

Automating Excel Spreadsheets

Dbank can produce Microsoft Excel spreadsheets that read Dbank time series directly. The specific approach, implemented using Dbank's time series object model, allows Excel to access time series stored in a Dbank database without actually invoking Dbank's graphical user interface. "Automated" spreadsheets can hot-link any cell in an Excel spreadsheet to any time series observations and metadata stored in a Dbank database. The link is achieved using a Visual Basic for Applications (VBA) macro that calls Dbank's Time Series Engine from Excel. The manner in which this is done is natural to spreadsheet users because it does not interfere with Excel's user interface.

Automated spreadsheets provide the basis for sophisticated management reports that use all the power of Microsoft Excel for formatting time series information. Moreover, these spreadsheets will automatically re-read the data stored in the Dbank database(s) whenever the spreadsheet recalculates. Since each Dbank database can be up to two gigabytes in size, automated spreadsheets provide a means of sharing very large time series databases online with Excel users. Moreover, Excel's memory is kept to a minimum because only those observations needed to recalculate the spreadsheet are actually "pulled" into the Excel spreadsheet by the VBA macro.

This manual describes how to create an Excel spreadsheet that can read Dbank databases directly.

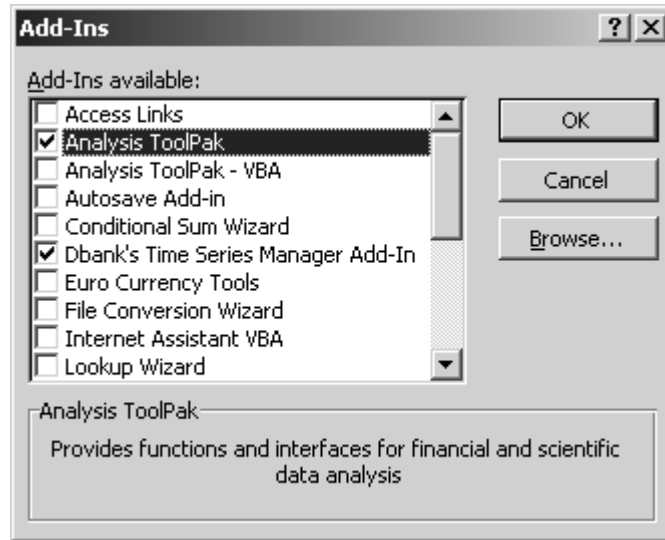
Automating an Existing Excel Spreadsheet

Before attempting the steps outlined below, please ensure that you have met all the requirements listed below:-

1. Installed the Dbank software.
2. Have access to any Excel spreadsheet you wish to automate.

Before you can start automating the spreadsheet, you need to carry out the following steps:

1. Open a new Excel spreadsheet, or open an existing spreadsheet that has a table to be automated.
2. Click on the Tools menu, select Add-Ins...
3. Select "Dbank's Time Series Manager Add-In" and then press OK, viz:



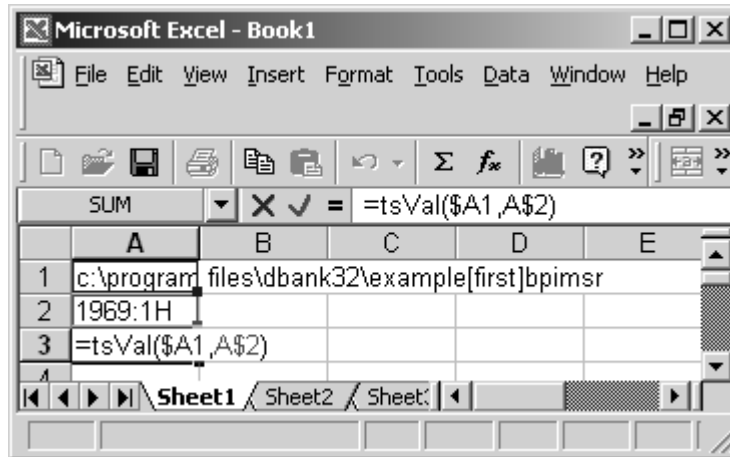
The add-in contains VBA functions that allow Excel to hot-link to a particular time series observation saved in a Dbank database. Note that the `tsVal()` macro in particular returns a blank string if (a) the time series does not exist; (b) there is no observation for the given date (i.e., it is missing).

4. Find and check the Read and Write Dbank Time Series under the “Available References” list. Then click OK.
5. Now you can close the Microsoft Visual Basic Editor and start working on the actual task of automating your spreadsheet.

Testing the Spreadsheet:

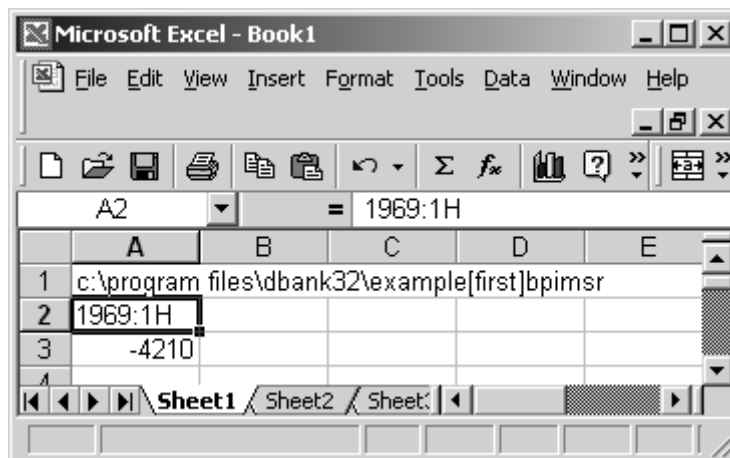
It is easy to determine whether your spreadsheet can link to a time series stored in a Dbank database. Try the following steps, which will read the variable `[first]bpimsr` from the EXAMPLE database stored in “ProgramFiles\Dbank32\” directory:

- Return to Microsoft Excel’s main worksheet by closing the VBA editor.
- Find an empty worksheet in your Excel file, or add an empty spreadsheet to the workbook using “Insert|Worksheet”.
- In cell A1, type in “c:\programfiles\dbank32\example[first]bpimsr”
- In cell A2, type in “1969:1H”
- In cell A3, type in “=tsVal(\$A1,A\$2)”, viz:

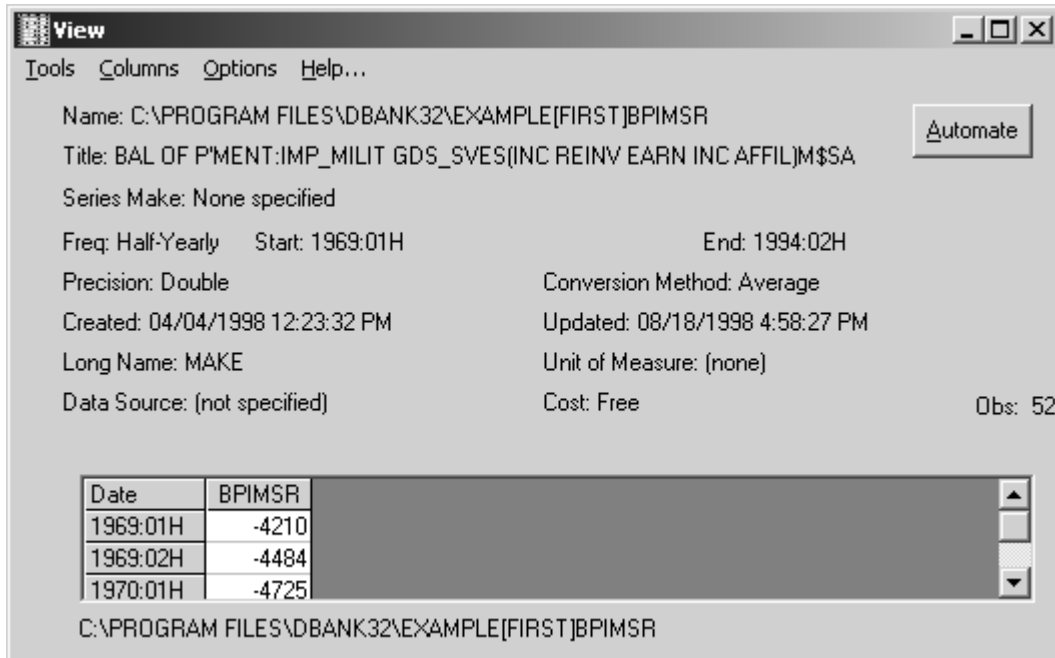


Note how the `tsVal()` macro is invoked. It is just like calling any other function available in Microsoft Excel. Since `tsVal()` is native to Excel, you can virtually forget about how it operates and continue to work with Microsoft Excel as you normally would.

Now press the Enter key to see the results of this function. You should end up with the following screen, which shows that the value of **[first]bpimsr** in the first half of 1969 was -4210.



To verify that the actual number is correct, we need to go to Dbank and view the time series using its native viewer. Here is the actual screen showing the values of **[first]bpimsr**:



Please note that you can now use all of Excel's formatting power, and cut-and-paste operations, to populate the spreadsheet.

Exporting an Automated Spreadsheet from Dbank:

The above steps help you train existing Excel spreadsheets to link to time series databases. Once you have imported the VBA macro, for each reference to `tsVal()`, you need to provide the full name of the time series and the dates for which you need data. You must supply these names and dates somewhere in the Excel spreadsheet (i.e., in a particular cell). You can (and should) draw on the power of Excel to construct these names and dates using standard cut-and-paste operations.

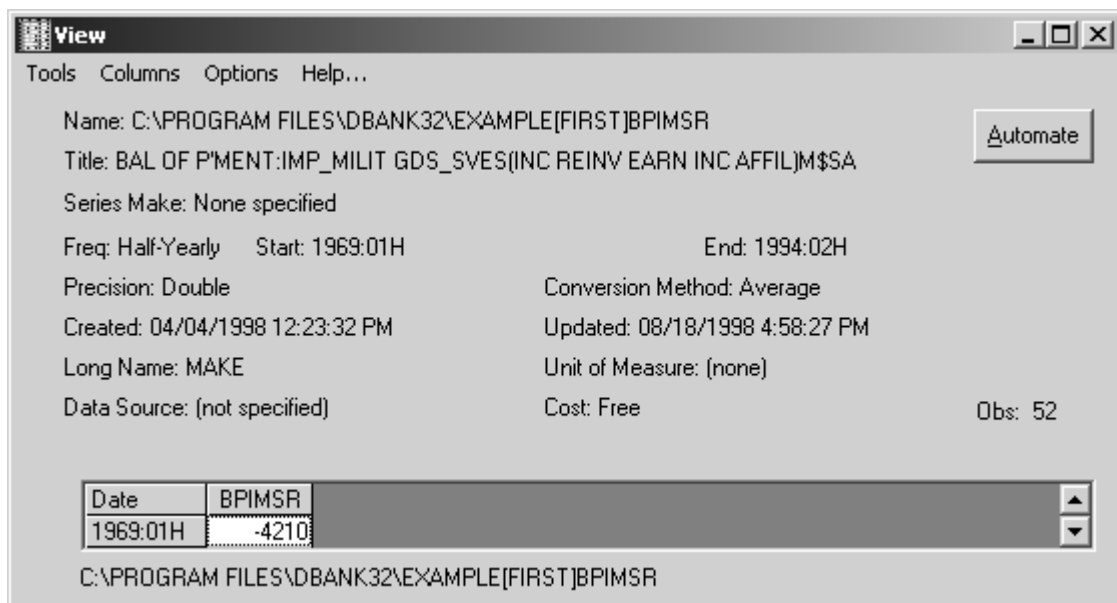
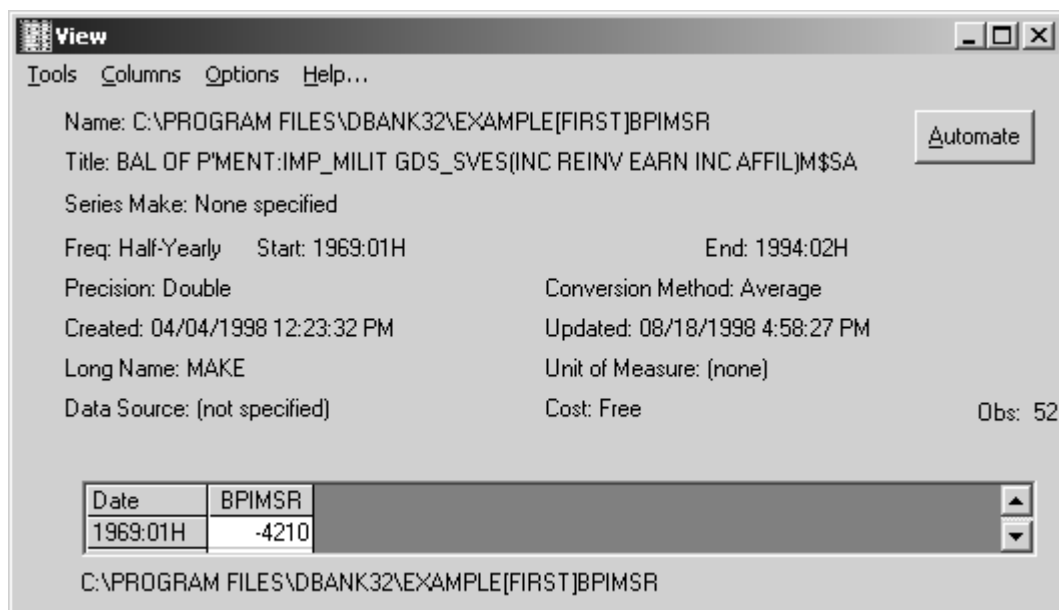
Nevertheless, constructing names and dates can easily become a tedious process. The following steps, however, show you how to create a new, completely automated spreadsheet using Dbank itself. The spreadsheet will have the "`tsVal()`" macro, as well as the dates you require. Using Dbank to create automated spreadsheets is far more efficient than manually typing in the series names and dates. It is also less error prone, since the computer is being used to generate the names and dates, as well as the appropriate cell references.

Let's assume that you want to create an automated spreadsheet containing for the

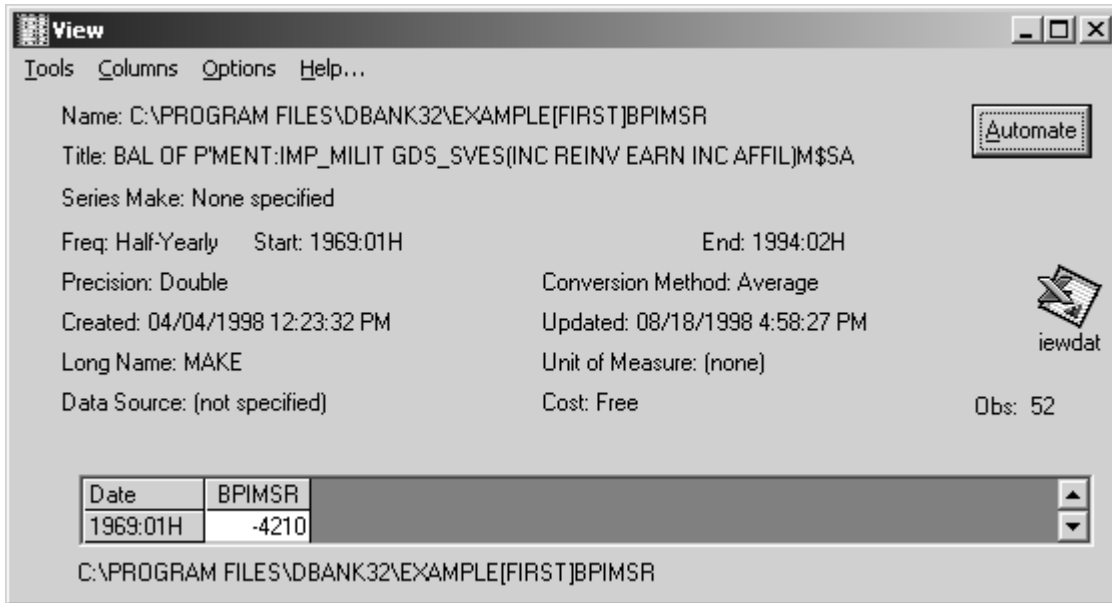
[first]bpimsr using

The following steps will create such a spreadsheet:

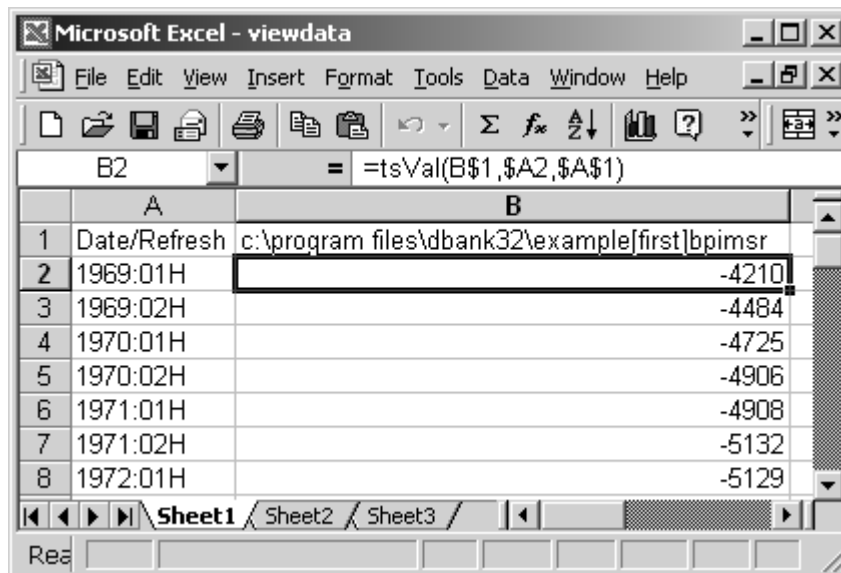
- Start-up Dbank by clicking on the Start Menu | Programs | Dbank for Windows, and open the EXAMPLE database “example.mdb” in: c:\programfiles\dbank32\;
- Click on the group called “first”, which will show all the series that belong to this group.
- Now select the series required using the mouse (hold the CTRL key down to select non-contiguous items), and invoke Series View, viz:
- Now invoke “Export to Automated Excel Spreadsheet...”



At the end of the process, you should end up with a form that looks like this:



Click on the Excel icon on the right-hand side of the form to retrieve the spreadsheet, which should look like the following:



Retrieving Other Time-Series Attributes

The following table documents all the function calls that can be made once you activate Dbank's Excel Add-In. These functions retrieve properties of the time series to particular Excel spreadsheet cells:

Function	Purpose
=tsVal(SeriesName,Date)	Returns an observation.
=tsDate(SeriesName,Offset)	Returns the date string of an observation
=tsTitle(SeriesName)	Returns the series' title.
=tsLongName(SeriesName)	Returns the series' long-name.
=tsUnits(SeriesName)	Returns the series' unit of measurement.
=tsFrequency(SeriesName)	Returns the series' frequency.
=tsGroup(SeriesName)	Returns the name of the group that the series belong to.
=tsName(SeriesName)	Returns the series' short-name.
=tsMax(SeriesName)	Returns the maximum value of the series.
=tsMin(SeriesName)	Returns the minimum value of the series.
=tsTotal(SeriesName)	Returns the sum of the series.
=tsMean(SeriesName)	Returns the mean (average) of the series.
=tsMedian(SeriesName)	Returns the median of the series.
=tsHarmonicMean(SeriesName)	Returns the harmonic mean of the series.
=tsGeometricMean(SeriesName)	Returns the geometric mean of the series.
=tsIsAConstant(SeriesName)	Returns TRUE (-1) if the series is a constant, 0 otherwise.
=tsVar(SeriesName)	Returns the variance of the series.
=tsStdDev(SeriesName)	Returns the standard deviation of the series.
=tsNobs(SeriesName)	Returns the number of observations in the series.
=tsValidCount(SeriesName)	Returns the number of non-missing observations in the series.
=tsMissingCount(SeriesName)	Returns the number of missing observations in the series.
=tsStartDate(SeriesName)	Returns the first date in the series.
=tsLastDate(SeriesName)	Returns the final date in the series.
=tsGroupName(ParentGroup,n)	Returns the name of the n^{th} sub-group in the parent group. Can be used together with tsGroupCount() to determine all the sub-groups.
=tsGroupCount(ParentGroup)	Returns the number of sub-groups contained in a group.
=tsGroupLabel(Group Name)	Returns the label of a group
=tsSeriesCount(Group Name)	Returns the number of series contained in a group.
=tsSeriesNames(Group Name,n)	Returns the name of the n^{th} series contained in the named group.