Exporting a Table

Only in DbVisualizer Pro

This feature is only available in the DbVisualizer Pro edition.

You can export an individual table using the Export Table assistant.

To export a table:

1. Select the table node in the **Databases** tab tree,
2. Open the **Export Table** dialog from the right-click menu,
3. Select an **Output Format**, **Output Destination**, and **Options**,
4. Click **Export**.

Output Format

You can export tables in one of these formats: **CSV**, **HTML**, **SQL**, **XML**, **XLS** (Excel), or **JSON**.

For the **SQL** and **XML** formats, you can choose to export the DDL and the table data; the other formats only export table data.

You can control whether to use delimited identifiers and/or qualified names by default in the DDL and INSERT statements generated for the SQL format, and you can override the defaults in the Export dialog for a single export operation.

Output Destination

The destination can be one of:

- a file,
- an open or new SQL Commander tab, with options for where in an open SQL Commander to insert the result,
- to the system clipboard.

Options

The Options are contains options common to all Output Formats at the top, followed by options for the selected format.

For the SQL and XML formats, you can choose to export the DDL, the DDL for indexes for a table and the table data: as INSERT statements for the SQL statement or in one of three XML formats.

For the XLS format, you can choose to export table data as either regular Binary Excel or OOXML for Excel 2007 and later.

Most formats also let you specify other options, such as delimiters, title and descriptions. Just select an Output Format to see which options are available. All options are described in the context of the @export command, as the Export dialog is just a GUI for the command.

You can adjust the **Data Formats** specifically for the exported table data. By default, the formats defined in **Tool Properties** are used, but sometimes you need to export dates and numbers in a different format because you intend to import the data into a different type of database.

If you are exporting table data in the SQL format from one database type (e.g. Oracle) to import it in a database of a different type (e.g. PostgreSQL) by executing the generated script, you need to be aware of differences in the literal formats for Date, Time and Timestamp data. If you connect to the other database using a JDBC client like DbVisualizer, you can select the **JDBC escape** format for these data format. This generates literals that the JDBC driver converts into a format the target database can interpret.

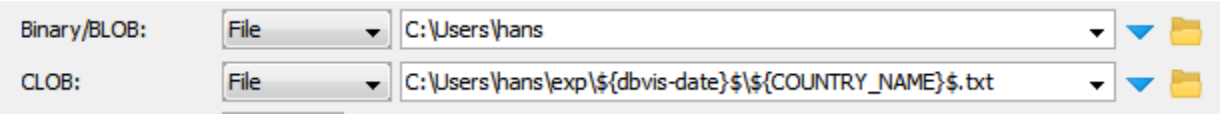
In the **Data Format Settings** dialog you can also specify how to quote text data and how to handle quotes within the text value.

Using Variables in Fields

You can use some of the pre-defined DbVisualizer variables (`${dbvis-date}$`, `${dbvis-time}$`, `${dbvis-timestamp}$`, `${dbvis-connection}$`, `${dbvis-database-type}$` and `${dbvis-object}$`) in all fields that hold free text (e.g. title and description fields) and as part of the file name field.

Exporting Binary/BLOB and CLOB Data

You can use the export assistant to export Binary/BLOB and CLOB data. You enable this by choosing **File** as the data format for **Binary/BLOB** and/or **CLO** data. Optionally, you can specify the directory or filename pattern for the data files. If you do not specify a directory or filename pattern, the operating system's default directory for temporary files (e.g. *C:\TEMP* or */tmp*) is used.



A pattern is a path with fixed and variable parts, where the variable parts are expressed as DbVisualizer variables, e.g. *C:\Users\hans\exp\${dbvis-date}$\${COUNTRY_NAME}$.txt*. You can select variables to insert at the current caret position in the path field from the dropdown. The variables that can be used are the predefined DbVisualizer variables plus variables for each column in the table, e.g. `${COUNTRY_NAME}$` in the example above. Another special variable that can be helpful here is `${dbvis-column-name}$`. If a table has multiple BLOB or CLOB columns, you can use it in the pattern to export the columns to separate files, e.g. *C:\Users\hans\exp\${dbvis-column-name}$.txt*. All variables that can be used are listed in the menu that is displayed when you click the blue arrow to the right of the field. You can select a variable from the menu to insert it in the pattern field at the care position.

The data for each individual value of this type is then exported to a separate file and a DbVisualizer variable referencing the file is inserted in the main export file.

Saving And Loading Settings

If you often use the same settings, you can save them as the default settings for this assistant. If you use a number of common settings, you can save them to individual files that you can load as needed. Use the Settings button menu to accomplish this:

- **Save as Default Settings**
  Saves all format settings as default. These are then loaded automatically when open an Export Schema dialog
- **Use Default Settings**
  Use this choice to initialize the settings with default values
- **Remove Default Settings**
  Removes the saved defaults and restores the regular defaults
- **Load**
  Use this choice to open the file chooser dialog, in which you can select a settings file
- **Save As**
  Use this choice to save the settings to a file

- **Copy Settings to Clipboard**
  Use this choice to copy all settings to the system clipboard. These can then be pasted into the SQL Commander to define the settings for the @export editor commands.

Other Ways to Export Table Data

- Export all or selected tables with the Export Schema assistant,
- Export a subset of the table data with the @export command.