DbViewSharp User Guide

Contents

Introduction

What is DbViewSharp and why would you use it?

DbViewSharp is an open source desktop application for searching SQL Server databases. But then so is SQL Management Studio. So why would you use DbViewSharp instead of SQL Management Studio (SSMS)?

When you want to find out something about a database; the fields in a table, the data in a table, the source of a view or stored procedure, you want it fast. In SSMS you wait for the program to load then find the connection or even re-enter the connection details; then expand the list of database names (which is sometimes so slow); then find the database name in the list of databases; then expand the list of table names; then find the table name in the list of tables; then right-click, locate the item in the context menu to show the data; then choose whether to select 1000 rows to view or 200 rows to edit. Then wait while the SSMS fetches the data. It's painful.

To obtain the same information in DbViewsharp start it up. Click to connect to the database and right there are the tables. Find the table, double-click on it and right there is the data.

You want to see the source for a view? With SSMS you expand the list of views, you find the view you want, right-click to get the context menu, search down the list of options, figure out you want "Script view as...", then Create to..., then New query window, wait for a moment, and finally see your view source. In DbViewSharp, click on the views toolbar button and you see the views. Double-click on the view and you see the source. That's all.

SSMS is a mighty application, which does (almost) everything that DbViewSharp does and much, much more. However it was designed so you could manage every single aspect of a SQL Server database through a GUI and compromises its search functionality with its all-inclusivity. DbViewSharp is tuned for searching, which is what I've wanted most for the last 10 years of working with databases. I have always been impressed with how easy it is to find what you want using Google and sensible keywords. I wanted to translate that experience to a database explorer application so you bring to the screen the items you are searching for with the absolute minimum of keystrokes or mouse clicks. The result is DbViewSharp.

Summary of features

- View tables and row counts.
- View column information.
- View stored procedures, views, functions, triggers, computed columns, service broker objects and other interesting things.
- Check the physical size of tables and database files.
- Search view and stored procedure text.
- Create customised grids of table data for easy searching and editing.
- Edit table data.
- Perform aggregation tasks (sum rows, breakdown data in a column).
- Execute ad-hoc Sql queries and commands.
- Script table schema and table data. Create C# classes for LINQ2SQL and Entity Framework.
- Import and Export data via CSV files.
- Report differences between two databases.

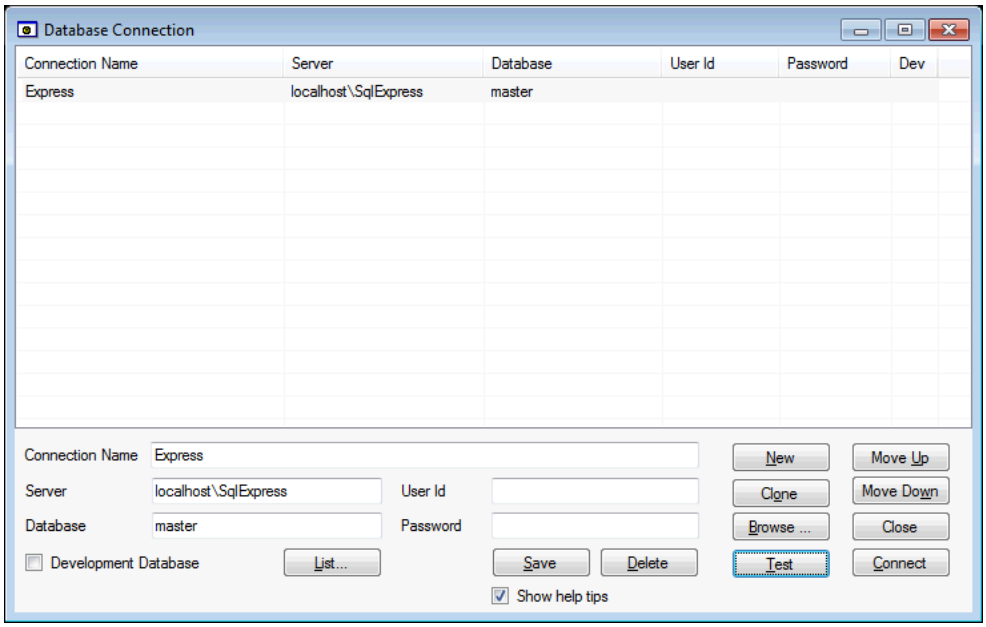
Recommended, Dubious and Dangerous Features

In the pages that follow features flagged ** Recommended ** are those that have been extremely to the author over the years he has used DbViewSharp. Conversely those flagged *? Dubious Merit ?* are rarely used and possibly underdeveloped. Finally, those with a “!” will change the database in some way and are only available when the connection is flagged as a development database.

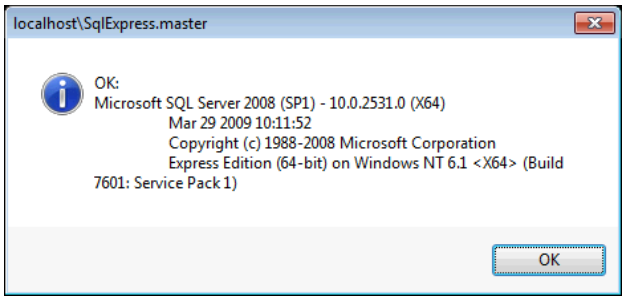
Getting connected

When you launch DbViewSharp the first screen you encounter is the connection screen. If you have previously entered connections you can select the one from a list. Otherwise if this is your first use you will have to enter the details of the server and database you want to connect to.

The following walkthrough assumes you have SQL Express installed on your local machine



1. [New]
2. Enter *Express* in the Connection Name box
3. Enter *localhost\sqlexpress* in the Server box
4. Enter *master* in the Database box.
5. [Test]



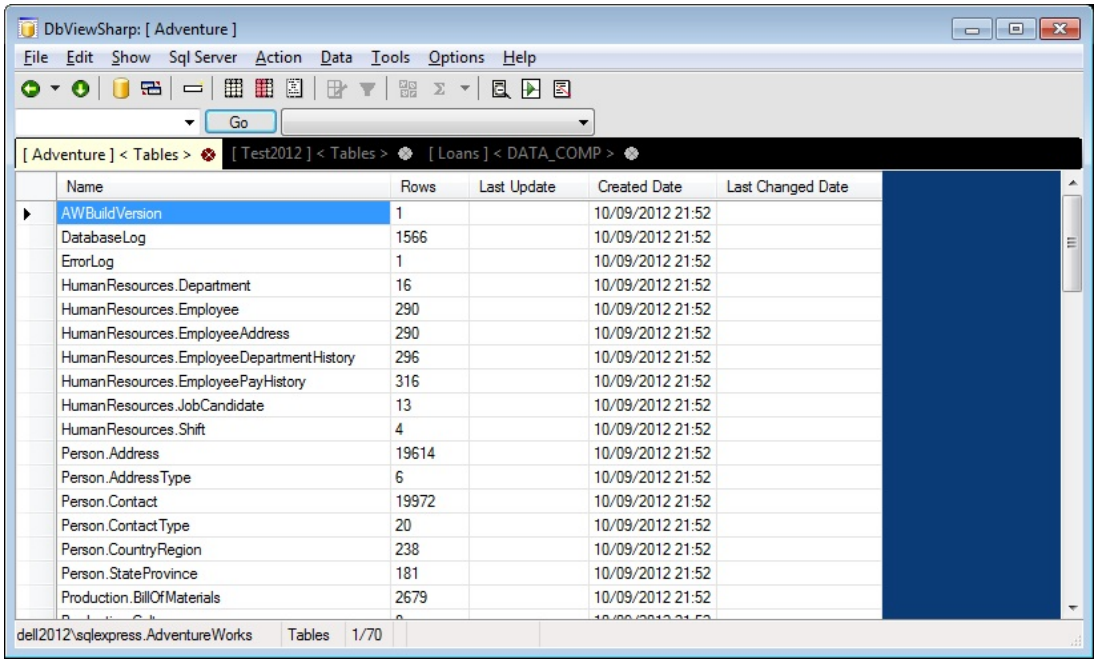
6. [Save]
7. [Connect]

DbViewSharp saves the connection details of all the databases you connect to. As you may build up a large set of these it helps you to assign a name tag to each one. This will attach additional information to the server and database name and act as a reminder of the purpose of the database. If you do build up a large collection of connections then devising a systematic naming convention will make it easier to find the right connection in the list.

Application Main Window Layout.

Overview and example of use

Once you have connected to a database you will gain access to the main window. This contains the standard elements of a desktop application: main menu, toolbar and status bar. Directly underneath the toolbar is the search bar and below that, normally occupying the bulk of the window, is the result grid. When tab mode is enabled and you have connected to more than one database then a tab bar is displayed between the search bar and the result grid.



The application is designed to display in the result grid the data you are seeking. To achieve this goal normally involves selecting an option to display an initial grid of objects then using the search bar to filter the rows displayed to include the items you are looking for. A common additional step is to “drill down” from a single row in the grid to another grid based on the object in the selected row using either double-click or right-click to select from a context pop-up menu. Below is a very common sequence used to examine the data in a single row of a table.

1. Select Show, tables from the menu. The grid displays tables.
2. Enter “Ord” in the search text bar. The grid displays just tables with “Ord” in their name.
3. Double-click the row for the table “Orders”. The grid displays row data for table Orders.
4. Enter 101 in the search text bar. A single row for order #101 is shown on the grid.
5. If the order data extends beyond the right edge of the window double-click on the row. The grid now displays row per column. Each row has two columns; one for name the other for data.
6. If there is a column containing a long order description, which is not completely displayed, right-click on that cell. The text from the cell is displayed in a new window.

Result Grid

The result grid shows the data from the database in response to menu selections you make. It will show a list of the user tables in the database when you first connect or re-connect. The grid will not normally be editable, with the exception of the table contents grid. Columns are not resizable as the grid sizes columns automatically to display all data and column headings. The one exception to this rule is when displaying the data content of tables with large numbers of columns and more than 500 rows.

Most result grids will have a right-click context menu from which you can access functionality not available from the main menu. You can access it from the grid's grey left margin. Clicking next to a row offers all of the right-click functions, most of which act on the object of the selected row. Some actions however apply to the grid as a whole. These are displayed if you click in the top left grey rectangle to the left of the first column. Every grid will include the "Show Sql" option there. This option reveals the Sql used to generate the grid. It can be handy to examine in cases where the grid is empty and you are not sure why.

Clicking on the column title sorts the grid by that column as normal and a second click changes the sort order to descending. However for data tables where there are many more than 500 rows in the table this will only show you the first 500 rows in descending order. If you click [GO] the program will apply the descending sort to the entire table and present a new grid with the last 500 rows from the entire table (note: unless you have already explicitly selected a sort order for the data).

A small quirk in the behaviour of the result grid is demonstrated when you right-click on a cell in the grid rather than the margin. This does not cause the context menu to appear. Instead the cell flashes red for a moment. This is visual feedback to confirm that the cell contents have been copied to the clipboard. In a further refinement if you right-click on a text grid cell that contains more than 128 characters then as well as copying the text to the clipboard it will be shown in a separate window. In yet another refinement if the text is Xml it will be formatted with lines and indentations before being displayed. If you right-click the column title it will copy the title into the clipboard.

Be sure to refer to the section on the table data contents grid for a discussion of all of the special features implemented for it.

Search Bar

Recommended feature The search bar provides a quick way of filtering rows from the result grid with the aim of allowing you to bring the rows you really want to see into view. However the intention is not necessarily that you enter text to make an exact match and get just the records you want in result grid. It is designed to behave more like a Google search where you enter a few keywords and find what you want on the first page. Exact match searches are boring because:

- You have to enter more text than you want to and may make mistakes.
- You may not remember the exact text to search on, then you're stuck.
- If you just look for what you know is there you'll never find other interesting stuff.

It is worth understanding how the search bar works. If you do then it should cover at least 80% of your searching needs; if you don't it may become frustrating and it's too good a feature to discard.

Search bar FAQ

Which column is it searching on?

This depends on the grid. In most cases it searches on the object name, so for the table's grid it searches the table names, for the views grid the view names etc. Where there are two columns of names (eg dependencies) it should search both and return any row where either matches. The exception is the table data contents grid. The default is to search the first three columns as long as they are character or integer types. So if the first three columns are all integers entering text, with the intention of searching the fourth column if it is a character column, will not work as expected.

What is the exact type of search?

For all grids apart from the table data contents the columns being searched on will match if they contain the search text in any position. If you are unfortunate enough to be working with a case sensitive database the search is adjusted to ignore case. A final special case is when object names use a C like naming convention and include underscores. The search will ignore that and behave as if the object names are CamelCase. To illustrate the text "ord" matches the following object names:

- **ORD**ERS (even on a case sensitive database)
- **Ord**ers
- C**ord**Widths
- RUM**OR_D**ATA

These rules also apply to searching character columns in the table contents grid.

So what is the difference between searching the table contents grid and other grids?

It is when you enter an integer in the search bar. In most grids this integer is treated just like ordinary text and searched as such. So "1" entered will match "Table1", "View13","sp_update101" etc. However experience has shown that it is overwhelmingly more useful when searching the data in tables to assume that a number entered in the search bar is intended for searching on the search columns with integer type only. This behaviour allows you to target rows by primary keys and is massively convenient. Now instead of writing: select * from Orders where OrderNumber = 101 just enter 101 in the search bar and press [GO]. This is a huge time saver when tracing data from one table to another.

But if I really want to search for a number in a text column am I thwarted?

Not really. Enter "%1%" instead. This is detected as not being a number and so is applied to searching the text fields.

So what happens if I enter "Ord%"? Does it...?

Just select Orders and not CordWidths? Yes. Normally the search text is enclosed with a "%" at the start and end to make the desired search work. However if you actually include a "%" in the search text this is not done enabling you, if you know the trick, to refine the search accordingly.

How about date fields? I can't seem to make them work at all

Filtering on date fields does not work. There are better alternatives for filtering on date and floating point numbers. See the description of the menu option Data, Filter.

Why does the search text sometimes stay put an sometimes disappear?

There is subtle logic used to determine whether to use the filter text or clear it. If you have just applied the filter search to one set of objects and are switching to another then you probably do not want the filter to be applied to the new object list. If you have just entered search text and are about to press the icon to show views then you do want to apply the filter.

But I don't think this is the right way for it to behave.

There is a common scenario where the search bar clearing logic appears to be wrong. It is the case where you enter filter text and search only to realise that you just searched the wrong set of objects. For example in your mind you are searching the views grid, but the grid was actually displaying tables. When you switch to views your search disappears and the full list appears. It is when this happens too often that you begin to doubt that the search bar is working correctly.

Fortunately there is an easy way to recover the search text just cleared. Instead of pressing the dropdown button on the search bar instead press the Clear search filter icon button. Despite its name if the search box is empty this button will apply the previous search.

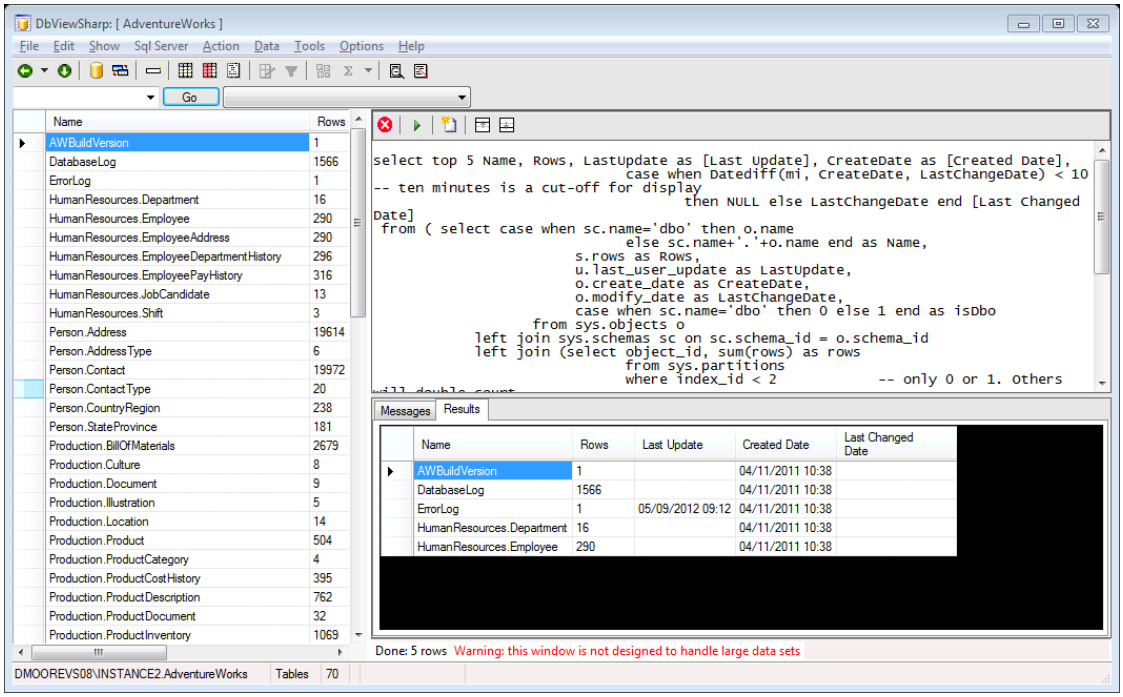
View Selector

The view selector is the drop-down list in the search bar located to the right of the [Go] button. It will be empty until you begin to save DataViews. *DataView* is the term used by DbViewSharp to refer to the combination of settings that define the columns and row filters that can be set to define the grid display for each table. Normally the DataView is constructed automatically, but many aspects of the automatic configuration can be modified. The topic is covered in detail in the Dialogs section.

The view selector displays the views saved for the current connection when you open the drop-down list and will display the table data using the saved settings when you actually select one.

Ad-Hoc Query Window

** Recommended ** The Ad-hoc query window appears beside the grid display when you select the menu option Tools, Adhoc Query Window. It is laid out in a standard manner with an upper pane for entering the SQL and a lower pane for displaying results. Use it like the SSMS version; enter Sql, mark the text to run and run it. If no text is marked then it's all run. If the Sql generates a grid the grid is displayed in the result tab of the lower pane. If it is a command that changes data rather than fetching it then the messages from that operation are displayed on the message tab of the lower pane.



At the top of the entry pane there are a few icons that are included to make the use more straightforward. They are in turn:

White cross on red circle: close or hide the window. The text entered is not cleared and will re-appear when the window is next opened. However if you change connection the Sql is commented out as a safety measure against running a destructive command on the wrong database.

Green Arrow: Execute the selected Sql.

White page with yellow flash: Clear the edit window.

White rectangle enclosing grey up arrow: Make the result pane big at the expense of the edit pane.

White rectangle with grey down arrow in: Make the edit pane big at the expense of the result pane.

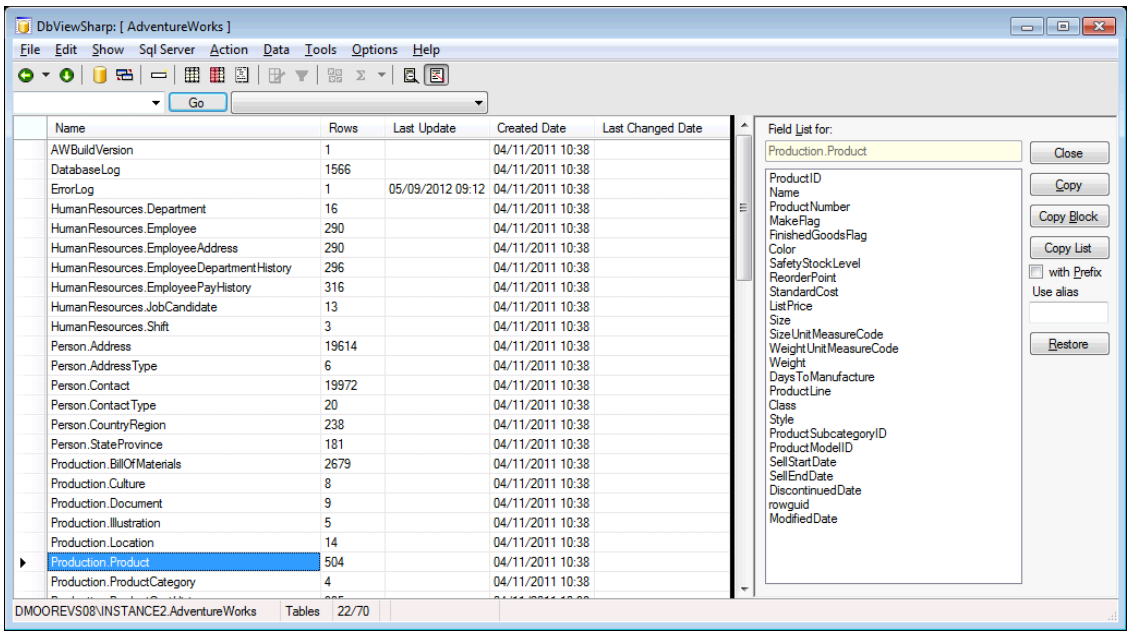
If you are used to SSMS the query window then you must be aware of a few limitations of the Adhoc query window. Firstly it does not support multiple data sets. If you execute commands that would generate several record sets then only the first one is displayed. Secondly it does not recognise the "GO" command for separating multiple sets of commands. Finally it will not handle very large data sets very intelligently or efficiently; there is a warning displayed to that effect on the status bar.

Be careful to remember the connection associated with the window. Each tab will maintain its own query window so that the query will always execute on the database shown in the status panel. However if you use the back button the connection associated to the tab will change. In case the SQL in the text pane would be dangerous to execute in the database just switched to it is commented out when you use the back button.

Field Selector Panel

The field selector panel is displayed to the right of the tables grid when you select Tools, Field selector from the menu or press the rightmost toolbar icon. It will disappear when you change the grid display or switch tabs, but reappear when you return to the tables grid.

There are two uses for this panel. Firstly it shows the fields of the selected table, updating the display as you change the selection. When investigating a database for the first time you can use this along with the row count to get a sense of the important tables (ie those with many columns and many rows) and possible relationships between them. Secondly it provides a convenient means for selecting columns for building Sql in SSMS or any similar external program. It is this latter requirement that is the reason for the controls. The panel is shown below with an explanation of the controls.



[Close]

Hide the field selector panel.

[Copy]

? Dubious Merit ? Copies the field list as in block mode if no fields are selected. Otherwise it copies the selected fields in block mode. This was the original control before the two following controls were added. It is practically a duplicate of [Copy Block], with the sole exception that it will copy a single selected field, whereas the other will copy the whole list if only a single field is selected.

[Copy Block]

Copies fields to the clipboard in block mode. Block mode means that line breaks are inserted into the list at about the 80-characted line length mark. This produces more manageable Sql for large field lists than the next option that adds a line break after every field. If no field or one field is selected then the whole list is copied. Otherwise only the selected fields are copied.

[Copy List]

Copies fields to the clipboard in one-per-line mode. This is useful for developing very clear Sql, particularly when you want to rename the fields for the result table. If no field or one field is selected then the whole list is copied. Otherwise only the selected fields are copied.

[x] with Prefix
Use Alias [ ]

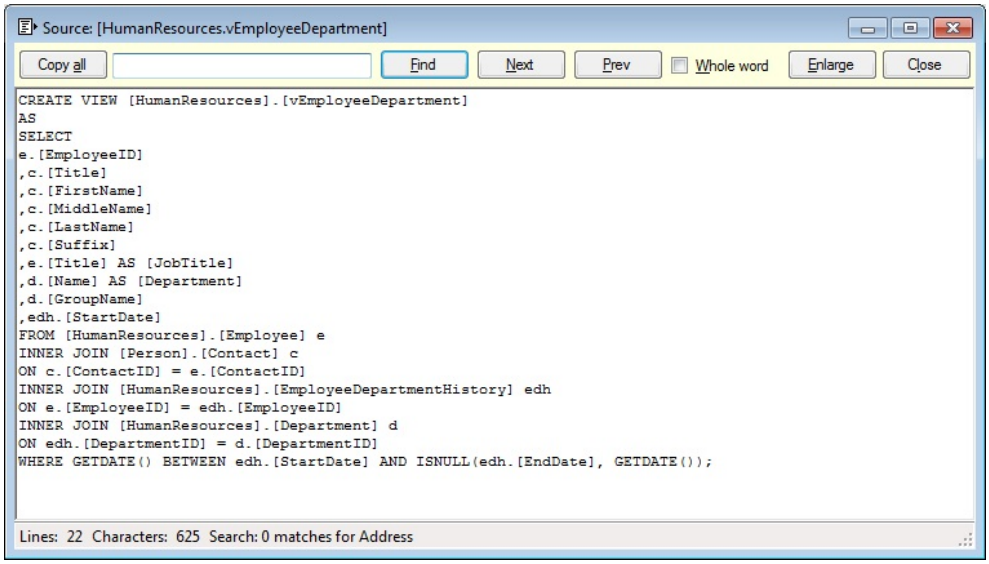
When you are creating a complex query with joins on many tables it is probably useful to prefix your fields with the table name or an alias. When you check this box the field list will be prefixed; either with the table name or the alias entered in the text control beneath it.

[Restore]

If your table contains a large number of columns it may be useful to filter the list to find the field you want. When field list control has the focus if you type in letters the list will change to display only matching fields. A cut-down list changes colour to red. Backspace undoes the last letter entered and the escape key restores the full list. As does this button.

Source Form

The source form is used by DbViewSharp to display view and stored procedure source code and any other multi-line text. It displays generated scripts of Sql, the content of large text columns and Xml. It works in “tear off” mode, which means that the window pops up on top of the main screen, but need not be closed before you can continue with the application. It’s possible to have many source forms open at once so you can compare the contents. The source form possesses one disconcerting feature: it can pop open in two different sizes. This feature developed following frustration at always having to enlarge the window to get a decent view of a large stored procedure. Now if the text to display has more lines than a given threshold the window opens at pretty near full screen size. Otherwise it opens at a size just sufficient to display the text.



The main body displays the source under examination. This is framed at the top by a control bar and at the bottom by a status bar showing a count of lines, characters and search text matches. The top bar contains the following controls.

[Copy All]

Copies the text displayed and closes the window.

Search Text window [ ]

Contains search text either entered while the form is open or pre-filled by previous operation of the program. An example of the latter situation is if you have selected to search stored procedure source for something and double-click on one of the names in the result grid. The source form will highlight all instances of the search text in the source and if necessary scroll the display to the first instance.

[Find]

If you enter new search text press this to make the search. It will highlight all of the matches and scroll the display to the first one.

[Next] [Prev]

Navigate the matches scrolling the display of text where necessary.

[x] Whole word

When checked this highlights only whole words in the display. “Whole words” are really table and column names. The matching routine identifies characters that are not normally part of a Sql object name and matches the target to the text between these boundary characters. Useful when searching for a field named “Id” in a stored procedure containing other fields named “CustomerId”, “OrderId” etc. This option is automatically set on when you have selected to search for an object via the right-click context menu.

[Enlarge] [Restore]

Two labels on the same button to toggle the size of the source display ? Dubious merit ? since the implementation of intelligent sizing on opening.

[Close]

Closes the window.

Toolbar icons.

Toolbar icons provide quick access to the most commonly used functions or ones that are not easy to place in the menu system. While it is not easy to guess the function from the icon image there is tool-tip help available for each. There is no option to hide the toolbar yet. The following paragraphs describe the image and action of the icons in the order they appear on the toolbar at the time of writing.

White left-arrow on green: Navigate Back

Displays the previous grid. This is intended to behave exactly like the back button in a browser did before Web 2.0. The down-arrow to the right will display drop down of recent grids. When two or more tabs are showing the back button navigates to just the grids that were displayed in the current tab.

White down-arrow on green: Show history

Displays a grid showing details of the grids previously shown. Double click on any row to re-display it.

Gold cylinder: Connect

Same as main menu *File, Connect*. Displays the connection menu.

Two rectangles: New tab or Launch new instance.

It is sometimes more convenient when tracing the path of data through a database to have two tabs to work with instead of using the back button or history to switch grids. Pressing this icon may perform one of two actions depending on the tab mode. If tabbed windows are on (the default) then pressing this button will create a new tab for the current connection as if you had pressed the connect icon and re-selected the current connection. If tabbed window mode is off then instead it launches a new instance of DbViewSharp connected to the current database. In either case you will end up with a second grid view.

Slim white rectangle: Clear or re-instate the search filter text.

This icon functions as it does because the search filter text box has a subtle behaviour (described previously). The action it performs depends on whether the search text box contains text or is empty. If it contains text then this is cleared and the current grid is refreshed. If the search text is empty then if text had been entered and searched on previously then this is redisplayed and the grid refreshed. Note: unlike the grid history which is distinct for each tab, the search text history is shared between tabs.

Black and white grid: Show tables.

This is the same as the menu option Show, Tables.

Grid with red columns: Show Views

This is the same as the menu option Show, Views.

White page with writing: Show Procedures

This is the same as the menu option Show, Procedures.

B&W window with yellow pencil. Modify data-view.

Same as menu option Data, Modify DataView. This icon is only enabled when the grid is displaying table contents. On double-clicking a table DbViewSharp will generate a default DataView that includes all columns, no filter and no sort order (although normally the sort order is by primary key). For tables with small numbers of rows and columns this may be suitable, but for larger tables it is often useful to limit the data fetched and displayed. Clicking this icon displays a dialog that permits

you to select the columns to display and sort by. All of the settings are described in the menu item description later.

Black funnel: Filter rows.

Same as menu option Data, Filter. This icon is only enabled when the grid is displaying table contents. Displays a dialog for adding a filter condition to the grid display. Select a column, a comparison operator (=, < etc.) and a value. Designed for the scenario where the table has many rows of data, you spot an interesting value in a column and wish to find the other rows of data with the same value. The fastest way to do this is to select the grid cell containing the target value and press this funnel icon. The dialog will display with the target column and value pre-set and the comparison operator set to =. Press the button [Apply] and the filter is applied. The complete filter dialog is described later.

Four coloured boxes: Aggregate.

* *Recommended feature* .* This icon is only enabled when the grid is displaying table contents. It will count the instances of values for data in the column of the current selected cell in the grid and present them ordered by most numerous to least numerous. This is a great way to discover the range of values a column may contain, the number of rows with null instead of data and data problems such as bad case (ABc instead of ABC) or typing errors (ABV instead of ABC).

Sigma: Count rows.

Press this icon to obtain a count of the rows containing non-null data in the column of the current selected grid cell. The down arrow to the right pops up further options; min/max and sum/average.

Magnifying glass over page. Search source code.

Same as menu option Action, Search source. This displays a dialog in which you can enter search text then select which type of objects to search including stored procedures or views. Any objects containing the text are displayed in the result grid.

Page with red pointer: Field selector.

Same as Tools, Field selector. Available only when the table list is displayed. Depress the icon to reveal a dialog listing the columns of the table. The dialog is intended to help when writing Sql by providing functionality for copying column names into the clipboard. See later for full details.

Main Menu Option Reference

The follow sections describe the main menu options as they appear along the top of the screen. Where the option results in a new grid then the contents are analysed briefly and the right-click context menu options are described. Where the option causes a dialog to display the controls are described in detail as is the result of pressing [OK] or equivalent. Features that will reward consideration are indicated with **Recommended **. Features tagged *? Dubious merit ?* have shown limited use to date in the working environment.

File

File, Connect (Toolbar: Gold Cylinder)

Displays the connection dialog , which serves three purposes:

1. Allows you to connect to another database
2. Enables you to manage the connection list; adding changing, moving or removing them.
3. Search and reference the connections and copy details to the clipboard by various methods.

The connection screen, pictured in the section *Getting Connected*, is one of the most important screens to become familiar with since you cannot use the application if you cannot connect to the database. It consists of a list of previously saved connections at the top, with a collection of text entry boxes and buttons below. See the dialog description for a full explanation of all the controls.

File, Change Database Manager

While DbViewSharp is primarily designed to connect to Sql Server and Sql Express database the code is structured to enable connection to other database managers. For example Sql Server Compact edition is sufficiently different from the other Sql Server variants to warrant a separate driver assembly. In the future there may be even more exotic drivers available: SqlLite, PostgreSQL or Oracle. This menu option displays a simple dialog that shows the installed drivers which enables you to change to a different one. When you change to a different driver all grids for the previous driver are closed and the connection dialog is displayed. On exit the application it records the driver of the last connection made. When you next open DbViewSharp it assumes you want to connect to a database managed by that driver.

File, Launch New Instance (Toolbar: Two rectangles)

Select this option to create a new grid from the current one. The form this takes depends on whether the application is in tabbed window mode or not. The default state is tabbed windows, but an Options menu item can change that. If tabbed windows are on then it will create a new tab for the current connection. Otherwise if tabbed window mode is off then it launches a new instance of DbViewSharp connected to the current database. In either case you will end up with a second grid view that initially displays the tables of the current connection.

File, Recent connections.

The File menu will hold up to 8 recent connections and preserve the list after you save. When you select a recent connection DbViewSharp will connect to the relevant database and display the tables. Changing connection like this will not create a tabbed window, but the history list is maintained. Should you delete a connection that still appears in this recent connections list you will cause a non-fatal application error if you attempt to connect with it.

File, Exit

Quit the application. Note: if you have been editing table data any outstanding unsaved edits will **NOT** be saved. The last connection is remembered for the next time you open the application when the connection dialog box will be set-up for you to reconnect.

Edit

Edit, Edit Mode (Ctrl +F2)

This option is only available when the result grid displays table data contents. Normally DbViewSharp should be considered as a read-only tool so the approach it takes to editing is quite conservative. All table data is displayed in read-only mode by default so you explicitly need to set edit mode to on with this option or accelerator key combination (Note: in practice it is more convenient to press Alt+E,E than Ctrl+F2). When the table is editable the selected cell background colour will change to provide a visual cue. Type replacement data directly into the cell to replace the contents entirely; alternatively press F2 to edit the contents.

You may change data in more than one cell and more than one row, but it will not get applied to the database until you perform an action to commit it. Actions include: pressing [Go], selecting another type of grid, selecting another connection; in fact practically anything. You will be prompted to save or discard edits, except when you exit the application as mentioned before. If you have pressed [Go] then the data grid is refreshed from the database with the edits made and edit mode switched off.

Table editing is not recommended for patching production data except under the direst circumstances. Please try to use a pre-tested script.

Note: one effect of setting development mode to on for a connection is to enable editing of table data by default.

Edit, Select All Rows (Ctrl+A)

Following a recent upgrade this menu option is slightly misnamed. If the focus is in the Ad-hoc query window (see elsewhere) then it selects all of the text in that window. Otherwise it selects all of the rows in the grid of the active tab. The same effect can be achieved by clicking the top left grey cell in the grid. Following this action you can copy all the grid contents to the clipboard by right-clicking in any data-cell on the grid.

Edit, Expand Rows

? Dubious merit ? This is such an esoteric option that it is best justified by giving a set of instructions to follow:

1. Connect to AdventureWorks database
2. Locate the table Production.ProductPhoto
3. Double-click to display data
4. Widen a row to display the whole photo.
5. Select that row and then select Edit, Expand rows.

Show

Show, Tables (Toolbar, Ctrl+T)

The grid showing the tables in database is the grid that is automatically shown when you connect to a database. The reason for this is because the primary purpose of a database is to store data and the most common reason for connecting to a database is to examine data. Since it is not possible to guess which particular table the user wishes to examine the best solution is to list the tables and provide convenient ways to navigate to the correct table and then “open” it to display the data. Indeed the tables grid is analogous to the normal view of files provided by windows explorer. DbViewSharp is unusual in that it displays row count and last-updated date information next to the table name. The row count is a **Recommended** feature to be aware of.

Since there is a lot to do with tables this grid has an extensive right-click context menu.

Data (default)

Recommended +. The ability to display data from tables is the most extensively used feature in DbViewSharp. The design has been honed over many years in order to achieve the ultimate goal of bringing the data being sought, ie the subset of columns and rows from the table onto the screen and to minimise scrolling and visual scanning. To this end a table data contents grid (hereafter "data grid") possesses a number of unique properties.

1. A row limit of 500 is set on grid. Since tables may contain millions of rows the application could either attempt to fetch and display all of the rows in the table. This is suicidal and would crash the system with many tables. Alternatively it could implement paging. However how many times do you find anything useful on page 2 of Google? The row limit is set with the intention of the application responding quickly and presenting sufficient rows for the user to determine his or her strategy for adjusting the grid parameters to target his/her information.
2. ** Recommended ** One tool for targeting the information is enhanced search bar functionality. The data grid is the only one that detects a search on a number and adjusts the column(s) used in the search accordingly. This is covered in more in the general introduction to DbViewSharp.
3. ** Recommended ** Another important tool is the ability to create a semi-persistent view on the table. It is not to be confused with the true database view, so it is termed a "DataView" in the application and the documentation. When you select to view the data of a table for the first time in the current DbViewSharp session it creates a default DataView that includes all of the columns, no sorting and no filtering. From examination of the result grid created by this initial DataView it is a fairly swift process to decide which columns are important to report, sort on, filter by and flag to include in search bar filtering.
4. Additional filter. Of slightly less importance than the previous features is the ability to set an ad-hoc filter from a value in a grid cell. The original design of the feature was to find either null data in a mainly populated column or non-null data in a mainly empty column. Over time the scope was increased to allow you to construct a quick filter search from any value. Scenario: the data grid for an unfamiliar table shows an interesting or unusual value in one of the rows. You are interested to discover how many other rows contain this value. Select the grid cell and press the funnel icon. The filter dialog is displayed with the value set ready for searching. Press [Apply] to make the search.
5. ** Recommended ** A similar, but more successful, operation that can be applied to the data grid is the aggregation operation. For a chosen column this reports the count of each different data value in descending order of frequency.
6. Coloured columns. Further information is added to the data grid by colouring important columns: primary keys, foreign keys and indexed columns.
7. Data Editing. Data grids are the only grids that can be edited.

The table data grid itself has an extensive right-click context menu described later.

Fields

Recommended Displays the columns for the selected table. The grid has been carefully designed to present as much useful information as possible that will fit the screen. In developer mode the right-click context menu provides access to schema-modifying functions; add, remove or change a field, and add a foreign key.

! Modify, Drop
! Modify, Truncate
! Modify, Reset auto-increment

In development mode only the options above perform destructive changes to the table. Drop will delete the table, Truncate remove all of the data from it and reset auto-increment will set the next value for the auto-incrementing field to be 1 greater than the previous last value.

Dependencies
? Dubious merit ? Displays the same grid as that from the *Show, Dependencies* menu option below, but exclusively for objects that depend on the selected table. The intention is to help determine the extent of use of the table in the database, but the flaky nature of Sql Server dependency information, the main menu report of all dependencies and the various source searching options tend to make this a little-used feature.

Find in source
**Recommended*.* Searches source code for the selected table name. The option to search to source code of views, procedures and functions is a response to the unreliable results obtained by using the dependency information held in Sql Server. In line with other areas of the application the results of the search for a table name in view or stored procedure code are not absolutely accurate, but only in as far as the search can return false positive results caused by simply matching text instead of understanding and parsing Sql. Again experience has demonstrated that this is not normally a big problem. What is a big problem is when the account connected to the database does not have access to the source code. In this instance the option is useless. See the discussion of permissions elsewhere in the guide. The dialog is described in more detail later under the main menu option Action, Search Source.

Descriptions
This is an interesting option in that it may have great use or no use at all depending on whether or not a database makes use of extended properties to add customised metadata to the database objects. The Descriptions option will display a result grid showing the *MS_Description* extended property for each field in the table, with its own context menu option to add or edit this property value. So it may be utterly useless if the property is never populated. Alternatively It could be very useful if the property is used as this information is not prominent in SSMS and has an awkward syntax in Sql.

Script Schema
Generate a “Create table .. “ script for the selected table. This script is intended for making a quick copy of a table either in the same database, but more normally in another one. It will script the fields and primary key correctly, as well as defaults and computed fields, but it may omit some more of the esoteric keywords. It is probably better to use SSMS for a serious purpose, but this will be OK for building up a development database. The script is loaded into a Source display window from which it can be copied to a query window.

Script POCO class
? Dubious merit ? If you haven’t encountered the acronym POCO then you will probably not find this option much use. It opens a window that will generate a C# class from the selected table for use in one of three ORMs: Entity Framework, LINQ2SQL and PetaPoco. The terms are explained later when the form and its functionality is described in detail. The feature is flagged dubious merit because the author has not had cause to use it in a real-world situation. A better label might be “Seemed like a good idea at the time”.

Copy
! **Recommended** Uses the table script generator and Sql Server bulk copy to create a copy of the selected table schema in another database and copy data from the selected database to the target database. It is recommended and dangerous as it is the only feature in DbViewSharp that will destroy data in a database not flagged as a development database. This will happen when a table of the same name and with the same schema exists on the target database. You will be warned, but not prevented from overwriting the data as in most cases you will be wanting to replace the data. The feature can be used to: a. create temporary data backups; b. quickly populate a new database with configuration and static data lookup tables; c. transfer production data to a safer environment

for analysis. It is as fast and more convenient that the equivalent SSMS functionality unless this is a regular task. The dialog is described in more detail later in the Dialogs section.

Export

Displays a form that enables you to export data from the selected table to a CSV file. You can select fields, set a filter on the rows to be exported and generate either UTF-8 or UTF-16 files. For more information see the detailed description of the Export form in the next section.

Import

Displays a form that enables you to import data into the selected table from a CSV file. For more information on this function see the detailed description of the Import form in the next section.

! New table
** Cautiously Recommended ** In developer mode this will pop-up a form window that enables you to enter the most commonly used information to create a table with a limited number of columns. Not all of the options or data-types are available, but enough are for situations where you either want to create a scratch table to check out some behaviour or for knocking up a quick prototype. You can even use it to create a temporary table as there is a facility to copy the script to the clipboard. The dialog is described in more detail later.

Physical Sizes
This option creates a result grid that shows the physical size of each table in the database sorted in descending order of size. It displays the size in three scales: Gb, Mb and Kb. Other columns in the grid are copied from the system procedure sp_spaceused from which the report obtains its figures.

Quick Diff
Compare the table schemas with those in another database and report the differences. This is another “itch scratch” feature which is a poor substitute for the leading commercial product or indeed several open source applications. However from the point of view of a DbViewSharp user it has the advantage of being on hand and not requiring any boring set-up of connection and filtering of options because the other products compare every little detail. This feature is used when two databases are supposed to be the same or very similar (say one on a development server and one on a UAT server) and programs are working on one, but not the other. The Quick diff report will perform a quick scan of the table schemas and report any tables or fields in one database not in the other and for each field, changed type or width, missing indexes or keys, changed computed columns. You can get a good idea of the scope of the comparison by looking at the fields result grid. If it’s reported there it’s checked in the Quick Diff report. See the Dialog description for more details.

Show, Views (Toolbar, Ctrl+W)

Displays of the views in the database. Double-click on a row in the grid to open a source display window. Assuming your database connection has sufficient permissions the source of the view is present. The other items in the context menu perform in a similar way to their counterparts in the tables grid context menu.

View Source: ** Recommended ** Displays the view SQL source .
Data: Displays data from the executing the view.
Fields: Displays the view column information.
Dependencies:
Displays objects that depend on the selected view and objects that the selected view depends on.
Find in source: Searches source code for the view name
Quick Diff: compares the view names and view source in two databases.

Show, Procedures (Toolbar, Ctrl+P)

Displays of the stored procedures in the database. Double-click on a row in the grid to open a source display window (permissions allowing). The other items in the context menu perform in a similar way to their counterparts in the tables grid context menu.

View Source: * *Recommended* * Displays the view SQL source .
Dependencies: Displays objects that selected procedure depends on.
Find in source: Searches source code for the procedure name
Quick Diff: compares the view names and view source in two databases.

Show, Functions

Exactly the same comments apply to this option as to the Show, Procedures option.

Show, Triggers

[Undocumented at present] similar to Show, functions.

Show, Indexes

Like a number of the grid displays it is occasionally useful to see objects normally shown in the context of a single table as a group to give you a feeling of how much use is made of the feature. The grid display for indexes is one of these. The columns are mainly directly reporting information from the underlying Sql view with little attempt to interpret the data further. There are no significant context menu options.

Show, Foreign Keys

The grid display shows the tables and columns involved in each foreign key. The column headed "DISABLED" indicates whether or not the integrity constraint is active. The column headed "TRUSTED" further refines this information. Generally the latter two columns either read DISABLED: blank, TRUSTED: Y when the FK integrity is enforced or DISABLED: Y, TRUSTED: * N when it is not. The context menu offers a variety of scripting options for removing and adding back foreign keys. In development mode foreign keys can be added, removed or disabled directly.

! Disable
Turns off integrity checking for the selected FK. (Display to DISABLED: Y, TRUSTED: * N)

! Enable
Turns on integrity checking for the selected FK. (Display to DISABLED: blank, TRUSTED: Y). There is a chance that this will fail if there is a value in the foreign key field not in the primary key.

! Remove
Deletes/drops the selected foreign key.

Script, Create this
? Dubious merit ? Generates a script to create the foreign key. This is old code and is not well researched.

Script, Create all
? Dubious merit ? As above, but for all foreign keys.

Script, Remove this
? Dubious merit ? Creates a script to remove the key and appends a script to restore it. This is for the scenario where you need to temporarily remove a FK.

Script, Remove all
? Dubious merit ? As above but for all foreign keys. Flagged dubious as the restore script will not create the keys with integrity checking on.

Quick Diff
Compares the Foreign keys in two databases. This could be used in the scenario where a command or stored procedure is causing an integrity violation error in one database, but not another. The report will show missing foreign keys and ones where integrity is in on in one database, but not in the other.

Show, Computed Columns

This is a feature for use when exploring unfamiliar databases. It shows at a glance the number of computed columns in the database and the actual computation. It is easier to read the formulae in the grid if you set the font to fixed width via the menu item Options, Font, Fixed. There are no significant context menu options.

Show, Dependencies

? Dubious merit ? Creates a grid based on the contents of the Sql Server dependencies system view. It is flagged as of dubious merit because there are questions about the accuracy of this dependency information and certainly omissions from the view have been encountered in practice. Context menu options are:

Show Source: Shows the source of the dependent object.
Show tree: Creates a dependency tree.

Show, Permissions

Displays the permissions granted on objects to groups or users. The scenario for use of this grid would be in providing a quick visual check that not permissions are omitted from objects. With such a display it is easy to spot objects with anomalous permissions, such as the case where a user has update permission for all tables except one. Investigating the objects with unusual settings may prevent a hard-to-trace bug from being introduced.

Sql Server

The Show main menu option holds a collection of items that are likely to be available on other database drivers; eg. all drivers should be able to create a grid display of table names. However as DbViewSharp is architected to support multiple database drivers there needed to be a menu item to hold grids and functionality specific to the driver.

Sql Server,Owners

This grid displays all the owners of database objects in the current connection. An owner is either a database principal or a schema. To enable management it also displays a count of tables, views and procedures owned by each. Context menu options are development mode only.

! Add Schema
This option will prompt you for a schema name and create it if it is valid and not already in use.

! Remove Schema
This will removes the selected schema provided there are no objects attached to it. The object display shows the count of tables, views and procedures. The schema will not be removed if any of the count values are non-zero.

The intention is to develop further manipulation features in particular as the ability to move objects from one schema to another. However preliminary research on this subject indicates that this will be a complicated task due to the problem of how to deal with objects referenced in views and stored procedures.

Sql Server, Processes

Shows a display of all connections to the server of the currently connected database. Useful for determining how busy a server is. Note this requires high priviliges. If the connection does not have sufficient then it will only display its own process information. A useful column is the "blk" column. When set to a non-zero value instead of 0 it indicates that the process is blocked by another. The number is the Id of the blocking process.

Context menu **Last Command** will sometimes show interesting Sql for a process which could provide more clues as to what is going on.

Sql Server, Users

Creates a grid that combines database principals and roles. Principals are sort of logins and roles are collections of permissions. They can be combined in a flexible manner to create complicated permission structure. Examining this grid may help unravel that a little.

Sql Server, Locks

Extracts and displays the significant locks from the system table. In most cases it can figure out the object the lock is applied to. Useful when queries are taking longer than expected or timing out for determining that it is a lock problem. Use this and the processes grid to discover the applications that have locked the objects.

Sql Server, Jobs

Displays a grid of the jobs on the server. There are two significant context menu options.
Job Steps: Shows the commands for each step of the job. This is an overlooked area to search when trying to establish where a stored procedure is being used. The name of the job and the rest of the job steps put its use into context.
Job History: Useful to check when tracing problems that may have been caused by a job either being disabled or failing at a particular step.

Sql Server, Linked Servers

Shows linked servers configured for use in the current server. Useful to discover the extent of interconnections between Sql databases and servers before planning to retire a server. Also useful for checking that linked servers are correctly configured to link to others in the same environment; ie. Checking that there are no development servers linked to production servers or vice versa.

Sql Server, Publications

Lists the replication publications set up on the database. Double click a publication to view the tables included. If replication is not enabled on the database the grid will display a message to that effect.

Sql Server, Articles

Similar to the previous item, but lists the publications and tables. If replication is not enabled on the database the grid will display a message to that effect.

Sql Server, Service Broker

This is actually a gateway to a comprehensive set of queries on service broker objects. It arose from the merger of DbViewSharp with a sister application. Rather than overload the menu system with too many Service broker items, which are a specialised requirement, they were tucked away behind this single menu option.
Once you display the initial Service broker object grid use the context menus to navigate the various Service broker grids available. Available grids display the following Service broker objects: Messages, Contracts, Queues, Services, Transmission Queue, End Points, Routes. In addition you can check the configuration of the database for Service broker and create scripts to add messages, contracts, queues and services. The Service broker sub-system deserves much more description, but in a future edition of this document.

Action

Action, Search Grid ([Go], Ctrl+S)

The same as pressing the Search bar Go button. Unlikely ever to be selected as a menu option, but is a convenient way to implement the keyboard command Ctrl+S.

Action, Clear Search (Toolbar)

Performs the same functionality as pressing the slim white rectangle toolbar icon.

Action, Search Source

Pops up the Search source dialog into which you enter the text to search for and choose the type of objects to search the source of. On executing the search a list of objects whose source contains the search text is displayed in the grid. Double click on a row in the grid to view the source of selected object with the search text picked out.

Action, View History (Toolbar)

Displays a grid of all the previous grids created in the session for the current tab. Double-click on any row to refresh and redisplay that grid. In practice it is far more convenient to use the tool bar icon than this menu option.

Action, Refresh Grid

Causes the current grid to be refreshed periodically. The menu item has a sub-menu with a variety of refresh rates. Be careful with the selection of rate in case the grid is expensive to refresh. After the each fetch the new data is compared to the old data. New rows and rows with edits are displayed in red. Rows that were previously marked red and not changed are faded to orange, then green and blue to present a more complete picture of how the data is changing in the grid. Note: the refresh directive is not preserved in history and will be automatically switched off if you switch to another grid. The intention is to select the data to observe, set the refresh rate and observe it. Also note: it is not advisable to click to display the context menu while this option is on.

Action, Test

? Dubious merit ? A menu option used by the developer for experimentation. It should be removed for official releases, but sometimes is not. You are advised not to select this option.

Data

Data, Modify DataView (Toolbar)

Recommended This item is only available when a table data grid is displayed. When selected this pops up a an important dialog that allows you to modify the display of the table data. You can change the columns displayed, the order of display and the sort order. Sometimes the columns selected for the search bar are not appropriate or too slow. You can change the selection of them in this dialog. You are recommended to study the detailed description of the dialog as fluency in modifying table data grid configuration is very likely to improve your experience of DbViewSharp.

Data, Filter (Toolbar)

This item is only available when a table data grid is displayed. When selected this pops up a filter selection dialog pre-set with the content of the current grid cell. The scenario is that you see an item of data in the grid and are interested in all other rows containing the same data. It was intended as to be a convenient and flexible way to enter a search to filter the data rows, but in practice the other search strategies are preferred. See later for a detailed description of the dialog, which is a little strange.

Data, Aggregate (Toolbar)

Count the instances of values for data in the column of the current selected cell in the grid and present them ordered by most numerous to least numerous.

Data, Sum

For numeric fields sums and averages the value of the data in the column selected for the rows currently filtered. Note the search bar filter is not applied for this operation.

Data, Min and Max

Displays the minimum and maximum values of the data in the column selected for the rows currently filtered. Note the search bar filter is not applied for this operation.

Data, Fix Bad DataViews

It may sometimes happen that you modify a view of a large table that causes a timeout error. The most common cause of this is selecting an inefficient sort column. The application may then get itself into a catch 22 bind; you cannot modify the view to remove the sort column until the grid is displayed, yet the application cannot generate the grid data until the sort column is removed. At this point in the development of DbViewSharp the design goal is still responsiveness before flexibility so instead of increasing query timeouts this menu option will attempt to locate and remove the potential cause of the timeouts.

Tools

Tools, Ad-hoc Query Window (Ctrl+Q)

! **Recommended** Opens or closes a window for executing ad-hoc Sql. It is not designed to compete with the query window of SSMS, but is useful for performing quick tasks without switching out of DbViewSharp. The window shares screen space with the results grid, having formerly been implemented as a "tear-off" separate window. Since the rest of DbViewSharp is mainly read-only this used to be the only area where you could use the application to alter table structure or even delete database objects entirely. One caveat when using this however is that sometimes you may be connected to a dev version and a production version of the same database, but connected with a higher power account than necessary for pure reading in order to access view and stored procedure source. In this scenario you must be careful when executing schema-altering Sql on the dev database that the ad-hoc window truly is connected to the dev database, because the Sql could be valid to operate on both the dev and production versions.

Tools, File Selector (Toolbar)

Opens or closes the File Selector panel. This is intended to help when writing Sql by providing functionality for copying the column names of a selected table into the clipboard.

Tools, Clipboard to comma-list

**Recommended*.* The scenario is that you discover a group of data rows in a grid that require further analysis in a complex query in SSMS. You want to restrict the query by an "Id in (999,999…)" clause where the numbers are the ones you have just marked and copied from the grid. When you paste them into the query window they appear, naturally enough, as a column which you then have to convert by deleting the line feeds and inserting the comma. Instead of that selected this option and the conversion is done for you in the clipboard. It is recommended because it is used so often. It works great for number columns, but with text columns in some cases you will want the text adorned with quotes and in others you won't. At present the text is unadorned, but you can achieve a quoted list by using this and the next item in concert.

Tools, Set Clipboard Template

? Dubious merit? A number of times you will copy text from a grid display and then apply a set of standard edits on the list to achieve your goal. For example you see a block of old tables that can be

dropped, select the list and then edit the list to add the DROP TABLE text in front of each to create the desired script. This option allows you to define a template into which data copied from the clipboard will be inserted. So to extend the example before copying the list of tables select this option and enter DROP TABLE {0} as the template. C# programmers will recognize that the {0} will be replaced with the table name. Note when you copy from the grid now the cells will flash a different colour to the usual red to remind you that the data will be formatted. This clue is useful to remind you to clear the template when you are done with it.

To return to the problem of preparing a list of text items to add to a "field in ('aaa','bbb'..) clause, first set the template to '{0}'; then copy the required items; finally select Tools, Clipboard to comma-list (lastly clear the template).

This feature is flagged as of dubious merit because it is normally too fiddly to be worth using (author's opinion only).

Tools, Sql Template Collection

Displays a grid which you populate with any snippets of useful Sql you always have to look up the syntax for before you can use. Examples: using a CURSOR, ROW_NUMBER syntax, CTE Expressions. Before you enter any of your own code the grid will display a single sample. Use the context menu options View/Edit, Add and Delete to manage the collection.

Tools, Create Portable DbViewSharp

Displays a form that prompts for a folder location into which the program will copy all of the system files and optionally the database connection information. This creates a special "portable" version of DbViewSharp with all its required files in a single subfolder. If the folder is created on a USB drive then this can be used on other machines without the need to download and install a full version.

Options

Options, Font

There are three options for the font used by the grid display: Default, which is what you see on first starting; Fixed, which can be useful for spacing out data to read more clearly; and *(? Dubious merit ?*) Unicode which was designed for viewing non-ascii characters in the days before the default font was upgraded to do this perfectly well. As mentioned earlier the fixed font option was developed to improve the readability of the computed columns grid.

The font reverts to default when the application is restarted.

Options, Tabbed Windows

The behaviour of tabbed windows was a bit of a departure from previous behaviour so this option was added in case existing users hated the change. The default is for tabbed windows to be enabled, which means that connecting to a new database will create a new tab and a separate history. Disabled, a new connection replaces the existing grid display and the history will always be able to access any previous grids.

This option is persisted from one session to the next.

Help

Help, About

The about box shows version and build date. It also gives an E-mail address for feedback to the developer. Be patient though as having only ever received one message to date he does not check the inbox very frequently (any more).

Help, Show context menu help

This is a toggle that switches on explanatory text to the context menus. It looks weird but it is a way to communicate a little more what each context menu item does. It was designed more for users too lazy to read this guide so if you are reading these words then it is probably of *? Dubious Merit ?*

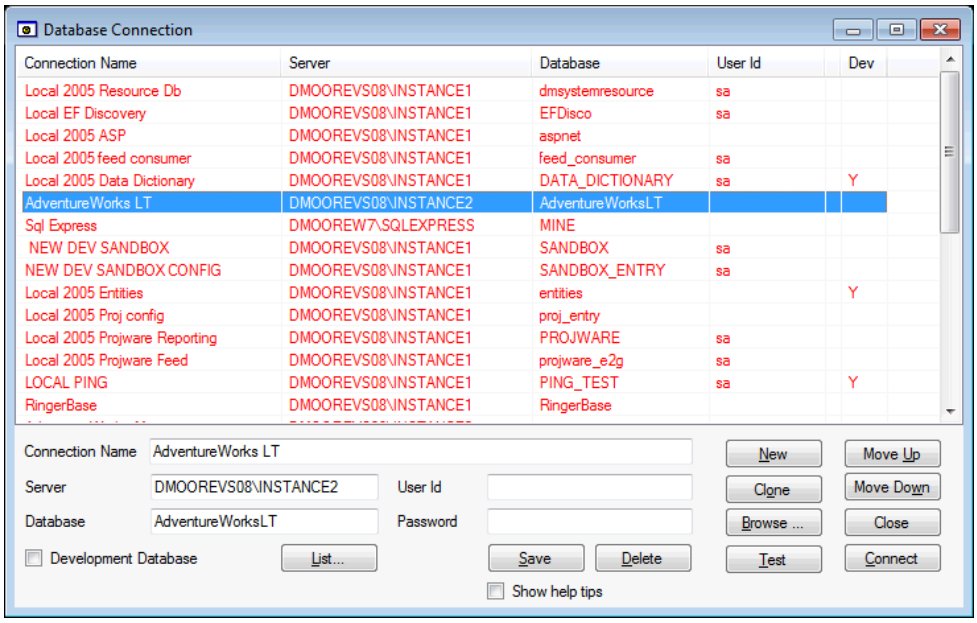
Help, Where is the config file

As a developer it is good to get access to the configuration and data files of the applications you use just in case they are hackable. The DbViewSharp configuration file is quite hackable, but being Xml it is easy to break. Since application relies on Windows routines to select the folder in which it will reside it is not possible to say in advance where it will be written. This option will display the location folder by Windows and the name chosen by the developer. As a bonus you can select to open the folder containing the file, but please heed the text of the messages box. At the time of writing there is no automatic recovery from a broken configuration file so remember where the file is located and make a backup before manually editing it.

Dialogs in Detail

Database Connection

The Database connection dialog has already been introduced in overview. This section discusses the controls on the dialog in more depth.



Connection list:

This List control displays or will display details of all the connections you have entered. The list is sortable and when you have many connections you can filter the list just by typing a few letters. A filtered list goes red and you restore the full list by pressing the escape key. There is no attempt made to hide the password strings. It is assumed that you are operating in an environment in which you trust your colleagues.

Connect directly by double-clicking on a line. Right-click to access a range of functions you can perform on the database. See below for a description of the right-click menu items

Connection Name: [ ]

Enter a friendly name for the connection that describes the database you are connecting to. Hint: it may be useful to indicate whether this is a development or production database eg. PROD Sales or DEV Documentation. As mentioned above a consistent naming convention will be a help as you add more and more connections to the list.

Server: [ ]

Enter the database server machine. Take care with spelling as it can sometimes be slow for the system to detect that the (misspelt) server does not exist. Remember to add the instance and port number if necessary eg. ServerA\instance2,5550. Use whatever you do to connect via Sql Server Management Studio. Localhost will workfor your own machine when connecting to SQL Express (try: localhost\sqlexpress).

Database: [ ]

Enter the database name. Press the [List] button to select from available databases, but for this you will need to be connecting with NT authentication or have already entered a valid username and password. The design of DbViewSharp means that you need a separate connection entry for each database on a server. It means more configuration work, but eliminates that fiddly step of switching to the correct database every time you connect to a server. Also note that a using a combination of

the [Clone] button (see below) and [List] makes it quite speedy to add configurations for a set of databases on a server.

User Id: [ ]

Enter a user account name to connect to the server and database. Leave this field and the password field blank if you are connecting with NT authentication.

Password: [ ]

Enter the password associated with the user name in order to connect to the server and database. Leave it blank if you are connecting with NT authentication.

[List]

Press this button to list the databases available on the server entered and to select one of them for the connection. You must either be able to connect to the server with NT authentication or have entered a username and password in the appropriate boxes.

Development Database: [X]

Check this box ONLY if the database you are currently connecting to is a development database where modification of data or schema is permitted. When checked the application will expose dangerous, but useful, functionality and more fluent data-editing, eg dropping of tables and adding of columns and foreign keys. Please don't be tempted to check this item for a production database; a) because you can't be sure that the options selected for the commands will always be exactly what you want and b) because it is too easy to lose track of which database you are connected to and do something rash in the wrong database. For production I would always script the change and test it on a non-production database first.

[Test]

Perform a connection test on the current connection or on connection details just entered. It is good practice to test a new connection just before saving the details.

[Save]

Save all the edits you have made to connections since either opening the dialog or last pressing save. Note: press Close without Save to abandon current edits.

[Close]

Close this form without saving any edits made or connections added since the last time you pressed the save button. This will return you to the application without changing the current connection. If you press Close instead of connect when this form is displayed at the start of the application then you will enter the application with reduced functionality.

[New]

Create an empty slot for details of a new database connection. You then need to enter details for Connection name, server and database at least. You will additionally need to supply a username and password unless you are connecting using NT authentication.

[Clone]

Press this to create a new connection from a copy of the details of the current connection. It is useful for when you wish to set up connections to many databases on the same server. Alter at least one entry before saving the new connection.

[Delete]

Remove the connection highlighted in the list. You will not be prompted for a confirmation. If you delete the wrong connection then press [Close] to leave the screen without saving changes.

[Help]

Activates tooltip help. Jiggle the cursor over a control and you should see a tool tip with helpful information.

[Browse]

This appears to be an obsolete function. It allows you to select a file name for the connection name box, but will not create a valid connection from there. It is best to ignore this button.

[Connect]

Connects to the database selected in the list. If you are in tabbed window mode this connection will create a new tab and start a separate history. If you have turned of tab mode then the present grid will be replaced by a grid containing the tables of the new connection. In the latter case you will be able to access the previous history; eg pressing the back button will return you to the old connection and the grid just replaced.

[Down] [Up]

Re-order the connection list by moving the current selection up or down. Hint: it is sometimes easier to re-order connections when the list is cut down.

Connection List right-click menu.

All options act on the connection of the currently selected line.

Connect:

Same as pressing [Connect]. This is the default option activated by the *Enter* key.

Test Connection:

Same as pressing the [Test] button

Show Database Information:

This displays a screen showing a variety of information about the database including SQL server version, case-sensitivity and physical size. Press [Drive info] for a further screen that displays physical file information for each database file attached to the SQL server instance. Use this to assess how close to filling the disk a database is and whether you have any unnaturally large log files that could be truncated to recover space.

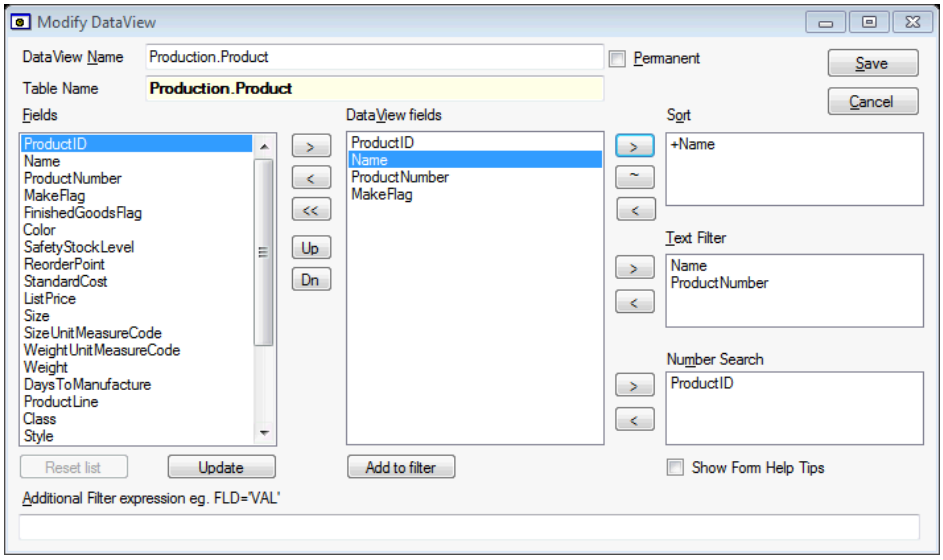
Copy to clipboard as...

Formats the connection information in a variety of ways suitable for pasting into other applications eg. a C# program, a web config file, a sqlcmd or BCP command line.

Modify DataView

This is the second most important dialog in DbViewSharp. You use it to change the presentation of data from tables and can configure the columns displayed and their order of presentation, as well as setting a row sort order and selecting columns to be scanned when text is entered in the search bar. It is here that you can create permanent DataViews, which will then appear in a the dropdown next to the [Go] button on the main screen.

The screen shot shows the dialog in the process of configuring an AdventureWorks DataView.



The dialog contains a lot of controls, but it is worth becoming familiar with them as with practise you might find using this feature will reduce the amount of ad-hoc Sql you need to write to access rows in a table.

DataView Name [ ]

This is either the name of the table on which the view is based or a name that you enter. If you enter a new name then the DataView will be saved and appear in the drop-down list on the main screen. If you leave the name the same as the table name then this DataView will not be saved after the application is closed.

[x] Permanent

If this box is unchecked then the DataView configuration will be discarded when the application is closed. Check the box to save it for later use. As soon as you change the DataView name this control is automatically checked as experience showed that really the only reason for changing the DataView name is when you want to keep the view.

Table Name [ ]

Shows the table on which the DataView is based. You cannot change it. It's there for information only in case you change the name of the view too much.

Fields (List)

This list control displays the available fields for selection into the grid. Double-click or drag and drop a field into the grid field list to the right. Use [>] to select multiple fields. If you select a field that has already been selected then nothing happens.

When the Fields list has focus typing letters will apply a filter to the list. When the list is filtered the text is changed to red. Backspace will undo the effect of the previous letter typed. Selecting a field or

fields will restore the full list. Pressing the space bar will select the current field. You can also restore the full list by pressing the [Reset list] button or the

[Esc] key. Dragging and dropping is **Recommended ** for single field selection as you can choose the position in the target list to drop them.

[Reset list]

This button is a means to return a filtered field list to the full list. The text will turn from red to black in a visual confirmation of the action.

[Update]

The design objective of DbViewSharp to be really responsive causes a problem with this dialog and the Fields list in particular. The dialog obtains its field list from the default configuration of the tabe data grid when it is first invoked. Any changes to the schema of table it is based on will not be reflected in subsequent use until the application is shut down and restarted. The situation is worse for permanent DataViews because the incomplete field list is persisted in the data file.

Rather than build complicated and potentially sluggish checks for table schema changes into the system the solution adopted is to provide a means for a manual refresh of the list. This button performs that referesh, adding new columns to the Fields list.

Press this button to update the field list from the database. You will need to do this when a field has been added to the table which does not appear in the list.

DataView Fields (List)

Lists the fields that will be displayed in the DataView in the column order that they will appear. Select from the Fields list on the left and double-click or drag and drop them into this list. Remove fields either by dragging them out of the list or pressing [<] on the left. DataView fields can be added to the lists on the right of this to select them for sorting or to be filter fields.

Fields buttons [>] [<] [<<] [Up] [Dn]

These are the buttons located between the two fields lists on the dialog. The two single-arrow buttons will add and remove fields, the **Recommended** double-arrow button will remove all fields following the one currently selected in the DataView Fields list. ? Dubious merit ? Up and Dn will move a field up or down the DataView fields list (dragging and dropping is actually a faster way to do this).

[Add to filter]

Press this button to add the selected DataView field to the Additional filter expression below. The default is to create a not null search, however this is readily changeable.

Additional Filter Expression [ ]

** Recommended ** Use this to enter a SQL clause to restrict the rows displayed by this DataView. Be sure to use correct SQL syntax or it will generate an error. Eg. remember to put single-quotes around literal strings.

Sort (List)

Drag and drop fields into this list from the DataView field list or select them with [>]. By default the sort order will be ascending and the fields will be prefixed with “+”. Toggle this by pressing [~]. Take care when sorting large tables as again the design objective of DbViewSharp to be really responsive can cause a timeout situation. Refer to the menu option Data, Fix bad DataViews should you get caught with a problem like this. For normal-sized tables there is not usually a problem with sorting.

Sort buttons [>] [<] [~]

These are the buttons located between the DataView fields list and the sort field list. The two single-arrow buttons will add and remove fields, the [~] button will toggle the sort order on the selected sort field from ascending to descending order and back.

Text Filter (List) and buttons [>] [<]

Lists the fields to search when text is entered into the filter box on the main screen. Press [>] to add a field selected from the DataView fields and [<] to remove a field from the filter list. Note: only character fields can be added to this list. Numeric search fields should be added to the Number Search list below. It is common to find that the default view included an inappropriate item in this list which you should remove.

Number Search (List) and buttons [>] [<]

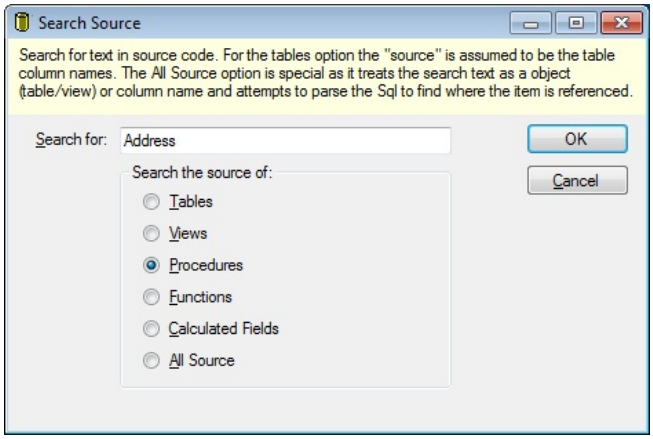
Lists the fields to search when a number is entered into the filter box on the main screen. Press [>] to add a field selected from the DataView fields and [<] to remove a field from the filter list. Note: only numeric fields can be added to this list. Character-based search fields should be added to the Text Filter list above. As previously noted date fields and other esoteric types are not suitable for inclusion in this list.

[x] Show Form Help Tips

A control added for users who do not read manuals. When checked it enables fairly extensive tool tips on all of the controls.

Search Source

Searching source code is a useful, but incomplete feature. If it was going to be thorough there would be more categories of objects to search, for example triggers. This dialog pops up when you select Action, Search Source, press the associated toolbar icon or select Search Source from one of the context menus.



Search For [ ]

Enter the text to search for. Searching source is mainly about tracking down where tables, fields and views are used in views or stored procedures to cover up for the undependability of Sql server dependency information. It needn't be limited to that though. If you have lost track of a particular piece of Sql then any half-remembered fragment of the code, if it is sufficiently uncommon, will serve as a good starting point for the search. If you half-remember the name of the view or procedure that is even better as you can always filter the search result grid.

Although requested (once) there is no facility to search more than one type of object at a time. Again the responsiveness objective is used to defend this decision. Searching text of all stored procedures on a heavy duty database can take time.

( ) Tables

** Recommended ** For tables the "source" is considered to be the column names. So selecting this option will search all tables for columns that include the search text in their name. Consider searching out date fields by entering "DATE". Experience has shown this feature to be useful.

( ) Views ** Recommended **

( ) Procedures ** Recommended **

( ) Functions

Search the source of the selected object list for the target text

( ) Calculated fields

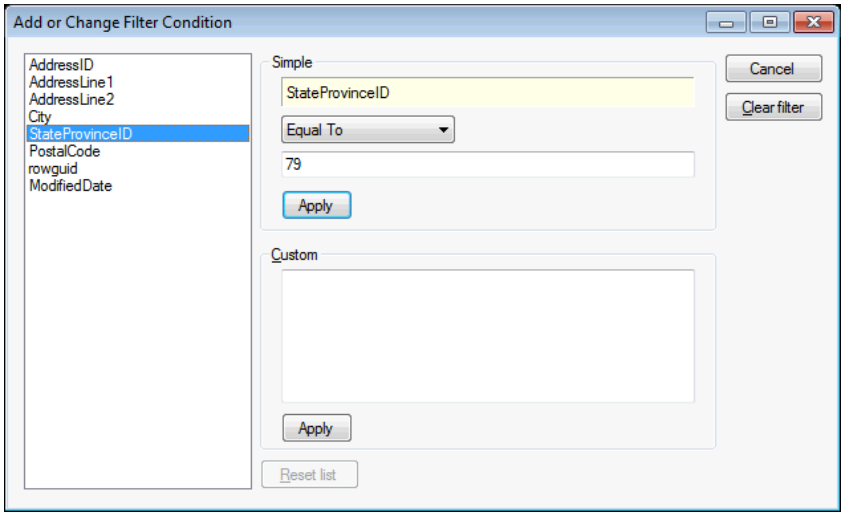
? Dubious merit ? Search the computed column source.

( ) All Source

** Recommended ** Searches all objects whose source code is stored in the table sys.comments. Objects include: procedures, views, functions, triggers, defaults and computed column definitions. This option matches whole words instead of partial text. This is useful for example, when searching for a column name such as "Id" in a database where many columns are also named xxxxId (as in CustomerId). The other searches will match both Id and xxxxId, however this option will only match Id.

Add or Change Filter Conditions

This dialog appears when you select menu option Data, Filter… or press the funnel toolbar icon. It is used to set a temporary filter clause on the current table data grid. The convenience feature it implements is when a particular grid cell is selected is to set the filter column selection to the column of the selected grid cell and the search value to the content of the selected grid cell. When a filter is active it is shown in the status bar. A neat **recommended ** short-cut to clearing the filter is to click on the status bar panel. If you click the filter when no filter is active then it will cause this dialog to open.



Columns (List)

This list control displays the columns of the table. Click on or navigate by the arrow keys to select the field to search on. When the list has focus typing letters will apply a filter to the list. When the list is filtered the text is changed to red. Backspace will undo the effect of the previous letter typed. Selecting a field or fields will restore the full list. Pressing the space bar will select the current field. You can also restore the full list by pressing the [Reset list] button or the [Esc] key.

[Reset list]

This button is a means to return a filtered field list to the full list. The text will turn from red to black in a visual confirmation of the action.

Simple [ ]

Read-only display confirming the column selected for the search.

Comparison [V]

Select a standard Sql comparison operation from the list. The default will be equal to if the selected grid cell contains data. If the cell is empty then the comparison defaults to “is null”.

Search value [ ]

Enter the value to compare to the data. You do not need to add quotes for text and date fields, but date fields and number fields must be a valid format otherwise there will be a Sql error.

[Apply] (upper button)

Performs the simple search defined by the controls above the button

Custom [* multiple line text box *]

Depending on whether or not a filter is already defined this box will either display the current filter condition or a “non-null” search on the selected grid cell column. By typing directly into this box you

can extend or overwrite what is there with an additional search clause. The end result must be valid Sql to add to the Where clause of the query on the current table. Press [Apply] beneath the box to use the custom filter instead of the simple filter.

On scenario for a custom filter is when the simple search filter returns too many rows and you wish to further filter the result. Simply press the funnel to load the existing filter and add to the search clause to make it more restrictive.

[Apply] (lower button)

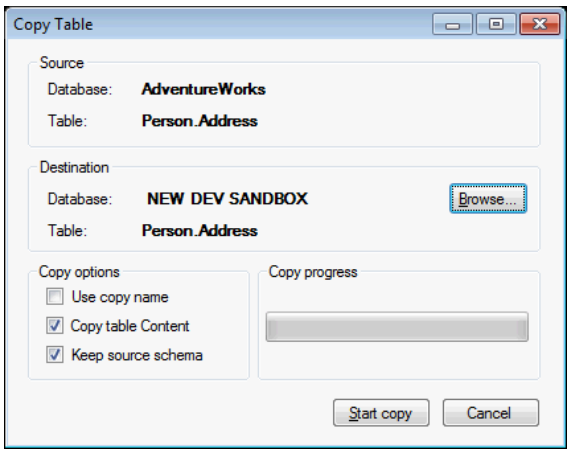
Performs the custom search defined in the text box above the button

[Clear Filter]

Clears the search filter and redisplays all the rows in the DataView.

Copy Dialog

The table copy dialog is accessed from the table grid context menu option “Copy”. On selecting Copy for the first time in a session the connection form opens for you to select a target database before displaying the dialog. On subsequent selections the dialog opens directly assuming the last selected target database will be used again.



Source Database
Source Table
This confirms the selection just made from the context menu

Destination Database
Destination Table
The database will be the one associated with the connection just selected or used in the previous copy operarion. The table will initially display the same name as the source table, but this may change when the “Use copy name” box is checked.

[Browse]
If the database is incorrect then this re-displays the connection screen for you to make the correction.

Copy Options
[x] Use Copy Name
Check this to change the target name to a safe name. There are two reasons for using the safe name here. Firstly you may want to create a backup of a table in the database you are already connected to. This is not possible unless the source table name and target name differ. With this scenario in mind the safe name is constructed from the source name plus a timestamp component. This will enable severable clearly dated backup copies to be kept plus keeping the tables grouped when the table names are displayed in sorted order. The other reason *? Dubious merit ?* is that when a table exists on the target database and you want to be cautious and compare the contents of the copy with the original before replacing the original with the copy.

[x] Copy table content
Check this box to copy the content as well as the schema. Since you almost always do want to copy content this box is pre-checked when the dialog is loaded.

[x] Keep Source Schema
Where the table belongs to a schema or user that is not the dbo this checkbox is enabled so you can select whether or not to create copied table in the original schema. The schema must exist on the

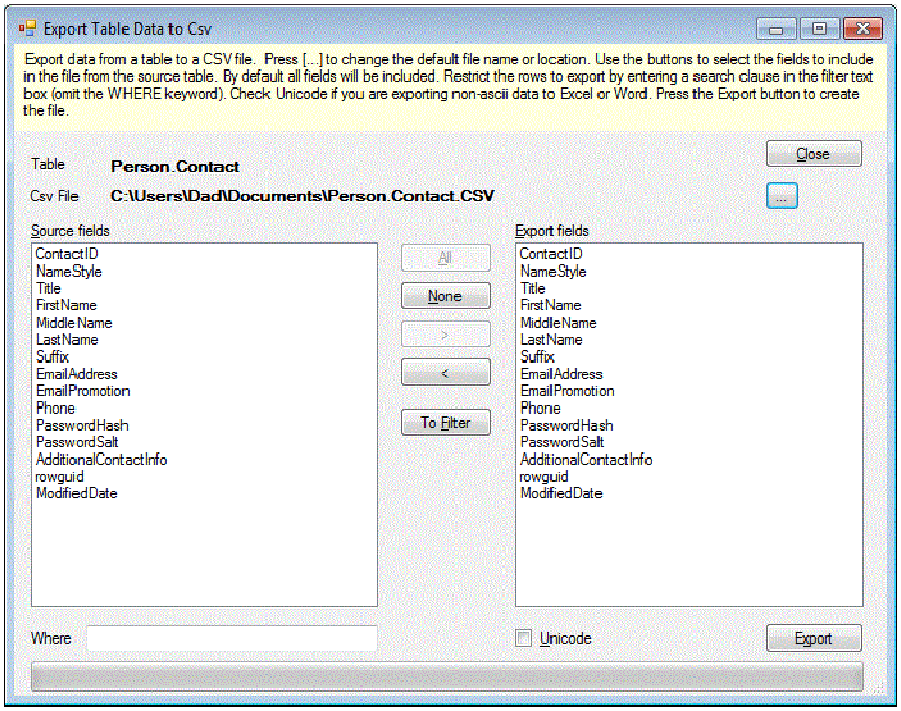
target database otherwise a Sql error will be generated. Where the table belongs to dbo this option is not applicable.

[Start Copy]
Attempt the copy. During the copy process, which may be long for large tables, the button changes to a cancel button, which can be used to abort a long copy. There are a number of scenarios to discuss here as there are plenty of failure points. The application may not handle every scenario perfectly, though it takes great steps to ensure that no data in the target system is destroyed before the copy completes.

The copy operation will fail if your connection does not have create table privileges on the destination database. If a table of the same name exists on the destination database the routine first creates a table with the safe name and copies data to that. Once this is complete it will attempt to transfer this data to the target table before removing the table with the safe name. If the target table does not have exactly the same schema as the original table then the data-transfer will fail, but the safe-name table will not be destroyed. If you cancel the copy the part-copied tables will not be removed.

Export to CSV File

The Export to Csv File dialog is displayed when you right-click on a table and select Export from the context menu. Fields to select are displayed in a list on the left. Fields to export are displayed in a list on the right. On entry all fields are automatically selected for export.



Table

Confirms the table containing the data is to be exported. This cannot be changed from within the form. If you have selected the wrong table just close the form, select the correct table from the grid and try again.

Csv File

The output file. On entry the folder is set to the document folder and name is derived from the table. Press [...] to change this

Source fields

List of fields in the table available for export. Select individual fields and use the [>] button to select it into the Export fields list. You cannot select a field twice.

Export Fields

List of fields to be exported. Remove fields from this list either by pressing [<] or clear the entire selection with [None].

[All] [None]

Select all fields for export or clear all export fields. Use these as appropriate.

[>] [<]

Select for export or remove from the export field list. Since there is no means of changing the export field order other than by deleting fields and then reselecting in the correct order you might need to use these a lot if you are fussy about order.

[To Filter]

Copy the selected field from the Source field list into the filter/Where clause text box. This is just a convenience when you wish to define a row filter clause.

Where [ ]

Enter a valid SQL WHERE clause to apply a filter to the rows being exported. Do not include the keyword “WHERE” as that is added for you. Be careful in construction of the filter as it is passed directly to the driver without validation. Should the export generate an error one of the first items to check for a problem would be the text you entered in this box.

[x] Unicode

Check this to generate a file in UTF-16 format. When would you do this? Well if the data you are exporting contains non-Ascii characters, eg Greek or Russian alphabet, Chinese or Japanese characters, then saving to a Unicode file causes fewer problems when loading the file into MS Word or MS Excel. Otherwise leave it unchecked to generate a UTF-8 file. This will be suitable for import of any characters via DbViewSharp.

Export

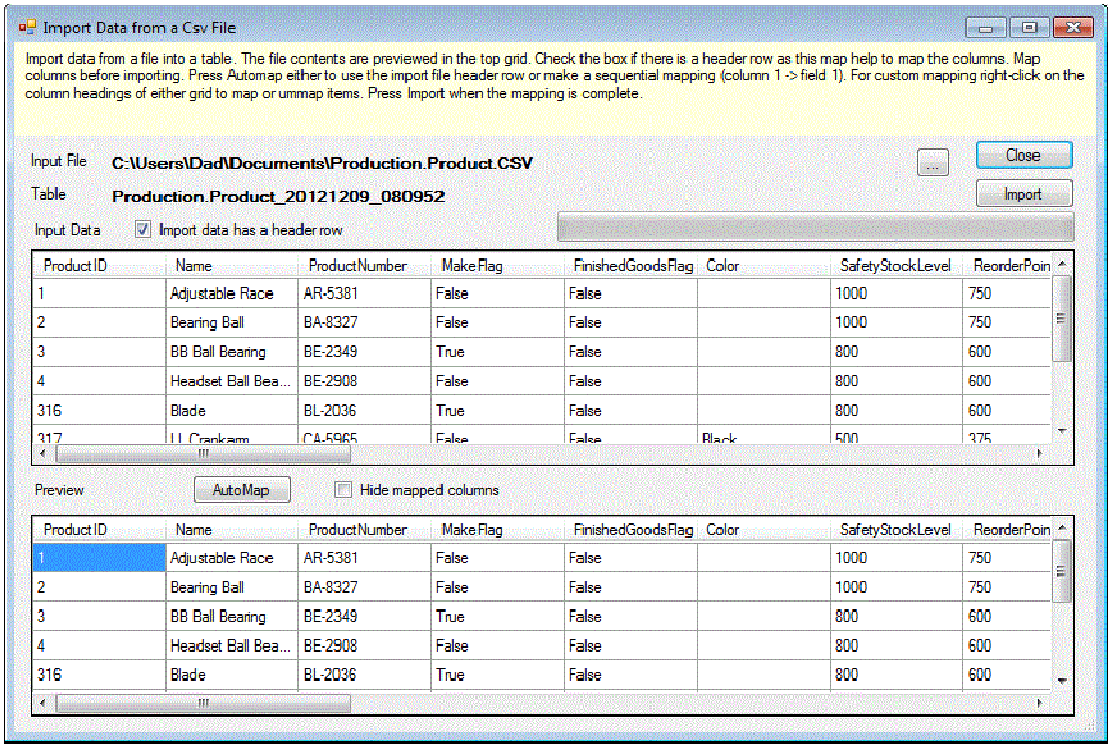
Generate the file. This is normally a pretty quick process unless you a generating a huge export. The Export button switches to a cancel button while the export is taking place. Note: for large exports the progress bar is not exactly linear so be patient while the data loads then watch it whizz through as it writes the file.

The format of the CSV file created conforms to the RFC 4180 specification for CSV. Data will only be quoted if it needs to be and data with line feeds in is treated according to the spec.

Import Data from a CSV file

The Import from CSV file dialog is displayed when you right-click on a table with no rows and select Import from the context menu. If you try to import to a table which already contains rows then you will get a warning message instead. This behaviour is in line with the DbViewSharp safety philosophy, which aims to prevent any accidental overwriting of existing data. Furthermore the developer of DbViewSharp does not believe in importing data directly into the target table. He would prefer to import it into a staging table and then use a stored procedure to clean it before updating the final destination table. Even furthermore, adopting this approach allows DbViewSharp to use the very fastest means of data-import.

If the table is empty you are prompted to select a file to import and once you have done this the dialog appears. You can select which input fields to match to which table columns although there is an automap feature that does this in simple cases.



Input File

Confirms the file you have just selected. If for any reason this is incorrect then press […] to reselect.

Table

Confirms the table into which to import the data. This cannot be changed from within the form. If you have selected the wrong table just close the form, select the correct table from the grid and try again.

Input Data Grid

The upper grid displays a sample of the first 10 or so lines of data to be imported. Use this to confirm that the data is what you expected and if necessary to map it the Preview table. More on mapping in the AutoMap section.

[x] Import data has a header row

Check or uncheck this box if it is incorrectly calculated by the program. When checked the first row is removed from the grid to become the Column headers. The rules used to determine whether the first line is likely to be a header line as follows:

a. There is no empty cell
b. No cell evaluates to a valid integer, number or date.
c. In at least one column the cells from rows 2-10 all evaluate to a valid integer, number or date or are empty.

The most likely cases for the program to guess incorrectly are for all character data where it will guess that there is no header row. The other problem area would be where there is more than one header row. The current import routine cannot process files with more than one header row correctly.

Preview

The lower grid is the preview grid. It shows the fields in the table and with data from any mapped input column. On entry if the routine matches header row data with fields in the table the columns will be pre-populated otherwise you need to take some action to perform mapping before any data will be imported.

[AutoMap]

In a number of cases it is possible to guess which table fields should be populated by which input columns. For example where data is exported from a table by DbViewSharp in one database for import into a table with the same schema in another then Automap will be applied before the dialog is displayed. If the import file has a header row then the data in this row will be matched to the field names in the table. If it does not have a header row then column 1 of the import data will be matched to column 1 of the table and so on until either the import data columns or the table fields run out.

There may be occasions where manual mapping is necessary. In these cases it is achieved by right-clicking on the column headers in either the Input Data grid or the Preview grid, whichever is more convenient. If a column is already mapped the only option you see will be to unmap it. If it is unmapped then you can select from the list of unmapped columns from the other table. Manual mapping is therefore not very sophisticated at present so it may be a better strategy to prepare your input file before this stage.

[x] Hide Mapped Columns

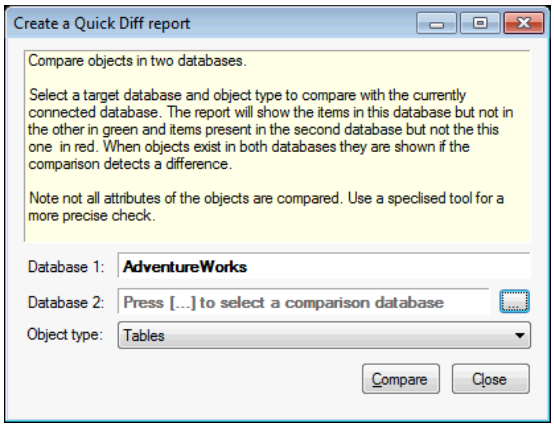
? Dubious Merit ? When importing to a table with a large number of fields this may help to manage the mapping process if necessary. Note also that the dialog is expandable.

[Import]

When all the columns are correctly mapped press import to start the process. This is normally quite quick although for very large files there might be delay. There is a progress bar to show how the import is going and this button becomes a Cancel button while the import is proceeding. When importing data there is always a risk that the input data is not valid for the receiving column in the table. DbViewSharp makes no check on data-format; instead it will check for and report any errors the Sql drivers return. It is your responsibility to interpret these errors and fix the data if you can.

Create a Quick diff Report

The Quick diff function will scan either table schemas or object source code and report differences in a report grid. It is accessed from the grids of the object types that can be compared. Unlike the commercial or specialised alternatives there is no facility to company all the object types and not all potentially comparable object types are selectable. The rationale for including an incomplete feature like this is as follows. Sometimes an application will be working correctly in one environment and not in another. The process of finding out why involves searching for differences in all of the components involved. Most programs are now developed and deployed using a version control system and installer programs. While it is possible to script and upgrade database schemas by the same process the practice is not so widespread. Furthermore a database schema can be modified so easily by typing commands directly into a query program that there is always the risk that a quick fix will bypass the scripting. The upshot is that it is quite easy for two databases that should be the same to contain subtle differences.



Database 1 [ ]

Confirms the source database for the comparison

Database 2 [ ]

Displays the target database you have selected or a prompt for you to make a selection

[…]

Opens the connection dialog for the selection of the target database. If it not already in the list you can create a new entry at this point.

Object Type [V]

Select the objects to compare in the two databases. Initially it is set to the object grid you right clicked on, but it is permissible to change this selection.

[Compare]

Run the report and display the results. The grid makes use of a status columns and colour to describe the differences. It is intended to describe the steps required to update the target to the same state as the source.

- Status +, row is green. The object is present in the source, absent in the target.
- Status -, row is red. The object is present in the target, absent in the source. Assuming the source database is correct then this indicates an object that might be dropped.
- Status >, row is blue. This is data from the source database for an object that exists in both databases, but contains differences.

- Status <, row is black. This is data from the target database for an object that exists in both databases, but contains differences. The expectation is that this data should be replaced by the corresponding blue row data above.

In the table schema comparison the cells that differ are highlighted. In particular where a feature is added (eg new foreign key) the highlight is green. Where a feature is removed (eg an index is in the target, but not in the source) then this is displayed in red. Incidentally indexes are features that have a tendency to change ad-hoc as performance is tuned.

! New table

This appears in response to selecting the New table context menu option on the tables grid. That option is only available when you set the database in development mode. As noted elsewhere this is not fully-featured covering all types and options for creating a table in Sql Server. Instead it is designed for ease of use.

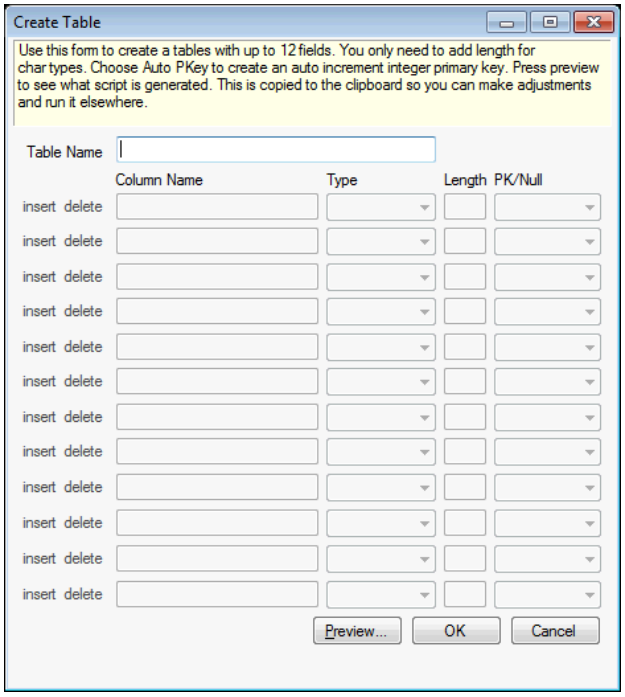


Table Name [ ]

Enter the name of the new table. You can enter a schema prefix here if necessary. Once you enter a table name the column definition controls are enabled

Column Name [ ]

Enter column names in the order you want them to appear. As you enter a new column name it enables the controls for the next .

Type [V]

Select from the list. The database types are ordered with most common at the top.

Length [ ]

The length control is only enabled for character fields. Enter the maximum length. The keywork MAX is not supported. See below for the workaround for this.

PK/Null [V]

Use this option to specify the primary key, auto-incrementing primary keys and non-null columns. Auto-increment configuration other than (1,1) is not supported, neither is a non-primary key auto-incrementing key. The default setting for this control is to allow nulls in the column.

Insert Delete

Insert will create a space for you to add a new column in the middle of the existing list. Delete will remove an unwanted column definition

[Preview]

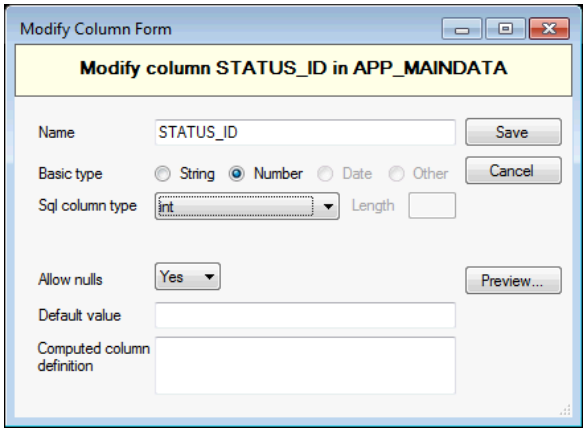
Validates your entry and displays the script to create the table in a message box. Crucially it also copies the script to the clipboard allowing you to paste the text into a query window and manually adjust it to add any table creation options that the dialog does not support. This then is the workaround mentioned above to solve the problem of creating a varchar(max) column.

[OK]

Validates your entry and executes the command to create the table. Normally if the input validates ok the table will be created, but on occasion it will not. You are left to interpret the Sql error message and make the appropriate adjustments.

! Modify Column Form

This is a development-mode only dialog for changing a column in a table. The dialog enables many changes to a column to be specified, however some changes will not be achievable by the program, when the data does not match the new type or there are constraints on the field such as a foreign key.



Name [ ]

Confirms the name of the column being modified. You can rename the column by changing it here. Note the change will fail if a foreign key is defined on the column.

Basic Type ( ) String ( ) Number ( ) Date ( ) Other

Restricts the type that the column can be changed to. String or character types can only safely be converted to other character types ie varchar to char or nvarchar. Number can be converted to other number types or to character types.

Sql Column Type [V]

Select the actual new column type from this list restricted by the selection above.

Length [ ]

For character columns specify or change the maximum length of the string to be held.

Allow Nulls [V]

Set to yes or no depending on whether or not the column will be permitted to contain null data.

Default Value [ ]

Add or modify the default value to set when a null would be inserted into a row.

Computed Column Definition [ ]

Add or modify the computation if this is to be a computed column.

[Preview]

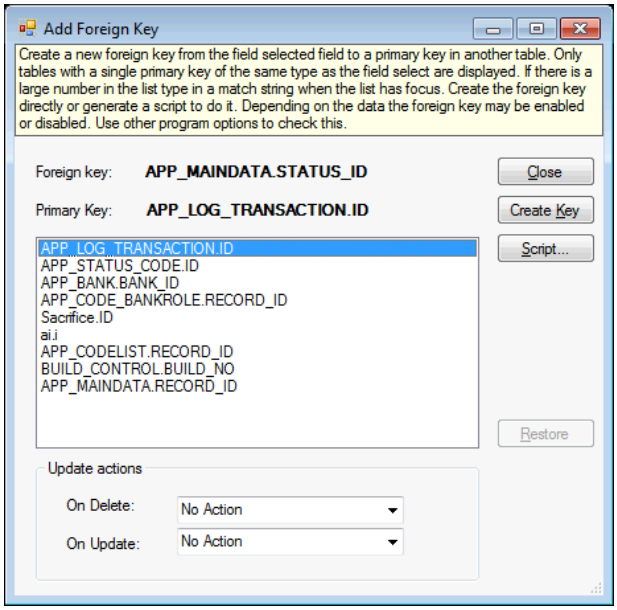
Display the script the will be run to attempt the change (attempt, because it is possible for the script to fail).

[Save]

Performs the script.

Add Foreign Key

This form is opened when you select Add foreign Key from the field/column grid. The column selected is the foreign key field, the form is for selecting the primary key to connect it to.



Foreign Key: Primary Key:

Confirm the current selections. The foreign field key cannot be changed as it was selected before the dialog was entered. The primary key display will change as the selected item in the field list below is changed.

Field list

This list contains all of the single primary key fields in the database that match the type of the field selected for the foreign key. If no matching primary key field exists (eg you select an Xml field), then this list will be empty and you cannot proceed.

Update Actions

On Delete [V]

Select the action that the foreign key initiates when a row is deleted. Unless you are familiar with the options you are recommended to keep to No Action.

On Update [V]

Select the action that the foreign key initiates when a row is update. Unless you are familiar with the options you are recommended to keep to No Action.

[Create Key]

Attempts to create the foreign key according to the selections made.

[Script]

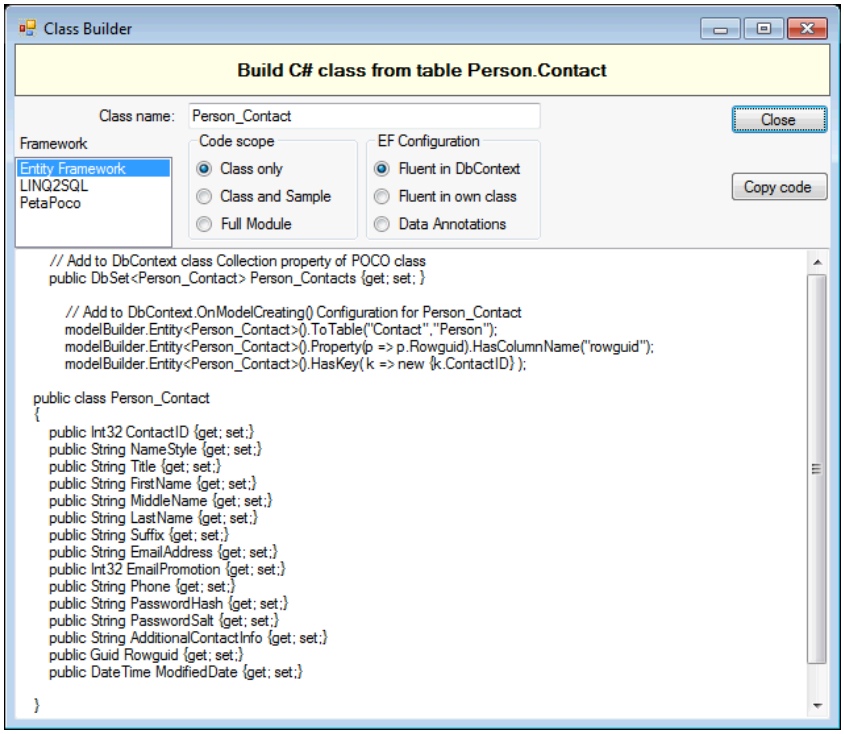
Shows the script that would be run from the selections made if you were to press [Create key]

[Restore]

You can filter the primary key list as you can other lists, by pressing letter keys. This button restores the display to the full list.

Class Builder

This dialog will generate a C# class from the selected table for use in one of three ORMs: Entity Framework, LINQ2SQL and PetaPoco. There are a number of choices for code generation depending on various scenarios.



Class name: [ ]

The class name is generated from the table name. The conversion will apply various conventions which you may want to override, hence you can change the default name provided. Conversions include: reducing all capitals to CamelCase; removing underscore from the name; except where a table contains characters invalid for a C# class name in which case the characters are replaced by underscore (see the screenshot above for an example).

Framework list

Shows the target ORM framworks. Entity Framework and LINQ2SQl should be familiar and are Microsoft offerings. PetaPoco is a lightweight ORM that the author is aware of.

Code scope

( ) Class only

( ) Class and Sample ? Dubious merit ?

( ) Full Module

There are three options for the scope of code generation. Only two are recommended. Class only generates code for adding to an existing module. Entity Framework will generate additional code depending on the EF configuration selection. Fluent configuration options separate the configuration code from the class definition. Even the data-annotations option will generate a property for inclusion in the DbContext class.

Full Module generates code for a standalone program that tests the class and configuration. You should be able to create a new console application in Visual Studio, and replace the default Program.cs file contents with the code generated by this option and have a working program (note

1: you will have to add references to Entity framework dlls; note 2: you will have to add the PetaPoco CS source file to the project).

Class and sample generates a class and the sample code to test it. It falls between the two other options and as such does not work very well as it generates too many code fragments.

EF Configuration

( ) Fluent in DbContext

( ) Fluent in own class

( ) Data Annotations

These options apply only when you have selected to generate a class for Entity Framework. There are at least three strategies for mapping the database table the class divided between the fluent approach and the data annotations approach. The former uses code separate from the class definition itself, while the latter sets attributes on the property values in the class (the same approach as LINQ and PetaPoco). All work, but the fluent configuration apparently implements more configuration options. You are recommended to research elsewhere on the differences.

[Copy code]

As you change your selections the code is regenerated and shown in the main display section of the dialog. This button transfers the contents of the display to the clipboard.

KeyBoard Reference

Some care has been taken to ensure that DbViewSharp can be used without a mouse. The application uses menu short-cuts and dialog accelerator keys. However some keyboard functionality is not visible. This section will describe it.

Main Window

The following paragraph describes a DbViewSharp session using the keyboard before listing all of the keys available.

Ctrl-O opens the connection window. A few letters typed in cuts down the connection list sufficiently for the right connection to be shown. Up and down arrow keys change the current connection to the target connection. Pressing the enter key connects. After the table list is displayed the grid has focus and arrow keys will allow navigation around the grid. To search the table list just start entering the letters of the text to search by; this will switch focus to the search bar. Use the Enter key to execute the search. After the search completes focus returns to the grid. To view the data in a table use the arrow keys to navigate to the row containing the table name and press Enter. To return to the table list grid press Backspace or Escape. To access the context menu associated with the current table press F10.

Below is the complete keyboard reference for the elements of the main window

- When only a grid is displayed (ie. The ad-hoc query window and field selector are not visible) then the tab key will cycle along the search bar to the grid. Since the grid uses the Tab key for internal navigation use Ctrl+Tab to move to the next control, in this case to cycle back to the search box.
- A short cut to jump from the grid to the search box is simply to type in the letters of the search term. The exception to this rule is when editing data. In this situation use Ctrl+F to move from the grid to the search box.
- Ctrl-Q will open the adhoc query window. When navigating from the grid to the query window focus is placed on the movable bar between them so the keys are Ctrl+Tab (to divider), tab to query window. In reverse use Shift+Tab, Shift+Tab.
- In the query edit window use Ctrl+Tab to enter a tab in the text box and Tab to move from control to control (Ctrl+Tab in the adhoc query result grid).
- Ctrl+F4 will close the query window.
- In the field selector window use accelerator keys and the tab keys as for a normal dialog. When tabbing away from this window the focus returns to the search box.
- The field list can be reduced by typing the search term directly when the field list has focus. Restore the full list with Escape.
- Alt-O will close the field selector (being the accelerator key of the [Close] button.
- Where there is more than one grid tab the keys Ctrl+, (<) and Ctrl+. (>) will cycle through the grids order. Ctrl+F4 will close the current tab.
- In a grid the arrow keys will move the current cell, as will Tab. Ctrl+arrow keys move to the limits of the grid (as do Home and End), Shift+arrow keys select groups of cells.
- In the grid [F10] displays the context menu. Ctrl+F2 or Alt+E,E switches the table data grid between editable and read-only (if the table has a primary key).

Modify DataView Form

This form has been given an enhanced keyboard interface to make moving items from one list to another easier.

- Left- and Right-arrow will cycle from one list to the other. If a list is empty it is skipped.
- In the Fields list box typing in the letters of a search term will filter the field list. Escape will restore the full list. Enter or Space will select fields to the DataView fields list.
- In the DataView fields list (ie. the columns from the table to be displayed), Space will add the selected field to the sort list, Delete will remove it. Shift+Delete will remove all items beneath the selected one. Keys + and – will move the selected field up and down the list, as will Ctrl+up-arrow and Crtrl+down-arrow. Ctrl-F will add the selected field to either the Text filter list or the Number search list depending on the type of the selected field (note: for some field types it won't be added to either list).
- In the Sort list Delete will remove the selected field, Space will toggle between ascending and descending sort order, Ctrl+up-arrow or + will set the field to ascending sort order, Ctrl+down-arrow or - will set the field to descending sort order.
- In the Text filter and Number search lists Delete will remove the selected item.