As Long
```

**Input**

**FileNameInput**
An absolute path of the form `dir\filename.foo`, where `.foo` is a file name extension (e.g., `.xlsx` or `.txt`).

- If `filename.foo` contains either `_1010log` or `_1010xml`, the file name is left alone. Otherwise, the `.foo` extension is removed and replaced with `_1010log.txt`.
- If XML logging is enabled, a text file is created with the name `dir\filename_1010xml.txt`. If a file with `_1010xml.txt` is entered, but EnableXML is false, a new file with the name `dir\filename_1010log.txt` will be created.
Overwrite  If True, existing log files with the same name as `dir\filename_1010xml.txt` or `dir\filename_1010log.txt` will be overwritten. Otherwise, they will be appended.

EnableXML  If True, an XML log will be created.

Quiet  *(optional)* If True, will not prompt user or display interactive error messages.

**Return value**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>success</td>
</tr>
<tr>
<td>1</td>
<td>failure</td>
</tr>
</tbody>
</table>

**NewestVersion()**

Returns the versions of the installed and most recently deployed 1010data Excel Add-in.

**Syntax**

```vbnet
Public Function NewestVersion(ByRef InstalledVersion As Long) As Long
```

**Input**

`InstalledVersion` Will be populated with the installed version of the 1010data Excel Add-in in the form `YYYYMMDD`.

**Return value**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>error</td>
</tr>
<tr>
<td><code>YYYYMMDD</code></td>
<td>an 8-digit long integer which represents the latest 1010data Excel Add-in</td>
</tr>
</tbody>
</table>
Troubleshooting and Support

You can try to troubleshoot issues with the 1010data Excel Add-In. If you cannot resolve the issue, file a support request.

You may be able to resolve some issues by troubleshooting.

Upgrade to the latest version

Some issues are caused by using a down-level version of the Excel Add-in. Try upgrading to the latest version. For more information, see Determine the Latest Version on page 27.

Disable and re-enable all add-ins to fix ribbon display issue

If your 1010data Excel Add-in is correctly installed and not displaying in the Add-Ins tab ribbon, try disabling all of the Add-Ins in your ribbon, then re-enabling them. This should cause the 1010data Excel Add-in to display in the ribbon.

If you cannot resolve the issues, you can submit a support request. For more information, see Submit a Support Request on page 28.

Determine the Latest Version

The 1010data Excel Add-in provides a way to determine the version that you are running as well as the latest released version.

To determine both the installed and latest version of the 1010data Excel Add-In:

1. From the 1010data menu, click About...

If the installed version of the 1010data Excel Add-in matches the latest released version, you will see a dialog similar to the following:

![About the 1010data Excel Add-in](image)

**Note:** Note that the versions are the same.

If the installed version of the 1010data Excel Add-in does not match the latest released version, you will see a dialog similar to the following:

![Dialog with mismatched versions](image)
Note: If the versions do not match, you will be provided with a hyperlink to the 1010data Excel Add-in Download Page.

2. If you need to upgrade your version of the 1010data Excel Add-in to the latest version, click the 1010data Excel Add-in Download Page and follow the instructions.

Submit a Support Request

You must send specific information to the 1010data support team when asking for assistance in resolving a problem with the 1010data Excel Add-in.

If you are submitting queries, uploading data, or executing queries from VBA macros using the 1010data Excel Add-in and are experiencing errors or unexpected behavior, 1010data offers full support. You must enable debug logging and XML logging to generate the log files that the 1010data support team needs in order to assist you in resolving the problem.

To submit a support request:

1. Enable debug logging and run through the scenario in the 1010data Excel Add-in in which you’re encountering problems. See Enabling Debug Logging on page 28.

2. To submit a support request, send an email to support@1010data.com.

   Attach the following files to the email:
   - Debug log file
   - XML log file
   - Excel workbook you were using when you experienced the problem

The 1010data support team will contact you to help you resolve your problem.

Enabling Debug Logging

A debug log file and XML log file are required when submitting a 1010data support request to resolve any issues with the 1010data Excel Add-in.

Note: You should only enable debug logging when you are trying to resolve an issue with the 1010data Excel Add-in.

To enable debug and XML logging:

1. Click 1010data > Enable Debug Logging.
2. Click **Browse** to open the **File Save** dialog and navigate to the folder where the debug file will reside. Double-click an existing *debug_file* or enter the name of a *debug_file* in the **File name** text box. Disregard the **Save as type** drop-down list, as 1010data will append a specialized extension to the log file(s).

3. If you would like to generate an XML log file, select the **Enable XML logging?** check box. This log will be a text file containing the XML interactions between the 1010data Excel Add-in and the 1010data server.

4. If you would like to append to the existing log file(s), select the **Append existing log(s)?** check box. If this check box is not selected, the log file(s) will be overwritten.

5. Click **Enable Logging**.

Debug logging will commence, and the resultant files:

- *debug_file*.1010log
- *debug_file*.1010xml (if XML logging is selected)

will reside in the folder specified in the **Debug Logging** dialog.

**Note:** Debug (and XML) logging will remain enabled until Excel is closed.

**Debug Logging**

The **1010data Debug Logging** dialog box allows the user to enable debug and XML logging.

The user can specify the path to the debug log file, enable XML logging, and select whether to append or overwrite the existing log files.

Specify the folder and file name of the debug log file in the **Enter log file name** field (or click **Browse** to navigate to the desired folder).

**Note:** 1010data will append specialized extension(s) to the log file(s).

Select **Enable XML logging?** to generate an XML log file. This log will be a text file containing the XML interactions between the 1010data Excel Add-in and the 1010data server. The resultant XML log file will contain messages similar to the following:
Select **Append existing log(s)** to append to the debug (and XML) log files. If this check box is not selected, the log file(s) will be overwritten. In either case, if the log file(s) do not exist, they will be created.

Click **Enable Logging** to commence debug logging.